SHOPPING RECOMMENDER SYSTEM FOR UNIVERSITY STUDENTS

BY

LIM WOON SHENG

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF INFORMATION SYSTEMS (HONOURS) BUSINESS INFORMATION SYSTEMS

Faculty of Information and Communication Technology

(Kampar Campus)

JAN 2024

UNIVERSITI TUNKU ABDUL RAHMAN

REPORT STATUS DECLARATION FORM

Title: SHOPPING RECOMMENDATION SYSTEM FOR UNIVERSITY STUDENT

Academic Session: JAN 2024

I LIM WOON SHENG

(CAPITAL LETTER)

declare that I allow this Final Year Project Report to be kept in
Universiti Tunku Abdul Rahman Library subject to the regulations as follows:

1. The dissertation is a property of the Library.
2. The Library is allowed to make copies of this dissertation for academic purposes.

Verified by,

Sheng [illegible]

(Author's signature)	(Supervisor's signature)
Address:	
697, Lorong Cengal 16/1	
Taman Ampangan Seremban	Dr Abdulkarim Kanaan Jebna
70400 Negeri Sembilan	Supervisor's name
Date: 24 April 2024	**Date**: 24 April 2024

Universiti Tunku Abdul Rahman			
Form Title : **Sample of Submission Sheet for FYP/Dissertation/Thesis**			
Form Number: **FM-IAD-004**	Rev No.: **0**	Effective Date: **21 JUNE 2011**	Page No.: **1 of 1**

FACULTY OF INFORMATION AND COMMUNITATION TECHNOLOGY

UNIVERSITI TUNKU ABDUL RAHMAN

Date: 24 April 2024

SUBMISSION OF FINAL YEAR PROJECT /DISSERTATION/THESIS

It is hereby certified that ***Lim Woon Sheng*** (ID No: ***20ACB03109***) has completed this final year project entitled "*Shopping Recommender System for University Students*" under the supervision of Dr Abdulkarim Kanaan Jebna (Supervisor) from the Department of Information Systems, Faculty of Information and Communication Technology.

I understand that University will upload softcopy of my final year project in pdf format into UTAR Institutional Repository, which may be made accessible to UTAR community and public.

Yours truly,

Sheng

____
(*Lim Woon Sheng*)

*Delete whichever not applicable

DECLARATION OF ORIGINALITY

I declare that this report entitled "**SHOPPING RECOMMENDER SYSTEM FOR UNIVERSITY STUDENTS**" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : Sheng

Name : Lim Woon Sheng

Date : 24 April 2024

ACKNOWLEDGEMENTS

I would like to express my sincere thanks and appreciation to my supervisors. Firstly, is a previous supervisor, Dr Mohammad Dalvi Esfahani who firstly providing a such idea on recommender system about. However, due to unfortune reason, I have to be reassigned a new supervisor. Who I would like to appreciate much. Dr Abdulkarim Kanaan Jebna who has willingly given me this bright opportunity to engage in a recommender system project and accept me as final year project student to provide guidance throughout the project development process. And keep provide me guidance and encourage me through this journey. It is the first and every opportunity that I have my practical on the field of knowledge that the flied I very interest in which are data analysis, machine learning, recommender system, and sentiment analysis. A million thanks to my two supervisors.

To a very special friends in my life, Voon Jun Hao, Kho Tian Song, and Chong Xin Yee, for their patience, and spending time for me, and help me to reduce the stress. And special friends I met during univeristi, Lim Kai Yi, Kuit Ying Qian and Ng Chun Ming who play a role of listener that allow me to express my negativity out. Especially, so many commitments I have been taking in the university. And they are the peoples who are willing to stay behind me and give support to me. Finally, I must say thanks to my parents and my family for their love, support, and continuous encouragement throughout the program (Bachelor of Information System, Business Information System) and my final year project (Shopping Recommender System for University Students) that I take in this university. Where, my family always acknowledge that and give support without asking any questions.

ABSTRACT

This project is a development-based project that referred to e-commerce platform to buildan system that recommend a physical shop and store. This recommendation area focuses on sentiment analysis by using user’s review and rating to classify in to three categogy which are negative, neutral, and positive. This project idea came from the market us lacking a platform that focusing recommending a physical shop and store. While most of the platform is focusing on online shop and store, only certain number of a physical shop and store will have an online shop in the platform such as Lazada, and Shopee. Hence, this project is aimed to design a application that focuses on recommending physical shop and store by suing sentiment analysis on review and rating. The project used rapid applciation development methodology and agile development methodology for entire project development. Which provide fexlibility, agility, and effectiveness to the project’s process. For the system development itself, it is using methodology know as Cross-Industry Standard Process for Data Mining. It is start from business understanding until deployment phases with total 6 phases. The project developed an application that integrated with a sentient analysis model that used BERT model to train. It is able to generate the result for the review and rating. The application able to provide different criteria and requirement for user to select to perform filtering and ranking for shop and store recommendations. Lastly, the projectmain to contribute to the axisting platform and software that apply sentiment analysis in terms of the methods on utilization of the sentiment analysis result for filtering functions.

TABLE OF CONTENTS

FYP2 CHECKLIST

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

CBRS	Content-Based Recommendation System
CFRS	Collaborative Filtering Recommendation System
ARBRS	Association rule-based Recommendation system.
UBRS	User-based recommendation system
KnBRS	Knowledge-based recommendation system
LBS	Location-Based Services
LDA	Latent Dirichlet Allocation
TF-IDF	Term Frequency-Inverse Document Frequency
BERT	Bidirectional Encoder Representations from Transformers
TP	True Positive
TN	True Negative
FP	False Positive
FN	False Negative
CRISP-DM	Cross-Industry Standard Process for Data Mining
CPU	Central Processing Unit
XML	Extensible Markup Language
ID	Identification
ONNX	Open Neural Network Exchange
TF2ONNX	TensorFlow to Open Neutral Network Exchange
SEP	Separation
CLS	Classification
H5	Hierarchical Data Format
TFLITE	TensorFlow Lite
MB	Megabytes
GRU	Gated Recurrent Units
BiGRU	Bidirectional Gated Recurrent Unit

Chapter 1

Introduction

This chapter will focus on the introduction of the project and its background. Including problem statement, motivation, objectives, project scope, contribution, and report organization. Which related to the recommender system and sentiment analysis.

1.1 Background

According to [1], in this generation, e-commerce has grown to simplify and fasten the purchasing and selling process with different functions in the application. [2] mentioned that e-commerce combines online payment, transaction, and business activities including retail, warehouse, marketing, and others. E-commerce can refer to Business to Customer (B2C), Business to Business (B2B), Customer to Customer (C2C), and Business to Government (B2G), among other things. Consequently, e-commerce is used by several parties. For example, local company in Malaysia will perform online procurement. For that, supplier announce the quotation list for the consumer. Furthermore, [3] stated that e-commerce allows users to research the information of the product and sellers online. It brings a significant advantage to the public: purchasing and selling actions can be done anywhere and anytime with any device. And it allows the sellers to build their business and brand globally. E-commerce also provides a function of analyzing different products from different sellers based on various criteria. Such as the review, rating, price, brand, and others. Based on this, the system will rank the items and display and recommend suitable items to users. It also stated, the e-commerce website created nowadays, the information state in the website are having possibility of fake information, hence people are not consumer to trust them. For that, consumer is required to pay more time and cost for gathering information to analyze which information are correct. This issues mostly happen in C2C e-commerce. Where the seller or supplier in B2C e-commerce are having enough trustworthy brand, hence there are less case of scam happened compared to C2C e-commerce.

[4] mentioned that B2C e-commerce focuses on the selling and buying process and customer engagement. Moving on, Customer engagement can be done through different method in websites, social media, and so on, where it plays an essential role between the sellers and buyers, as its various significant information to the public. There are two main processes

that the consumer will go through. Firstly, consumer will engage the seller or shop by referring to the information stated by the seller at the website, social media accounts, e-commerce account and others. For example, the way of delivering message that seller chooses. Such as, including text, picture, video, and others to convey message to increase the effectiveness of customer engagement. The other, the customer service. A 24 hour per days services will giving customer a good experience. The faster the seller solve the customer concern, the higher satisfaction from customers. By applying artificial intelligence chat box in e-commerce will also help to increase the satisfaction of customers as well as reduce the burden of the workers of the shop to handle the cases. For that, the important information suggested to utilized are the reviews and rating. After analyzing them, this information bring a good reference for other customer, where a positive reviews and rating bring high possibility on success purchased order by customer. On the other side, if a shop or product are having many negative reviews and rating. It definitely will not attract customer.

There are reasons why e-commerce is a famous and successful tool for the public to purchase the item. Such as it includes different information for the user to refer to. For example, ratings, user reviews and comments, brands, prices, etc. [1] mentioned that reviews and comments play a crucial role in online purchasing, as buyers will read them to consider purchasing or not. This is because reviews affect buyers' consumption psychology and the decision-making for purchasing, as stated in [5]. According to [6], the e-commerce system consists of a function on the ranking, which uses various information and presents the output to users. And these order features consist of two parts: qualitative and quantitative information. Quantitative data, such as user reviews and comments, are challenging to study for ranking and filtering purposes. Hence, the system consists of machine learning and data mining techniques, which can achieve the objective of sentiment analysis and recommendation with filtering and ranking features. For example, model-based and memory-based filtering. Which will study and learn the customer's background and behaviour and recommend the items automatically. Or customers can use a filter function to filter out the product, and the system will recommend the product or seller based on the filter criteria mentioned in [7].

Furthermore, a more advanced recommendation system plugged in sentiment analysis of customer reviews and comments, which analyses the reviews and comments and segments the item, product, and seller into different groups. Not only that, but recommender system also refers to the sentiment analysis on reviews and rating. By analyzing the user satisfaction known as the reviews and rating. Which act as an important source for the customer to refer. Hence, the more positive reviews and high rating will definitely attract more customer; however, the more negative and low rating will cause a business loss. And this analysis will not only provide the benefits of ranking items from most quality to bad quality but also let sellers study the research to improve their product's quality, improving customer satisfaction as stated in [2]. Moving on, [6] noted that the filtering function could improve customers' purchasing experience. Where it acts as a tool for user to use to get a reference. By filtering or ranking the shop with using different criteria such as review, rating, prices, and other. It will help on user to perform decisions making.

Categorizing the item is always challenging for an e-commerce system. Where it gathers many sellers and billions of products inside the platform. [9] stated that, to ensure customers' satisfaction and experience, the system organizes and categorizes items into a taxonomy of fine-grained categories. It is a fundamental feature of an e-commerce system as it can simplify versions of the types of products. Moreover, the system collects the basic information of buyers to perform analysis and prediction—based on records to recommend products or shops. Or search history in another connected platform will be one of the pieces of information to let the system study and perform recommendations. It is known as personalization marketing and recommendation.

However, this project focuses on building a recommendation system for in-store shopping for university students, which has similarities and differences with e-commerce. This is due to the market lacking an application to recommend to the public to shop in-store shopping with the function that e-commerce applications have. Such as recommending shops based on price, etc. This project will focus on sentiment analysis and using the analyzed data for filtering features to ensure the user can filter the shop list based on various criteria, such as reviews, rating, price, distance, and others. Yet, this project use machine learning to perform sentiment analysis. It is known as text mining, which analyses the reviews and comments.

1.2 Problem Statement and Motivation

Nowadays, many features and functions are applied to the application. However, there are a few gaps faced by the in-store recommender system, and some exist in e-commerce applications. Effectiveness of filtering features with sentiment analysis on essential factors, such as consumer reviews and comments. [10] mentioned that reviews and words play a crucial role in online purchasing by using machine learning and performing sentiment analysis and classification on the items or the sellers. However, the system did not apply automation in it, such as after the classification and segmentation of the sellers, the result will act as a piece of important information to the recommendation and ranking system. For example, the buyer searches an item using a filter function, such as price and brand. And the system will automatically add a significant factor which is the results of sentiment analysis of reviews, and rank it based on the outcome. Such as from positive reviews to negative reviews of a sellers.

This project aims to design a better recommendation system for an in-store purchasing application. I am motivated by this project is due to in the market are full of e-commerce application. There are a few applications related to in-store shopping recommendations. However, it did not apply a perfect and complete function as an e-commerce application. Also, I believe university student faces an issue as most of them move from their hometown to a new area. And I am struggling with where they can buy their lifestyle product for their university journey. I was moved by this concept; therefore, I made the choice to give this project my full attention. From a different angle, it is also as a result of the skills and methods required for this system, including data mining, text mining, machine learning, and others. I have a high curiosity and am passionate about learning and practicing it. And due to that, I decided to focus on the shops located near by the university. And using the technique to design a system that can let the user filter the shops or items effectively.

1.3 Objectives

The project aims to propose a new shopping recommendation system for a university student focused on in-store shopping that slightly similar with e-commerce application. The primary goal of this project is to create a recommendation system with utilizing the sentiment analysis result.

1. To develop a system that can use the result of sentiment analysis of customers' reviews and comments to classify the shops into different groups.
2. To develop a system that utilize the result of sentiment analysis to improve customer purchasing experience and satisfaction.
3. To develop a system that achieve at least a minimum 80% average of accuracy to the unseen data which is testing dataset.

1.4 Project Scope and Direction

The project scope includes various aspects to form a complete in-store shopping recommendation system. This system's targeted audience is university students, and in this project, the system will focus on University Tunku Abdul Rahman as an example. And to target all shops and sellers within the surrounding area of Kampar, Perak, for UTAR students. The project's scope is sentiment analysis can analyze online customer reviews and ratings. And the result will be used in filtering functions. It will also be used in the application interface to provide a reference role for the user to advise in their decision-making. The technique might use, such as a Content-Based Recommendation System. It is to get the preference of item types from the user, and the system will perform filtering and recommendation based on preference. The second technique is the Collaborative Filtering Recommendation System, which will perform filtering and recommendation entirely based on the user's history and items or shops. Such as purchase history, search history, and so on. However, this project will use hybrid filtering techniques that combine both mentioned techniques.

Next, this project only targeted not more than three categories of items or shops, such as food and groceries category product that includes food, beverage and so on. Examples of shops such as Lotus, Seven Eleven, etc. This project's system requires collecting basic customer information such as address and financial information. This project is designed to recommend shops and items with a ranking based on the user's filtering criteria. And it should include basic details of the shops with a feature of ordering the items. This system allows users to decide whether to purchase online with or without delivery and pay in-store. It also allows users to view product information. These systems should include an algorithm for filtering and ranking purposes and a simple search engine function to search products or shops.

1.5 Contributions

This project's contribution can be separated into two parts: theoretical and practical, for the theoretical is the knowledge of this project. I hope this project may spark an idea for other researchers to seek an innovative feature to add to related work, such as the theory behind the technique, such as test mining to perform sentiment analysis. For the practical part, I hope the outcome of this project may be an improvement suggestion or reference for other similar applications. From the shop owner's perspective, they may refer to the outline and the information needed for the application if the shop owner would like to use this application as one of the marketing strategies. As learned the design of the shop's information display in the application to ensure it is attractive. Moving on, from a customer perspective, we can be more active in providing reviews and comments for the item or the shops, as this project will focus on user friendly to get information from the customer. As a result, this project's primary contribution is the novel way it uses the results of the sentiment analysis for recommendation-making. Additionally, greater filtering functions for the user are provided based on the sentiment analysis result, as well as more sophisticated filtering methods for the system.

1.6 Report Organization

This project report consists of total 7 chapters which are Chapter 1 Introduction, Chapter 2 Literature Review, Chapter 3 System Methodology, Chapter 4 System Design, Chapter 5 System Implementation, Chapter 6 System Evaluation and Discussion, and Chapter 7 Conclusion and Recommendation. First chapter, **chapter 1** included six main element which are background, problem statement and motivation, objective, project scope and direction, contribution, and report organization. Second chapter, **chapter 2** known as literature review which have research on recommendation system and sentiment analysis. It also includes research of previous work in sentiment analysis with previous study. Next, **chapter 3** focuses on the system methodology and the system design such as use case diagram, class diagram and so on. **Fourth chapter** is about system design. Four elements are being explained which are block diagram, flow diagram, draft design of application interface, system component specification, and system implementation planning according to the components. Moving on, **chapter 5** focus on the content of system implementation which starting from software and hardware setup, modelling and application development, implementation issues and challenges, and concluding remark. **Chapter 6** are the content of system evaluation and discussion. Firstly, there is a system performance definition with plan. It also included the testing setup and result. In this chapter, the project challenges and objectives evaluation will be discussed. Last chapter, **chapter 7** is conclusion and recommendation. Which according to the progress and result of the project, provide a conclusion and recommendations.

Chapter 2

Literature Review

This chapter will divide into two parts: the introduction of related techniques or models used and, previous work on some similar systems. To study on the background of recommender system and sentiment analysis. In terms of different techniques and method used by different systems. Eventually, by comparing and research different studies to decide the model for this project.

2.1 Introduction of Recommender System

According to [11], the recommender system idea was started in 1990 by Resnick and Varian. This system is an automation system applied in e-commerce to recommend products. And provide the product's information to the customer. The recommender system has used different filtering techniques (Refer to Figure 2-1-1). There are five recommender system types: content-based, collaborative filtering, association rule-based, user-based, and knowledge-based recommender system. In this system, the most famous system. These different recommender system types are used collaboratively in the application, such as three-dimensional recommendation methods focusing on content-based and collaborative filtering in an e-commerce framework. Moving on, [11] stated that item-based collaborative filtering is famously used in e-commerce applications as one of the techniques to perform recommendations. And all of them serve a different purpose. Content-based refers to the data given by the user, such as ratings. Collaborative filtering refers to past interactions between users, sellers, and products. Association rule-based refers to rules selection that applies in the recommendation system. And user based refers to a specific item that the user used to search and, based on that, predict and recommend similar or related entities to users. Lastly, knowledge-based refers to the system that will advise the user for decision-making.

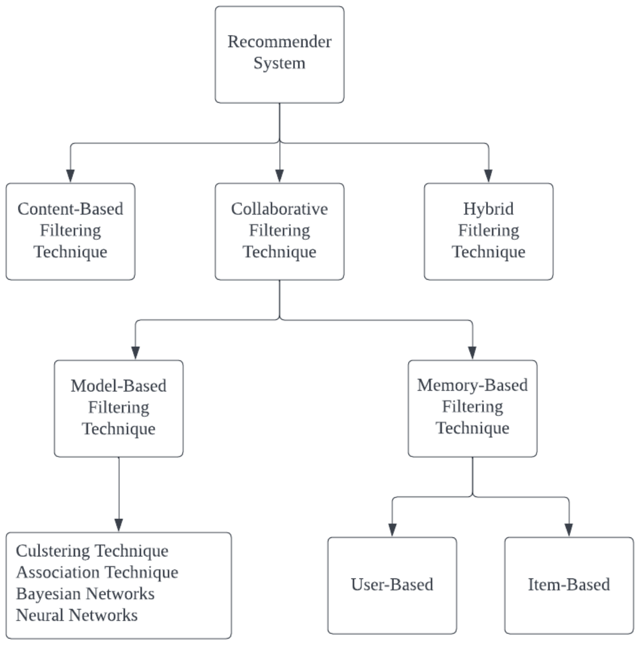

Recommender System
Content-Based Filtering Technique
Collaborative Filtering Technique
Hybrid Fitlering Technique
Model-Based Filtering Technique
Memory-Based Filtering Technique
Culstering Technique
Association Technique
Bayesian Networks
Neural Networks
User-Based
Item-Based

Figure 2-1-1 Different Techniques of Filtering

2.2 Introduction of Sentiment Analysis

According to [12], sentiment analysis can be performed using many techniques and methods, such as machine learning and lexicon based (Refer to Figure 2-2-1). It is also known as option mining. Yet, sentiment analysis has performed research on subjective tests such as reviews and comments that customers wrote down their feelings, experience and so on. And sentiment analysis is to analyze them and perform classification. And classified the reviews and comments of the shops and items into different representative groups. The process of sentiment analysis are finding, evaluating and categorizing the reviews to negative, neutral, and positive. As stated in [13], machine learning is classified into two main categories, which are supervised and unsupervised. And two sets of data are essential when performing sentiment analysis: a training set and test data. Training data is used to differentiate the characteristics and test data used in calculating research performance in terms of accuracy. At the same time, the lexicon-based approach does not require a dataset as this approach is supervised. This approach focuses on identifying the keywords in the text.

Furthermore, this paper concludes that the machine learning approach has better accuracy in analysis, while the lexicon-based do not require time for training. Hence, in real cases, these two approaches are practised to be used collaboratively to solve their weaknesses. According to [5], natural language processing can also perform sentiment analysis. Additionally, it is primarily split into two categories: fine-grained analysis and coarse-grained analysis. Coarse-grained analysis refers to text based on the model and machine learning, whereas fine-grained analysis relates to classification using the dictionary. The result of sentiment analysis is not just in the filtering function; it can also be used as an essential business factor that may affect business strategies.

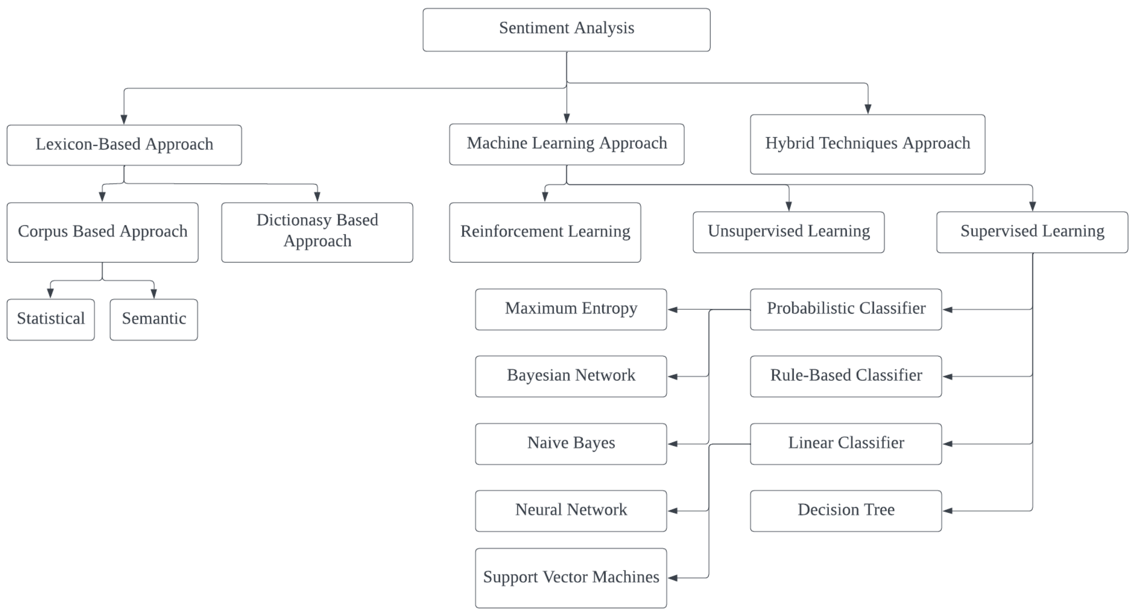


Figure 2-2-1 Different Approaches in Sentiment Analysis

2.3 Previous Work in Sentiment Analysis for Customer Rating and Reviews

According to [5] and [10], it studies and analyses customer reviews and comments. Besides machine learning, the research stated that the developers also used natural language processing to perform sentiment analysis. It is also known as option mining to segment related factors such as price, quality, performance, function, logistics, the experience of using, and others into different groups. Fine-grained analysis and coarse-grained analysis are the two primary divisions of sentiment analysis. And the findings of this study can aid customers in making wiser decisions while making purchases. Fine-grained research focuses on the corpus and dictionary of sentimental analysis. So, the coarse-grained study focuses on using machine learning and comment on the overall sentimental orientation of the text-based—keywords such as significantly, very and others. The researcher can draw conclusions about the emotions of the review—such as good, neutral, and negative emotions—by using them. Additionally, the outcome will be used to display an overall analysis result to the user. And this will help users to understand the product or shop quickly by directly showing the comprehensive common reviews and comments. According to [13], although machine learning offers a higher level of analysis accuracy than lexicon-based methods, in practical business cases these two methodologies are employed in tandem to address their deficiencies. In contrast, lexicon-based does not require time for training. Hence, in the actual case, these two approaches are practised to use collaboration to solve their weaknesses.

According to the other study, [13] stated that sentiment analysis is mostly dependent on three techniques: deep learning, machine learning, and sentiment lexicon. The sentiment lexicon integrated the emotions, modifiers, and particular terms of the customers' remarks to do analysis using sentiment normalisation and evidence-based combination functions. However, they first used the unigram mixture model to classify the emotional text. Some people added the degree adverb lexicon, network word lexicon, and negative word lexicon to the WKWSCI sentiment lexicon, which has a distinct function, to analyse the comment. A sentiment lexicon-based approach can provide a quality analysis result. But the cost of manual maintenance is much higher than another approach. Hence developer tends to use machine learning with an automation function on analysis. By doing sentiment analysis utilising a multi-modal joint sentiment model, SVM, and the k-nearest neighbour technique, it offers an automation function on analysis. However, it has a drawback that frequently requires manual feature selection. Deep learning also does not need human involvement. Additionally, deep understanding analyses the relationship between the target word of the comments and the surrounding words using a target-dependent convolutional neural network. Using a sequence model based on neural networks to conduct categorization. Deep learning is one of several sub-elements of machine learning. And to conduct sentiment analysis mostly using machine learning theory. Process of implementing sentiment analysis include many steps. Embedded layer, convolution layer, pooling layer, BiGRU layer, attention layer, and fully connected layer are the different layers that make up this process.

First, Embedded Layer. After the word in the reviews or comments refer to w_i, and sentiment weight refer to $sw_i\ or\ senti(w_i)$. Then, each word will be corresponding to one dimension. And ensure there is high dimensional in the word vector, the developers decided to use the BERT model instead of One-Hot. As the BERT model can describe more details and create more relationships between words where One-Hot is simple to perform but cannot describe complex relationships between phrases. And the word is converted into word vector where using Equation 2-3-1.

$$v_i^t = v_i * senti(w_i)$$

Equation 2-3-1 Weighted Word Vector Matrix

Next, Convolution Layer. Matrix V refer to the different word collected into the vector. $V = [v_1^t, v_2^t, \dots, v_n^t]$. Convolutions refer to the following Equation 2-3-2. The activation function ReLU is denoted by the symbol f, and the weight matrix is denoted by the symbol W R(k+m), where k and m are the height and width of the convolution kernel, respectively. The multi-layer neural network uses the ReLU function. The eigenvector matrix defined as, after the convolution procedure, Equation 2-3-3 will be implemented.

$$v_i^n = f(W.V_{[I:I+k-1]} + b)$$

Equation 2-3-2 Convolution Operation

$$V' = [v_1'', v_2'', \dots v_i'',, \dots v_{n-k+1}'']$$

Equation 2-3-3 Eigenvector Matrix

Moving on, Pooling Layer. The k-max operation will be implemented, which refers to the text sentiment analysis, as there usually will be few words or sentences in the reviews or comments. M refer to the dimensions of the vector 〖v_i〗^".And max is the maximum function. By using equations below.

$$x = [x_1, x_2, \dots, x_i, \dots, x_{m-k+1}\ ]$$

Equation 2-3-4 K-Max Pooling Operation 1

$$x = max\ (v_i{}'', v_{i+1}{}'', \dots v_{i+k-1}{}'')$$

Equation 2-3-5 K-Max Pooling Operation 2

In BiGRU Layer, the GRU model combines historical information with the current output. BiGRU represent forward GRU and reverse GRU, which refers to Equation 2-3-6 and Equation 2-3-7. By combining these two outputs of the equations, it refers to the hidden state output at time t, which is $h_t = [h_t'; h_t'']$.

$$h_t' = \overrightarrow{GRU}(x_t, h_{t-1}')$$

Equation 2-3-6 Forward GRU

$$h_t' = \overleftarrow{GRU}(x_t, h_{t-1}'')$$

Equation 2-3-7 Reverse GRU

Moving on, a sentence includes some words that can bring essential meaning and some that cannot, so in the attention layer, it will define the weight of different words in the reviews and comments. W stands for the weight matrix, b for the offset, and finally, the global context vector, or the parameter under study, is mentioned, u_w. By using Equation 2-3-8 to Equation 2-3-10 , sum of the vector of a input sentence S. $a_i$refers to the weight, and $h_i$refer to the hidden state output weighted.

$$u_i = tanh\ (W\ .h_i + b)$$

Equation 2-3-8 BiGRU Layer Output Calculation 1

$$a_i = \frac{e^{u_i^T.u_w}}{\sum_i e^{u_i^T.u_w}}$$

Equation 2-3-9 BiGRU Layer Output Calculation 2

$$X = \sum_i a_i . h_i$$

Equation 2-3-10 Feature Vector Representation

The fully connected layer, which is the final layer, classifies the input according to the function described in the equation below, where f stands for the sigmoid activation function, w for the weight matrix, and b for offset. If the result of this equation is close to 0, it means something bad; if it is close to 1, it means something positive.

$$Y = f(W . X) + b$$

Equation 2-3-11 Input Feature Matrix Classification

Lastly, after the analysis, the developer conducts a performance evaluation, refer to Table 2-3-1 and Table 2-3-2 explanation of definition for the terms use in the equations.

Equation Name	**Equation**
Accuracy	The proportion of correctly prediction among all prediction. $Accuracy = \frac{TP+TN}{TP+TN+FN+FP}$
Precision	The true positive prediction among al positive prediction. $Precision = \frac{TP}{TP+FP}$
Recall	The proportion of true positive among all actual positive prediction. $Recall = \frac{TP}{TP+FN}$
F1 Score	Mean of precision and recall. $F1 = \frac{TP+TN2*precision*recall}{precision+recall}$

Confusion Matrix		Actual Positive	Actual Negative
	Predicted Positive	True Positive (TP)	False Positive (FP)
	Predicted Negative	False Negative (FN)	True Negative (TN)

Table 2-3-1 Performance Evaluation for Sentiment Analysis

Term	**Definition**
TP	The number of positive reviews and comments is actually positive.
FP	The number of positive reviews and comments is actually negative.
TN	The number of negative reviews and comments is actually negative.
FN	The number of negative reviews and comments is actually positive.

Table 2-3-2 Definition of Terms in Performance Evaluation

Moving on, [14] stated the papers reviewed in [14], there are total 24 papers that show 10 papers prefer to use machine learning, lexicon-based method used by 7 paper, and other paper use both method in analysis. Most of the technique used in the analysis are Sentiwordnet and TF-IDR. And these two techniques rely on the quality of resources. For example, when the data input or the reviews is having high complexity which will affect the effectiveness of the techniques. These techniques are categorised in lexicon-based methos or unsupervised learning. While, for machine learning techniques, SVM and Naïve Bayes Model are frequently used in sentiment analysis. However, it could perform poorly due to the quality of data input. For example, spelling mistake of reviews. Which cause the model need a huge dataset and time for adaptation and learning. Other than machine learning method, there are also deep learning method. [15]-[17] mention that the most common techniques are long-term memory (LSTM), recurrent neural networks (RNN), and convolutional neural networks (CNN). Deep Neural Network (DNN), CNN, and Deep Belief Network (DBN) are other techniques that perform well, particularly for multilingual languages.

BERT is a cutting-edge machine learning model utilised in tasks involving sentiment analysis or Natural Language Processing (NLP). Which developed by Jacob Devlin and other people. Additionally, 800 million words from a book corpus and 2,500 million words from the English Wikipedia were used to train the BERT model. According to [17], the fundamental BERT model has a neural network design with 12 layers, 110 million parameters, 768 hidden nodes, and 12 heads. While the huge BERT model differs slightly in that it has 24 layers, 340 million parameters, 1024 hidden nodes, and 16 heads in its neural network design. Based on [19], BERT can use for different types of models, such as Fine-tuning BERT model, Ensemble BERT model, RoBERTa, ALBERT and others. Next, it was stated that a Bidirectional Encoder Representations from Transformers (BERT) model, which comprised cognitive-based attention models, word embedding models that were sentiment-specific, and common sense knowledge, may perform better than others. Other than, BERT model, Natural Learning Processing (NLP) can be one of the most effective methods for sentiment analysis, by having a high accuracy of predictions.

2.4 Previous Studies

According to previous study that show in Table 2-4-1, the techniques used mostly either machine learning or lexicon-based, and less on hybrid techniques. Sentiment analysis can be applied in many areas, such as movie, book, hostel, hotel, e-commerce, social media, and others. Hence to perform sentiment analysis for recommender system, there are two most important data which are review or comments, and rating or star. For that, different areas collect review and star vie different method; however, their intrinsic are the same. Alternatively, sentiment analysis can provide analysed results which is to present the review either negative, positive, or neutral.

No	**Paper**	**Title**	**Model**	**Context**
1	[10]	Online Product Reviews and Their Impact on Third Party Sellers Using Natural Language Processing	Bag-of-words model	Online Product Reviews
2	[5]	Users' Sentiment Analysis of Shopping Websites Based on Online Reviews	Machine learning	Online Review

3	[12]	Sentiment Analysis for E-Commerce Product Reviews in Chinese Based on Sentiment Lexicon and Deep Learning	SLCABG model	Twitter
4	[13]	Feature based Sentiment Analysis for Product Reviews	SVC model	Online Shopping Review
5	[20]	Research on Sentiment Analysis Model of Short Text Based on Deep Learning	Machine learning	Online Shopping Review
6	[21]	Sentiment Analysis of Twitter Data	Machine learning	Twitter
7	[22]	Sentiment Analysis for Social Networks Using Machine Learning Techniques	Machine learning	Twitter
8	[23]	Social Media Analysis of User's Responses to terrorism using sentiment analysis and text mining	Lexicon-based	Twitter
9	[24]	A Study of Sentiment Analysis:Concepts, Techniques, and Challenges	Machine learning	Social Media
10	[25]	Social Media Sentiment Analysis on Employment in Malaysia	Lexicon-based	Multiple channel social media

Table 2-4-1 Previous Study on Sentiment Analysis

Chapter 3

System Methodology/Approach

This chapter will introduce the methodology of implementing this project works, and the methodology on the project's model development. The guideline and methodology of this project are referring to agile development and throw-away prototyping methodology. In this methodology, the phases of design, development or implementation, prototyping, testing phases is mainly following the methodology known as Cross-Industry Standard Process for Data Mining (CRISP-DM). CRISP-DM provided a clear guidance and including from system pre-implementation, system implementation, and system post-implementation. This chapter included the system design diagram including use case diagram, class diagram, architecture diagram, and so on.

3.1 Methodology

3.1.1 Methodology for The Project Development

The development methodology used in this project is agile development's characteristics (refer to Figure 3-1-1). This methodology provides different benefits for system development, such as agility, flexibility, face-to-face communication with the supervisor, etc. This project period takes a long time, which can let me discuss with the supervisor and adjust. I also apply rapid application development methodology such as throw-away prototyping (Refer to Figure 3-1-2). Whenever the developer faces an issue, the developer can return to the analysis and design phases to make an adjustment and discuss it with the supervisor. The process will keep implemented until no problems are discovered in the system. Both methodologies have similarities, such as unclear system requirements and clear schedule visibility, where they complement each other to help me develop the plan. This two methodology allow the process of the system development from system pre-implementation to system post-implementation to be more agile, flexible, and efficient.

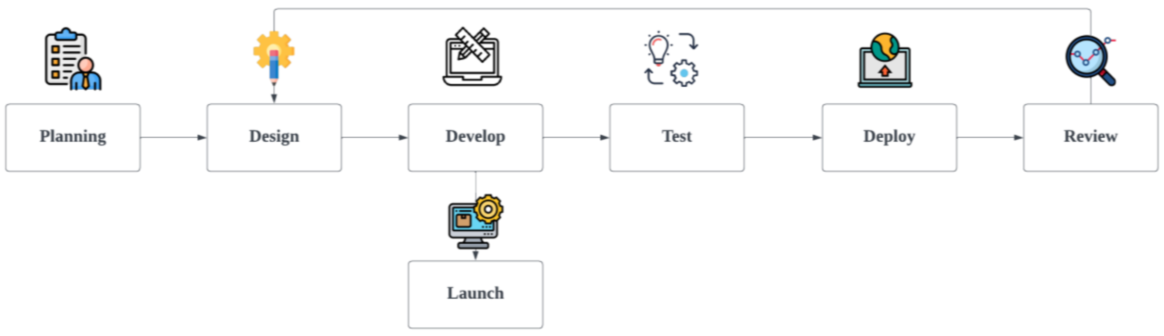


Figure 3-1-1-1 Agile Development Methodology

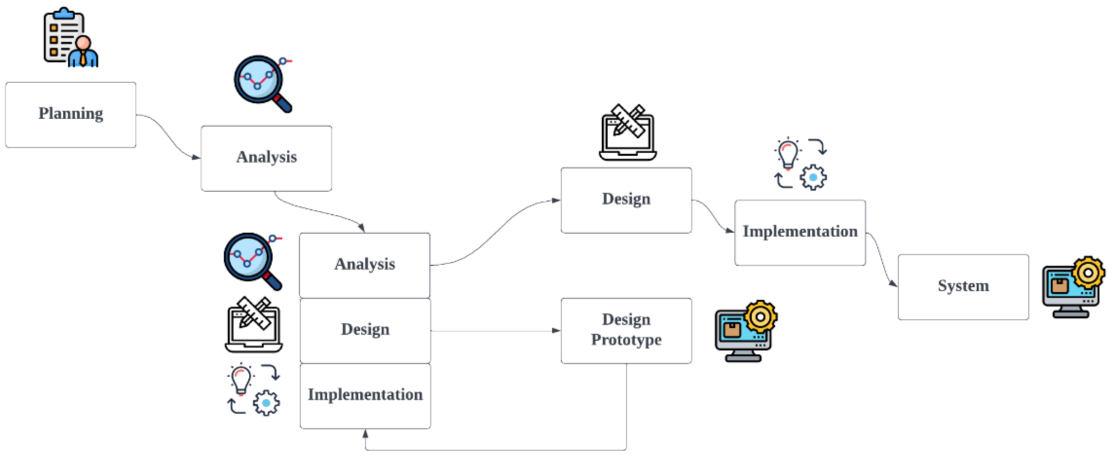


Figure 3-1-1-2 Rapid Application Development: Throw-away Prototyping

3.1.2 Methodology for Sentiment Analysis Model Development

This project consists of two main development parts which are sentiment analysis modeling and mobile application development. For sentiment analysis modeling, there is a methodology play as a process and guideline. Which known as Cross-Industry Standard Process for Data Mining (CRISP-DM). According to [27] and Figure 3-1-2-1, the process such as business understanding, data understanding, data preparation, modeling, evaluation, and deployment. Further system implementation planning of this project according to CRISP-DM will be discussed in Chapter 4.

Business understanding is the first and important phase of the methodology. In this phase, a data mining goal will be determined for the model. The data mining goal suggested to be a data driven way. This is because a goal in data driven way can be interpret by using numbers and diagrams which the presenter can easily communicate and transmit the meaning to the audience. For example, classification is one of the data mining types, and its goal criteria can be determined using calculation such as accuracy, precision, and others. And it can be present in percentages, or diagram such as pie diagram, histogram diagram, and so on [27].

Second step is **data understanding**. This step consists of collecting data from data sources, exploring, and describe the data. Data understanding is an important step that enable user to understand the structure of data in terms of data types, format of data value, and purpose or meaning of the data. It plays a role as guidance for user to implement task easily [27].

Next, according to [27], in **data preparation** phase, the main objective is to ensure the data quality by performing data preprocessing such as data cleaning, attribute selection and other. For example, removing the anomalies data. Such as, in a total number 1,000,000 data, there are almost fall in RM100 to RM 500. And there are about 100 data falls in above RM3,000 which can consider as anomalies that will affect the model’s performance. And prepare different data for modelling such as training data, validation data, and test data. Hence the model will be performing well.

Moving on, during **modelling** phase, the user need to select the modelling technique depend on the business problem or goal, and the data. In modelling, the user needs to set a specific parameter to assess the model either the model is appropriate on model evaluation and its evaluation criteria or requirement. Lastly, choosing the best model trained. For example, the model technique that is suitable for price prediction such as linear regression, logistic regression, decision tree and others [27].

Evaluation is an important step which enable to monitor the performance of the model trained. According to [27], it is to determine the model achieve its goals. The result can be interpreted in numbers and diagram to ease the audience understand them. The evaluation phase shall determine the next action. For example, if the model achieves its goal, the user can proceed to deployment phase, else the user may need to reset a parameters and other requirements of the models to ensure the model perform in a high-quality way.

Lastly, **deployment** phase. According to [27], in this phase, the user can generate a report or guidance. Which describe in the details about the future maintenance action, the way of monitoring, and planning deployment for the software. The user shall state the infrastructure, software, or any resources that needed for the model usage. As well as it shall be providing the integration methods that suitable for the project such as software, application, website and so on.

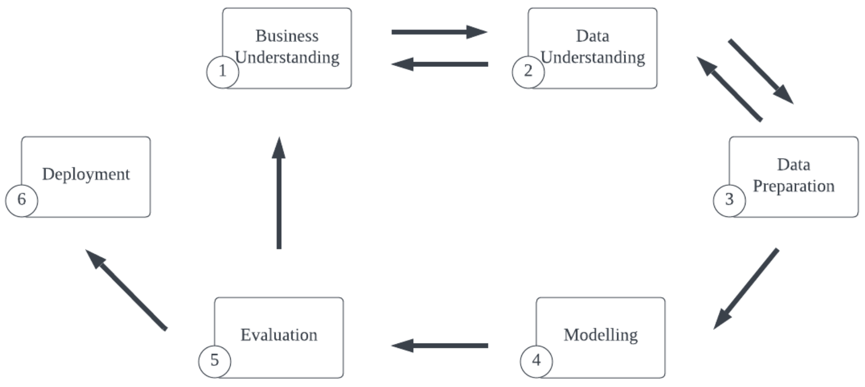


Figure 3-1-2-1 Cross-Industry Standard Process for Data Mining (CRISP-DM)

3.2 System Design Diagram

3.2.1 System Architecture Diagram

According to Figure 3-2-1-1, the project will consist of a user, application interface, database storage, and database server. Firstly, there are mainly two users: the normal user, the customer, and the shop owners. Where, normal user or customer use the application for search shop or store that near by them. Shop owner user this application for marketing and branding purposes that attract customers. Hence, there shall be different interface between customer and shop owner. However, it will not cover in this project. There will be an interface for user to key in rating and review by selecting category of shop or store and selecting a shop or store. Other than giving review and rating, customers can also use this application to get information about shop or store before they take action to consume any item or product. Next, this application shall also allow user to select different criteria for filtering or ranking to achieve recommendation purposes. Next, all data shall store in either Firebase, Supabase, or CSV file. To achieve the project's purpose, Python programming language is used for the application back-end, including sentiment analysis.

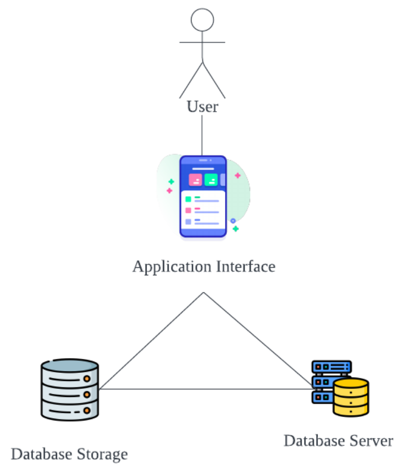


Figure 3-2-1-1 System Architecture Diagram of Shopping Recommender System for University Students

3.2.2 Use Case Diagram

According to Figure 3-2-2-1, the shopping recommender system for University Students have three main actors who are normal user, shop owner, and admin. Each of the actor can perform different or similar activity. For example, in this system, both normal user and shop owner have to register themselves to use the application. Main difference is shop owner had to provide a description and information for normal user or known as customer to refer. Description such as, location, what type of shop or store, what and how many types of products is selling in the shop, and other information. And normal user will provide review and rating to the shop and by selecting different criteria such as sentiment analysis result, price and other to get recommendations of shop by filtering or ranking. Lastly, admin is the sentiment analysis or known as the back-end system that support and run the application. It collects the data for analysis purposes and utilize the analyzed data for application interface. This user case diagram consist of complete elements; however in this project it will not focus on normal user and shop owner login, but the simple application interface to get user input and display the shop recommendation after selecting criteria for filtering or ranking, and sentiment analysis that support the application interface.

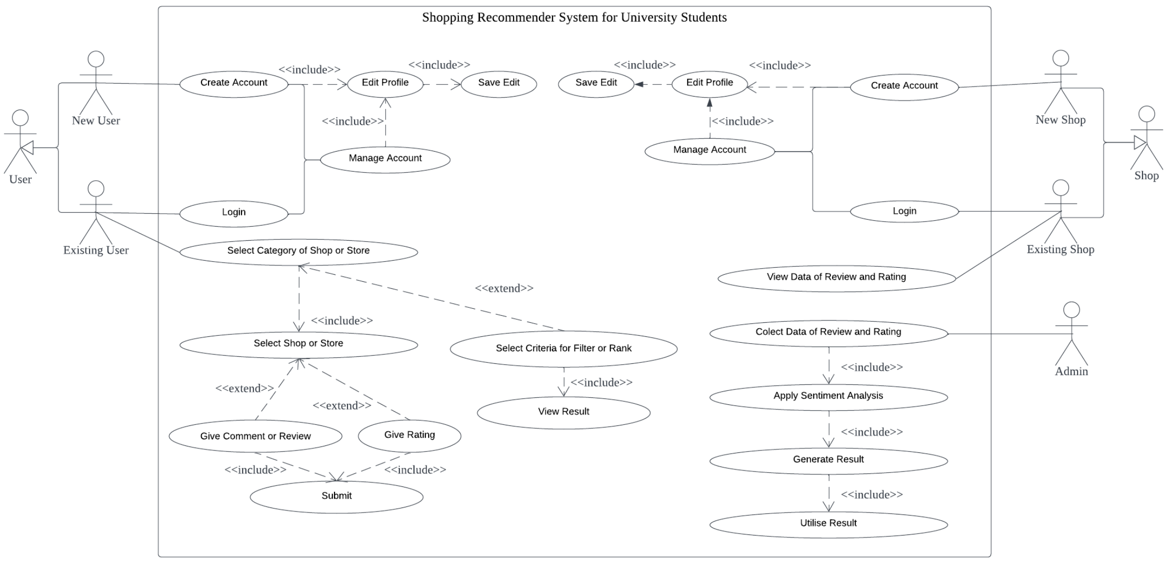


Figure 3-2-2-1 Use Case Diagram of Shopping Recommender System for University Students

3.2.3 Activity Diagram

Activity 1: Normal User or Shop Owner Login

Figure 3-2-3-1 show a process flow of a normal user or known as a customer and shop owner to login or create account for themselves in the applications. There will be two different interfaces for normal user and shop owner. However, this project will not cover user account creation and login feature in the application.

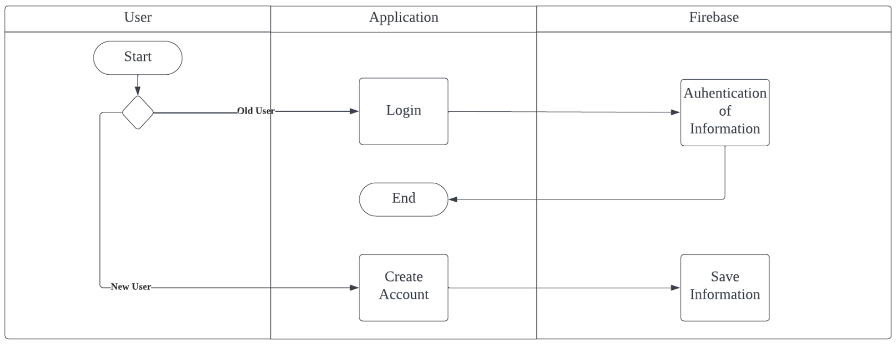


Figure 3-2-3-1 Activity: Normal User or Shop Owner Login

Activity 2: Normal User Provide Review or/and Rating to A Shop or Store

Figure 3-2-3-2, show a process flow of a customer to perform action of giving review and rating to a shop or store. Firstly, user can choose the categories of shop or store. Then, user shall select a store or shop. Next, user can give review and rating to the shop or store, then submit. While submit the review and rating, the sentiment analysis is performed and saved.

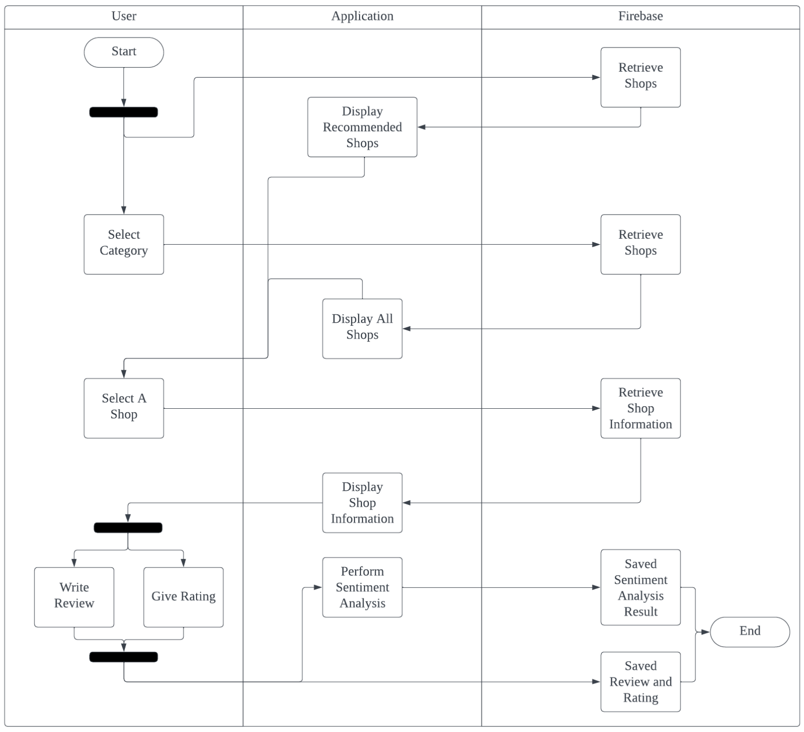


Figure 3-2-3-2 Activity: Normal User Provide Review or/and Rating to A Shop or Store

Activity 3: Normal User Choose Criteria for Shop or Store Recommendations

In this Figure 3-2-3-3. A user can choose one or more criteria for a filtering or ranking, lastly, recommend a list of shop or store for user. Criteria such as average rating and 5 level in range from percentage of a shop getting positive review will be used in filtering and ranking. Lastly, displaying the shop fulfilling the requirement and criteria set.

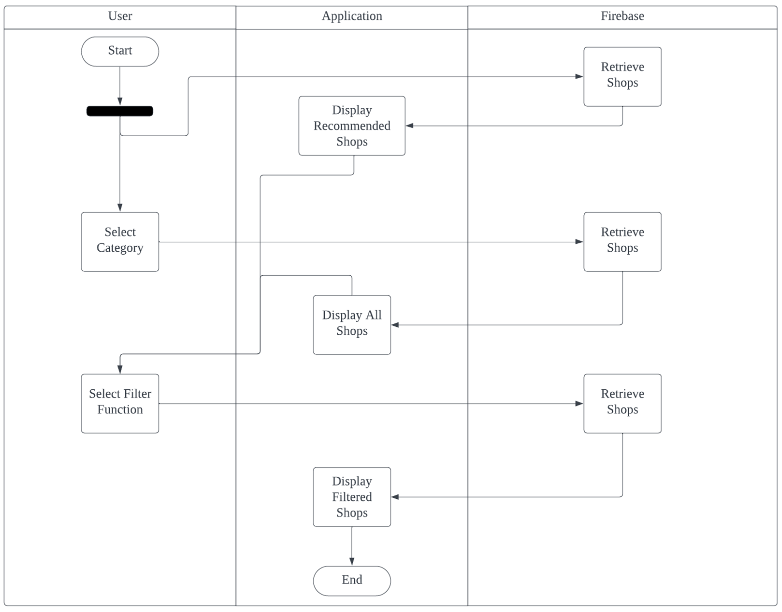


Figure 3-2-3-3 Activity: Normal User Choose Criteria for Shop or Store Recommendations

Activity 4: Shop Owner View Data

In this Figure 3-2-3-4, the shop owner can view the data given by customer such as review and rating. Hence, shop owner can perform changes to their shop environment, customer services, and other. Lastly, able to build the brand and attract more customers. However, this feature is not cover in this project.

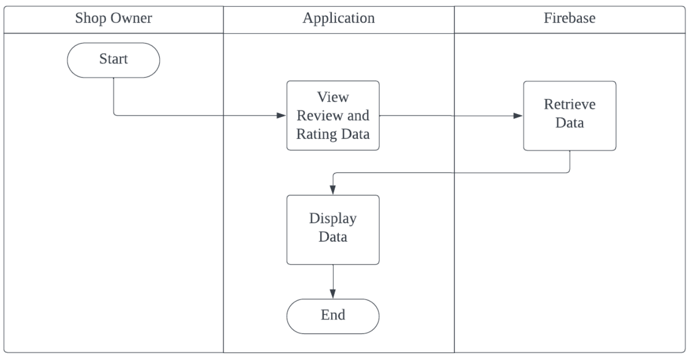


Figure 3-2-3-4 Activity: Shop Owner View Data

Activity 5: Admin

According to Figure 3-2-3-5, there is an admin exist in the recommender system for university student. It act as a back-end system which is the sentiment analysis that used the data when it is submit to save in database, it will perform the analysis. Lastly, updating the current content displayed to the application interface. Hence, increase the customer satisfaction on recommending customer the shop or store.

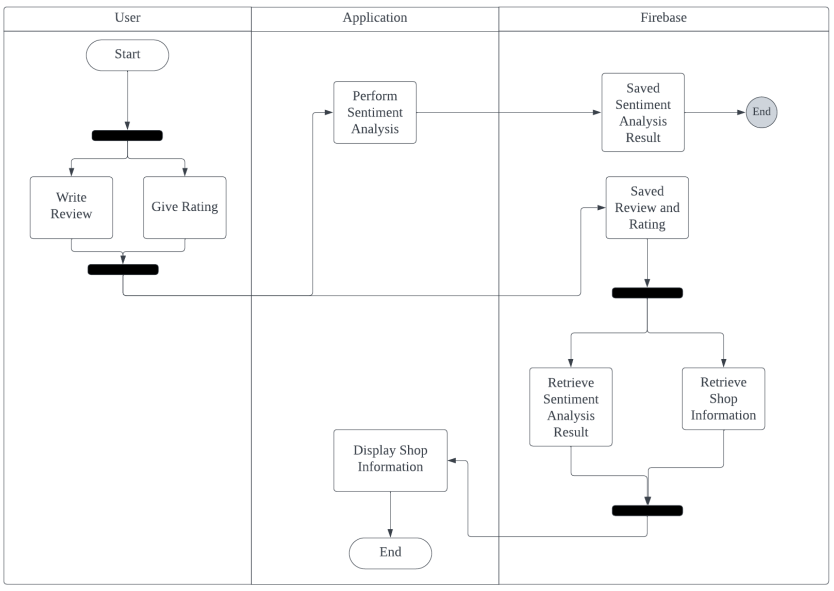

User
Application
Firebase
Start
Write Review
Give Rating
Perform Sentiment Analysis
Saved Sentiment Analysis Result
End
Saved Review and Rating
Retrieve Sentiment Analysis Result
Retrieve Shop Information
Display Shop Information
End

Figure 3-2-3-5 Activity: Admin

3.2.4 Class Diagram

Based on Figure 3-2-4-1, there will be mainly 4 classes consist of unique attributes and relationship between each class. Firstly, Shop Information class that contain a list of attributes such as shop_id, shop_name, shop_location, shop_phone, shop_email, and shop_desc. In this project, the information of shop is predefined where it will not include a application's function for shop owner to register themselves for using this application. Next Customer Review class consist of main two data input from user which is rating and review. To perform sentiment analysis and utilizing the analyzed data for application interface design. There is a class named Sentiment Analysis Result, to store the result. Hence, increasing the satisfaction of customer where it is user friendly to provide enough information as a recommender system. Lastly, User class which contain user information; however, this project is not covering an application function for user entering and creating their own account.

The relationship between User class and Customer Review class is aggregation. As customer's review only exist if a user providing. Next, Customer Review class and Shop Information class indicate a composition relationship. Where a shop can exist even there is no customer review and rating. Lastly, relationship of Customer Review class and Sentiment Analysis Result class is aggregation. Due to whenever a rating and review is given, sentiment analysis will be performed and saved the result.

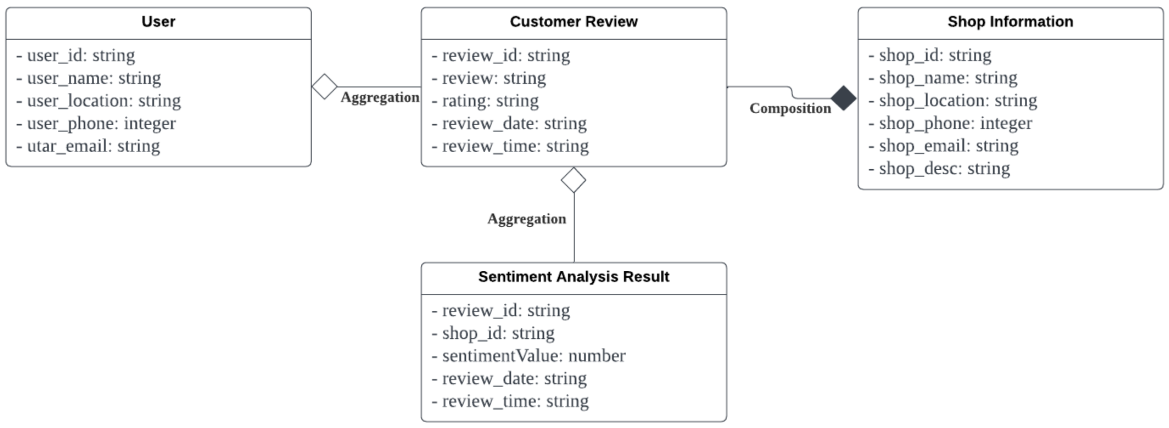
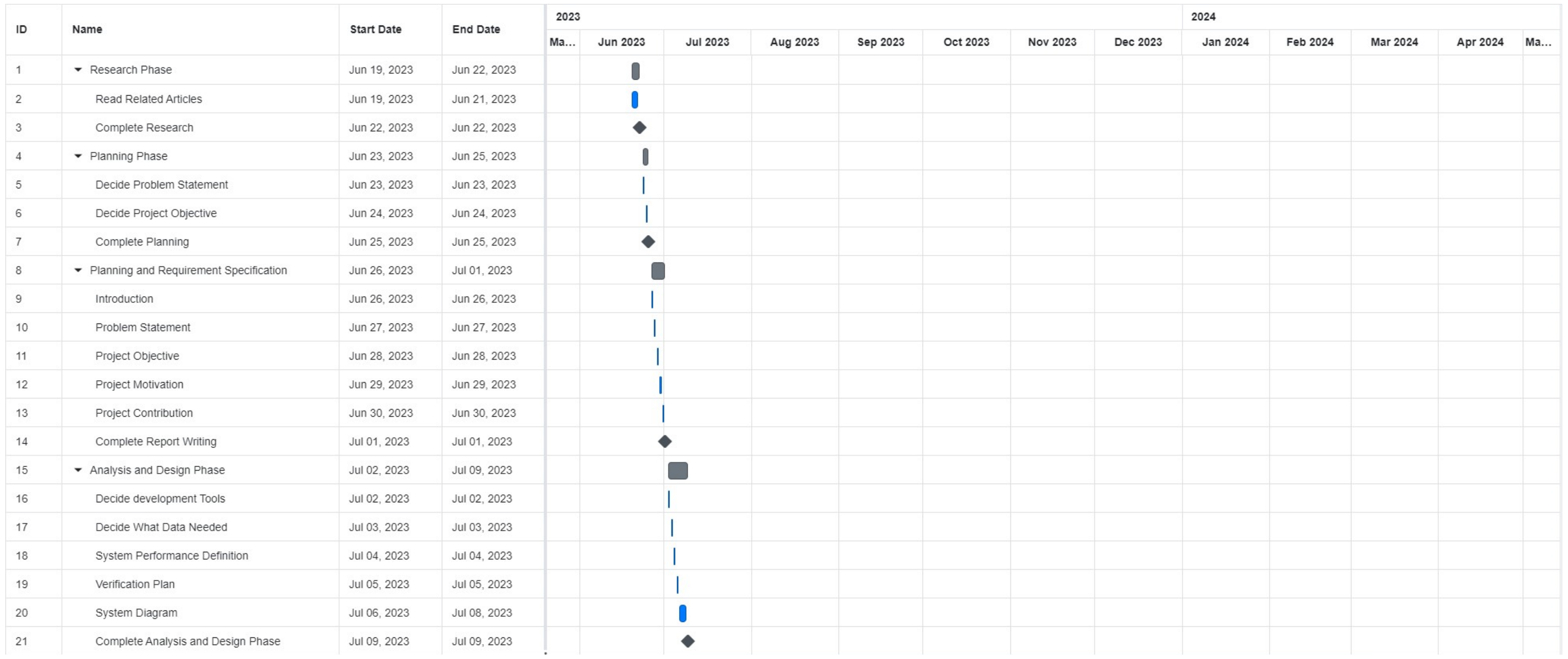


Figure 3-2-4-1 Class Diagram of Shopping Recommender System for University Students

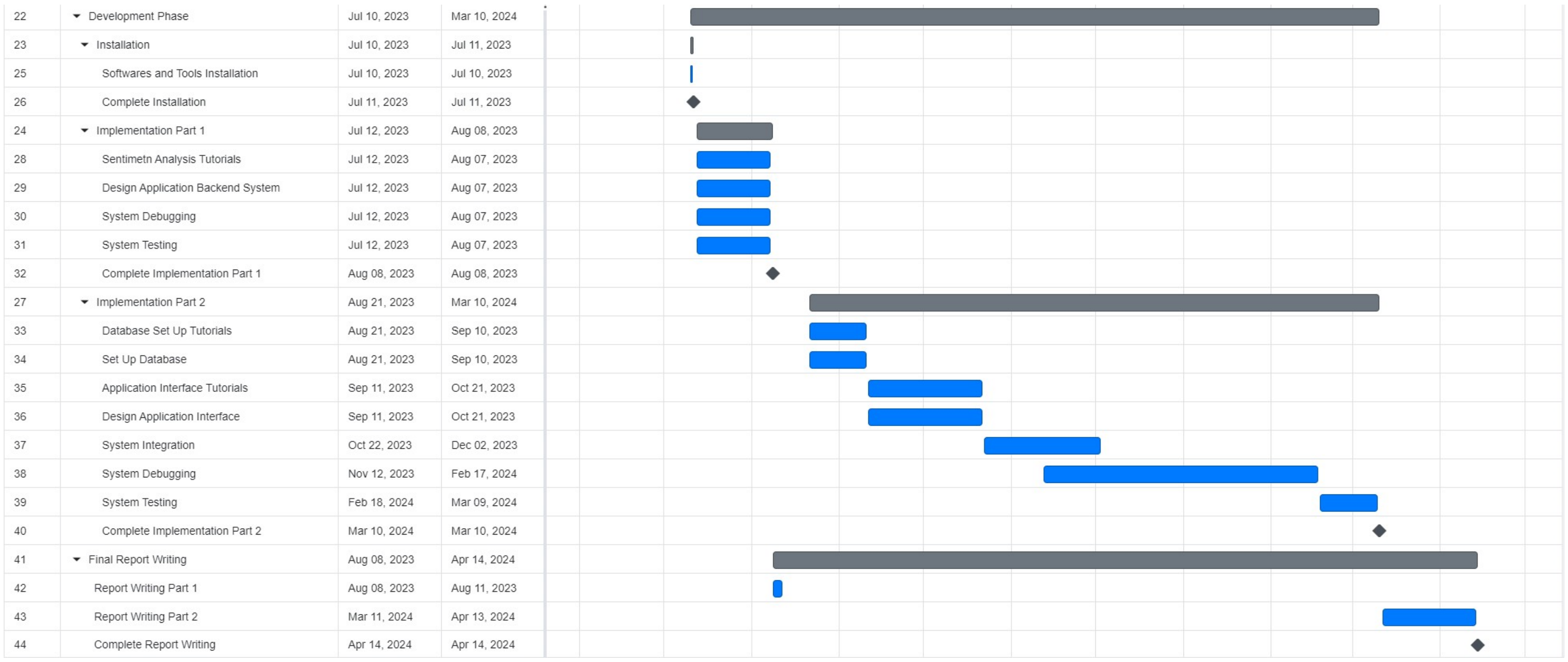
3.3 Timeline

Figure 3-3-1 and Figure 3-3-2 is the timeline of the project. The development process consists of 5 phases which are: research phase, planning phase, planning and requirement specifications phase, analysis and design phase, development phase, and final report writing. The estimation of project's development duration is about 10 months, starting from Jun of 2023 until April of 2024. According to the project development methodology, it allows the developer to perform backward if necessary, during the development period. As conclusion, it able to give developer flexibility to develop the recommender system application.



ID	Name	Start Date	End Date
1	Research Phase	Jun 19, 2023	Jun 22, 2023
2	Read Related Articles	Jun 19, 2023	Jun 21, 2023
3	Complete Research	Jun 22, 2023	Jun 22, 2023
4	Planning Phase	Jun 23, 2023	Jun 25, 2023
5	Decide Problem Statement	Jun 23, 2023	Jun 23, 2023
6	Decide Project Objective	Jun 24, 2023	Jun 24, 2023
7	Complete Planning	Jun 25, 2023	Jun 25, 2023
8	Planning and Requirement Specification	Jun 26, 2023	Jul 01, 2023
9	Introduction	Jun 26, 2023	Jun 26, 2023
10	Problem Statement	Jun 27, 2023	Jun 27, 2023
11	Project Objective	Jun 28, 2023	Jun 28, 2023
12	Project Motivation	Jun 29, 2023	Jun 29, 2023
13	Project Contribution	Jun 30, 2023	Jun 30, 2023
14	Complete Report Writing	Jul 01, 2023	Jul 01, 2023
15	Analysis and Design Phase	Jul 02, 2023	Jul 09, 2023
16	Decide development Tools	Jul 02, 2023	Jul 02, 2023
17	Decide What Data Needed	Jul 03, 2023	Jul 03, 2023
18	System Performance Definition	Jul 04, 2023	Jul 04, 2023
19	Verification Plan	Jul 05, 2023	Jul 05, 2023
20	System Diagram	Jul 06, 2023	Jul 08, 2023
21	Complete Analysis and Design Phase	Jul 09, 2023	Jul 09, 2023

Figure 3-3-1 First Part of Timeline of Shopping Recommender System for University Students Project



22	Development Phase	Jul 10, 2023	Mar 10, 2024
23	Installation	Jul 10, 2023	Jul 11, 2023
25	Softwares and Tools Installation	Jul 10, 2023	Jul 10, 2023
26	Complete Installation	Jul 11, 2023	Jul 11, 2023
24	Implementation Part 1	Jul 12, 2023	Aug 08, 2023
28	Sentimetn Analysis Tutorials	Jul 12, 2023	Aug 07, 2023
29	Design Application Backend System	Jul 12, 2023	Aug 07, 2023
30	System Debugging	Jul 12, 2023	Aug 07, 2023
31	System Testing	Jul 12, 2023	Aug 07, 2023
32	Complete Implementation Part 1	Aug 08, 2023	Aug 08, 2023
27	Implementation Part 2	Aug 21, 2023	Mar 10, 2024
33	Database Set Up Tutorials	Aug 21, 2023	Sep 10, 2023
34	Set Up Database	Aug 21, 2023	Sep 10, 2023
35	Application Interface Tutorials	Sep 11, 2023	Oct 21, 2023
36	Design Application Interface	Sep 11, 2023	Oct 21, 2023
37	System Integration	Oct 22, 2023	Dec 02, 2023
38	System Debugging	Nov 12, 2023	Feb 17, 2024
39	System Testing	Feb 18, 2024	Mar 09, 2024
40	Complete Implementation Part 2	Mar 10, 2024	Mar 10, 2024
41	Final Report Writing	Aug 08, 2023	Apr 14, 2024
42	Report Writing Part 1	Aug 08, 2023	Aug 11, 2023
43	Report Writing Part 2	Mar 11, 2024	Apr 13, 2024
44	Complete Report Writing	Apr 14, 2024	Apr 14, 2024

Figure 3-3-2 Second Part of Timeline of Shopping Recommender System for University Students Project

Chapter 4

System Design

This chapter will introduce the system block diagram and flow diagram for the model and application development for system implementation planning according to the components involved. These components used for system development which are model and application development. These components are Database Firebase, Python IDLE for modelling based on CRISP-DM, and Android Studio for application development. It also part of the deployment process under CRISP-DM. This chapter included the system component information such as specification, and interaction operation. Hence, this chapter shows the system implementation planning for the Firebase, model, and application development.

4.1 System Block Diagram

In Figure 4-1-1, this application consists of front-end and back-end. Firstly, front-end of the mobile application. It consists of few interfaces including made page for displaying the category choices and top recommended shops, shop list displaying page, shop display page, shop review displaying page, and filter page. In the shop display page, there are two function is provided to user including function of giving review and rating and view the past reviews and rating. In shop list displaying page, a filter function provided to user for selecting criteria for filtering and ranking to recommend shop or store to user. Lastly, back-end of the mobile application shall perform sentiment analysis by integrading and loading the model trained to generate result by using the initial data that stored in Firebase and store the result of sentiment analysis.

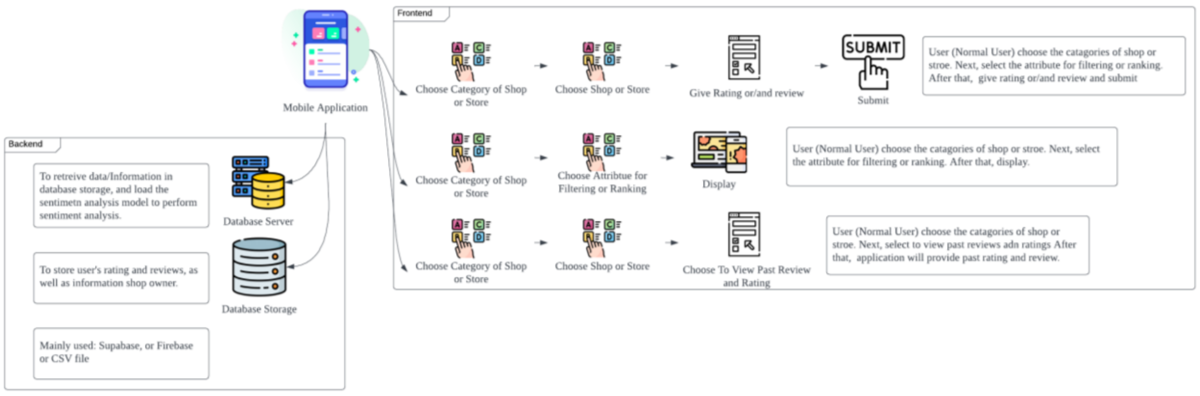

Mobile Application
Frontend
Choose Category of Shop or Store
Choose Shop or Store
Give Rating or/and review
SUBMIT
Submit
User (Normal User) choose the catagories of shop or stroe. Next, select the attribute for filtering or ranking. After that, give rating or/and review and submit
Choose Category of Shop or Store
Choose Attribtue for Filtering or Ranking
Display
User (Normal User) choose the catagories of shop or stroe. Next, select the attribute for filtering or ranking. After that, display.
Choose Category of Shop or Store
Choose Shop or Store
Choose To View Past Review and Rating
User (Normal User) choose the catagories of shop or stroe. Next, select to view past reviews adn ratings After that, application will provide past rating and review.
Backend
To retreive data/Information in database storage, and load the sentimetn analysis model to perform sentiment analysis.
Database Server
To store user's rating and reviews, as well as information shop owner.
Database Storage
Mainly used: Supabase, or Firebase or CSV file

Figure 4-1-1 Block Diagram of Shopping Recommender Application

In Figure 4-1-2, it shows the process of the model development. This process mainly based on the methodology known as CRISP-DM. The model needs to be trained and integrated int the application developed by using Android Studio. Hence, it able to perform analysis when having review and rating submit by the user. Firstly, to train a model, data understanding is important as a clear understanding on the nature and structure of the attribute that dataset contain will help and ease the process of next phases. Next, the data need to be prepared well before modelling. Which data quality ensuring by checking and cleaning, data restructuring and tokenization for the review. And prepare set of data for different usage including model training, model validating, and momdel testing. Next, modelling can be started once data is well prepared. During modelling, it is expected to show the accuracy and loss for futher study and evaluation purposes. Before the model integrate into application, model testing is needed to evaluate the model's performance with the unseen data. To ensure the model performing well in learning instead of memorizing. After a model is trained, the model can be integrated into the application developed to perform sentiment analysis when review and rating is given by user. And it will generate the result and save into the Firebase Firestore for future query purposes to perform recommendation based on filtering and ranking criteria that user set.

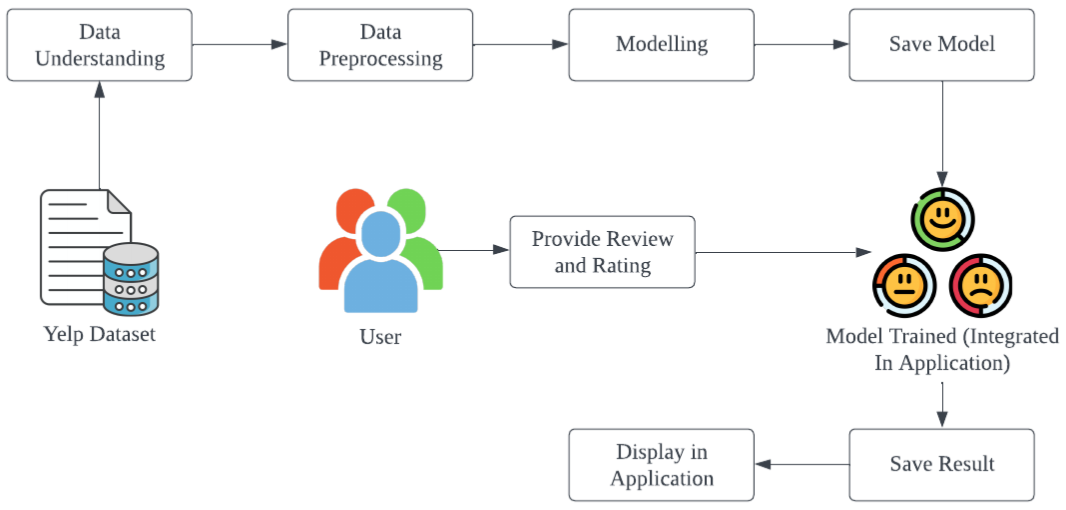


Figure 4-1-2 Block Diagram of Sentiment Analysis Model

4.2 System Flow Diagram

Figure 4-2-1 present the system flow diagram of the application. In this system, there are mainly four main role which are user, application, firebase, and sentiment analysis model. In this project, the system provided a basic function of displaying shop with recommendations. The recommendation is performed by using the sentiment analysis model. At first, the application will display 4 categories of shop for user to choose, with a number of shops recommended with its requirements. Where the requirement based on the average rating and sentiment analysis result. Next, the user either choose the category to view the list of shop or direct view the shop information by clicking on the shop. If the user chooses a category of the shop, a list of shop will be displayed with detail ordering based on the average rating. Moving on, the user can choose a filtering criterion based on the average rating and sentiment analysis result to display the list of shop that fulfill the criteria. And, once user would like to check information of a shop, the user can direct click on the screen and the screen will display the information of the shop. The user can perform either to view all past review or give their own review and rating. If the user prefers to the former, the screen will display the review according to the data and time. For the later, after user wrote review and rate the shop, the result will save into firebase and simultaneously perform sentiment analysis and saved result into firebase.

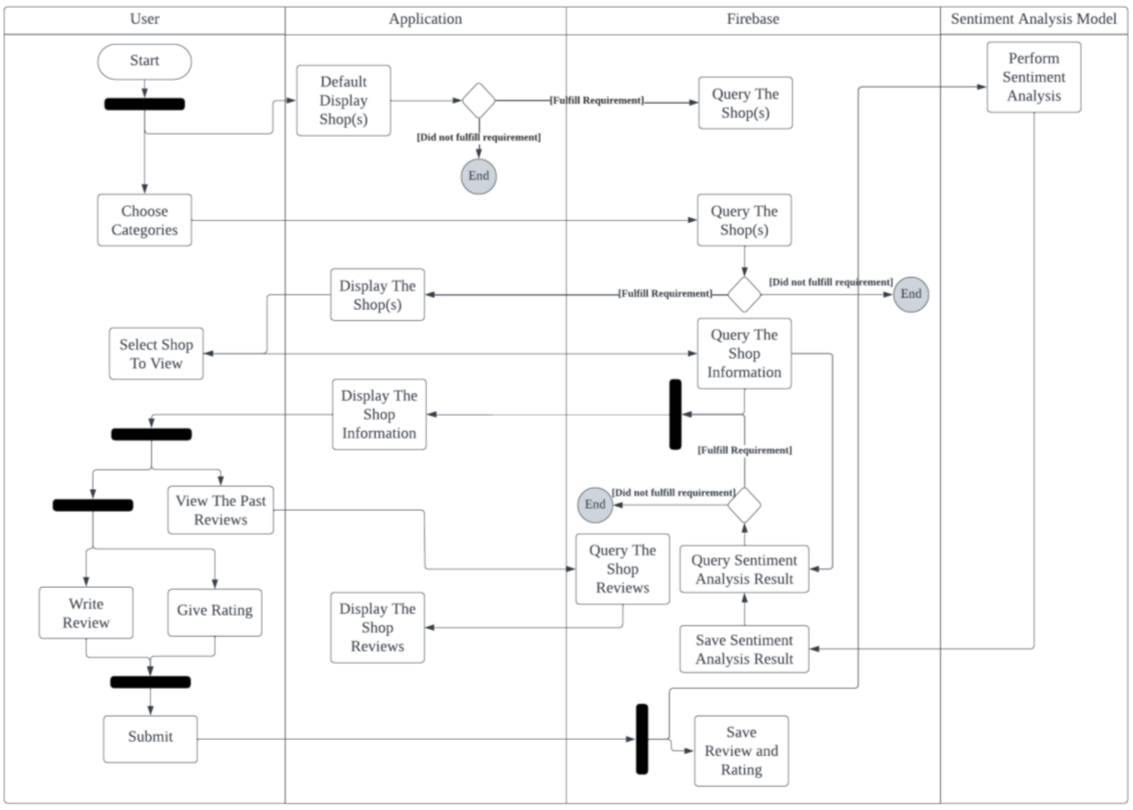


Figure 4-2-1 System Flow Diagram of Shopping Recommender Application

For model development process according to the Figure 4-2-2, the dataset that model will use is epxteced in CSV file format. After having a dataset for model development, data understanding will perform by using four function to get the information, shape, attributes, and example value stored in the dataset. This process is important, for example, the data type of the attribute and signature input that set for the model need to match. Else, the model may perform enexpectally low quality and cause error. After this process, the data needed to be prepared well before modelling start. Firstly, setting the dataset size to split the dataset into three different dataset for different purporse such as training dataset, validating dataset, and testing dataset. These datasets shall not be repetitive or duplicated. After dataset is created, data preprocessing needs to be performed. In the sentiment analysis model, it takes mainly two attribute which are reviews and ratings. And there is a data restructuring need for the rating. And tokenization needs to be performed for the review by using the tokenizer from BERT. After the data is prepared well. The modelling can be start with paremeter set. For example, number of the iteration, batch size, max length of token, and so on. During modelling, the accuracy and loss of training and validating is expected to calculate for evaluation. After the model completed the training, it needs to test with using testing dataset or known as unseen data to evaluate its performance. Eventually, the model can be saved and resources saving for further peployment purposes such as integrating into application developed.

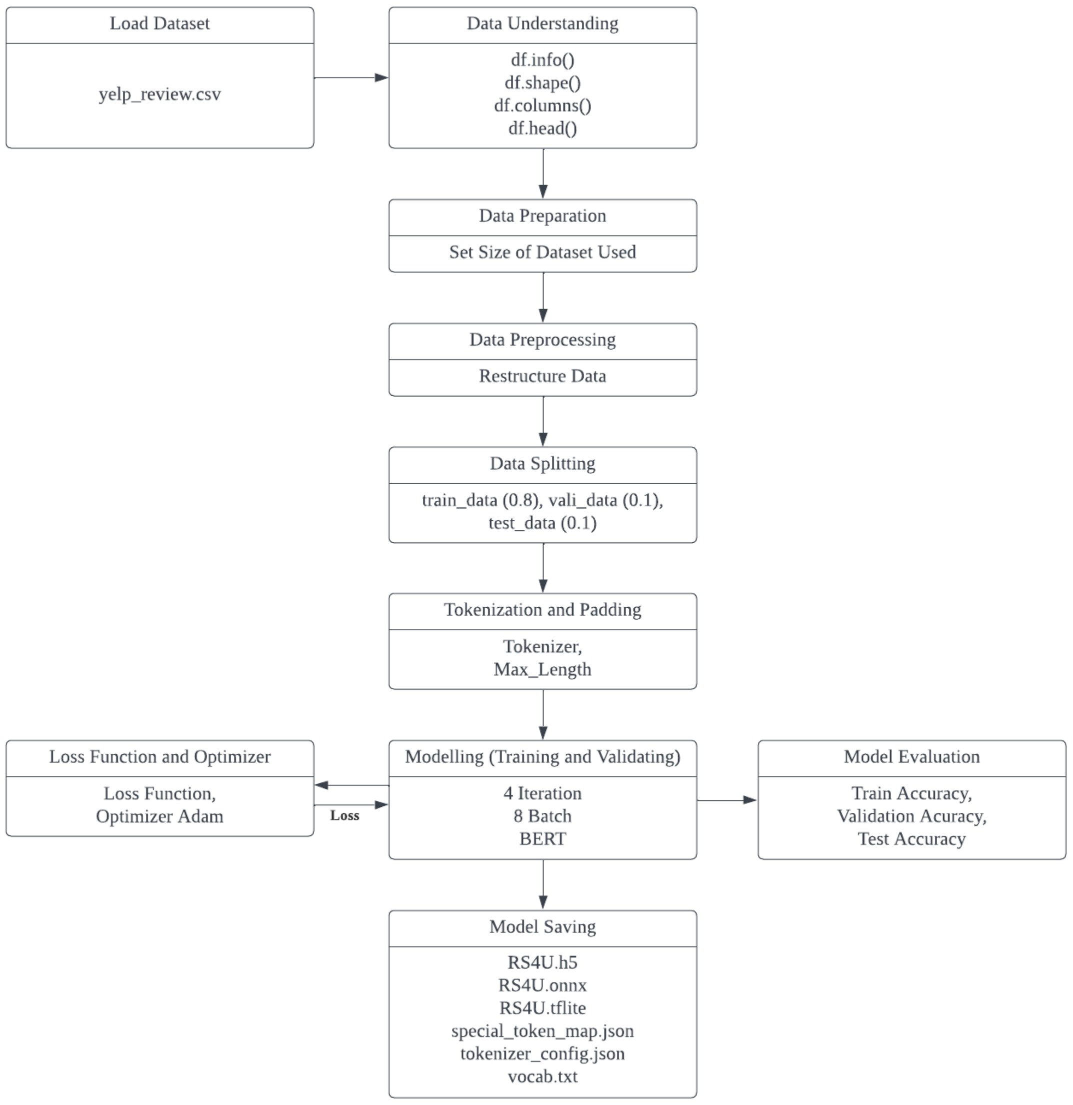


Figure 4-2-2 System Flow Diagram of Sentiment Analysis Model

4.3 Draft User Interface of Application

In this project, there will be a mobile application with a simple interface which enable the user to utilize the analyzed data, and initial data. In this subtopic, will be introduced the draft interface of the application. Figure 4-3-1 show a main page of application’s interface. In this page, the user can choose the shop or store categories. This project will only provide 4 shop categories which are healthcare, mini market, stationary, and technology.

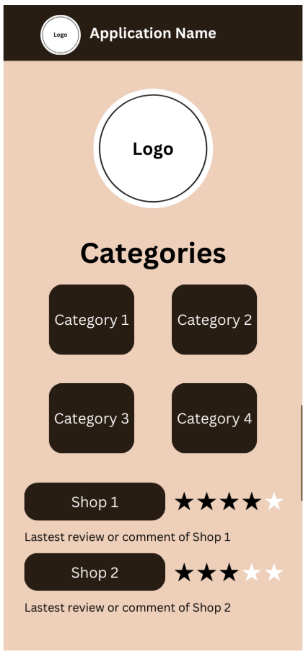


Figure 4-3-1 Draft Design: Main Page of Application

In Figure 4-3-2, it shows the list of shop or store. With showing the average of rating and the latest comment of user or the total number of comments. The user able to click the shop or store to view more details and information. This interface includes a function for user to perform filtering and ranking by choosing the criteria.

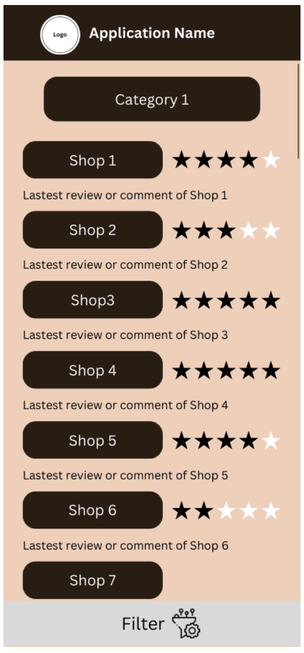
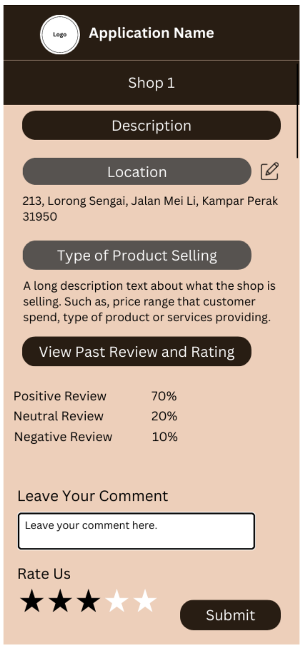

Logo
Application Name
Category 1
Shop 1
Lastest review or comment of Shop 1
Shop 2
Lastest review or comment of Shop 2
Shop3
Lastest review or comment of Shop 3
Shop 4
Lastest review or comment of Shop 4
Shop 5
Lastest review or comment of Shop 5
Shop 6
Lastest review or comment of Shop 6
Shop 7
Filter

Figure 4-3-2 Draft Design: List of Shop on Interface

Figure 4-3-3 show the information and detail of the shop or store. Information included such as location, and a short description of the shop or store related to product selling and others. In this interface, there are two functions that user can use which are view past review and rating and giving comment and rating to the shop or store.



213, Lorong Sengai, Jalan Mei Li, Kampar Perak 31950

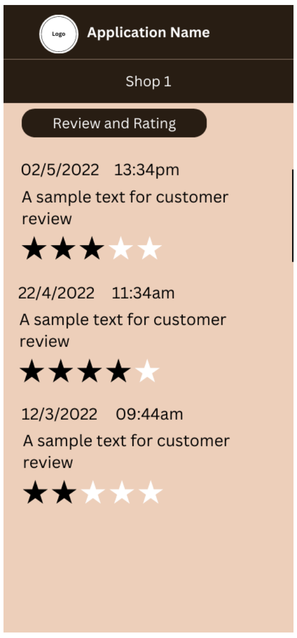
A long description text about what the shop is selling. Such as, price range that customer spend, type of product or services providing.

Positive Review	70%
Neutral Review	20%
Negative Review	10%

Leave your comment here.

Figure 4-3-3 Draft Design: Shop Details Interface with Commenting and Rating Function

Figure 4-3-4 is an interface that show list of past review and rating. The displaying logic will be set to based on the ordering of date and time.



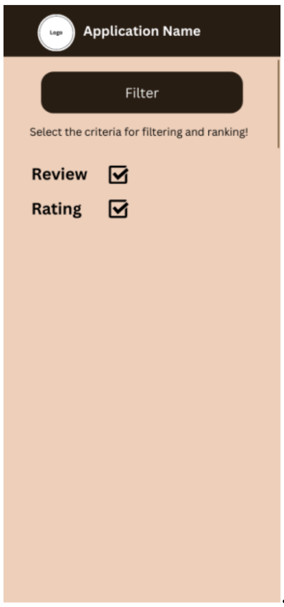
02/5/2022 13:34pm
A sample text for customer review

22/4/2022 11:34am
A sample text for customer review

12/3/2022 09:44am
A sample text for customer review

Figure 4-3-4 Draft Design: List of Past Review and Rating

In this interface (Figure 4-3-5), it is an interface that lets user to choose the criteria for ranking and filtering. Users can choose either one or both of the rating and review criteria. In this draft design, the page is expected to provided two criteria for user to choose. Firstly, review represent the percentage of the shops getting a positive review. Based on this, the page will display the highest to lowest percentage. Next, the rating indicate that the average rating of the shop.



.

Figure 4-3-5 Draft Design: Interface of Selecting Criteria for Filtering and Ranking

4.4 System Components Specifications

In this project, it consists of three main components. Which is Firebase for storing and retrieving data, Android Studio for mobile application development, and lastly, Python IDLE for sentiment analysis model development. According to [27], Firebase is used for many purposes. It is designed with the functionalities of cloud firestore, could function, authentication, real-time database and so on. Firebase can be used in all parties including IOS applications, Android application, and web applications. Hence, the project will use one of the functions of Firebase which is Firestore database for saving and querying data. And it is using NoSQL database function.

Android Studio is a platform for developer to develop applications. It contains three main component which are front-end and back-end. Front-end refer to the Extensible Markup Language (XML), it is to design the user interface by simple coding. For back-end, Android

Studio support two programming language which is Java, and Kolin. Among these two components there is a resources file to contain of non-coding resources such as bitmap or images, and others [28].

[29] mentioned that every Python installation, it includes one component which known as Integrated Development and Learning Environment (IDLE). It enables developers to perform coding with importing different libraries that needed such as TenforFlow, Keras, PyTorch, NumPy, and others. And the output of coding will be shown in a component that downloaded known as shell. For that, in this project, the sentiment analysis model development will be using Python IDLE and Python Shell. Hence, these three components interact between each and other to perform as a whole system. Therefore, the application will be performing the main function and user interface. Where, the user input that saved via application will be stored into the Firebase's firestore database, and the sentiment value that generated by using the model saved in the application will be stored into the firestore database as well. Lastly, to apply the sentiment analysis model in the application. The model will be store into a TensorFlow lite that ease the process of model integration in applications.

4.5 System Implementation Planning

4.5.1 Firebase: Database

In this project, Firebase consist of three tables which named as "ShopInformation", "CusReview", and "SAResult" in Firestore Database. Firstly, "ShopInformation" contains seven attributes with string data type. Table 4-5-1-1, shown the attribute name and data types with its description. However, the shop information of this project is key in manually by developer. Which this project did not consist of function or feature for shop owner to register their shop. The attributes that planned to create which include shop ID, shop name, shop category, shop location, shop phone number, shop email address, and description of shop. In this table, shop ID will be set as primary key for further query activities. Next, this project planned to contain a minimum 10 shops of each four categories which stated in Table 4-4-1-2.

Attribute	**Data Type**	**Example Value**	**Description**
shop_id	String	S002	Primary key in the table that is unique and not repeatable. It eases the application to

			retrieve its data and display and perform different calculation function. The value will be starting with S capital letter and with series of unique number.
shop_name	String	Woh Peng Kee Mini Market	Name of the shop or store.
shop_cat	String	Mini Market	It represents the shop categories which are mini market, technology, stationary, and healthcare.
shop_location	String	2243, 2244, Jalan Batu Sinar, Taman Bandar Baru. 31900, Kampar, Perak.	Location of the shop or store.
shop_phone	String	016-281 2847	Office number or mobile phone number of the shop or store.
shop_desc	String	Selling different item and food.	Description or basic information of the shop such as the operating hours, services provided, and others.
shop_email	String	whopengkee@gmail. com	Email of the shop or store.

Table 4-5-1-1 Shop Information Table

Mini Market	**Healthcare**	**Technology**	**Stationary**
Mesra Mini Market	HTM Pharmacy Kampar	Send Computer Services	Ke Yi Enterprise
J.K.M Mini Market	Yen Who Yong Medical Hall	Cell Tech Enterprise	Uni Culture
Who Peng Kee Mini Market	Hai-O Raya Bhd	Kedai Komputer Lenovo	Yew Yew Stationary
Loy Fatt Mini Market	BIG Pharmacy Kampar	Play Gadget Store Kampar	Xin Min Zhong Enterprise
Amuthan Mini Market	Kedai Ubat Weng Onn	Acai Tech	Chee Yuk Sports & Stationary
One Stop	Sunday Multicare Parmacy Kampar	Mr Gadget Kampar	Cherry Hourse

Mix Kampar	Kean Fatt Medical Shop	SK Super Tech	HP Bookstore Kampar
Yew Fatt Mini Market	Tian Tian Herbs	Mango Technology Enterprise	XMZ Printing
OK Mart Pasar Mini	Kampar Medical Supplies Sdn Bhd	Tech Team Computer & Software Solution Kampar	DE Stationary
Y&Y Mini Market	Siang Pharmacy Kampar	Cynetech Kampar	WF Lam Sports & Stationary

Table 4-5-1-2 List of Shop

Next, another table is "CusReview" contains six attributes with string data type. Table 4-5-1-3, shown the attribute name and data types, which the user's review and rating that submitted by the user in application and being stored in Firebase. In this table, the planned attributes will be created including review ID, shop ID, review data, review time, review, and rating. Where review ID will play a vital role for representing each review and rating as primary key. As for shop ID, it is a foreign key that linked with the other table, which ease the process of querying activities.

Attribute	**Data Type**	**Example Value**	**Description**
review_id	String	[0, 0, 8, 4]_2024-03-13_09:22:24	Primary key in the table that is unique and not repeatable. It eases the application to retrieve its data and display and perform different calculation function. The value will be starting with an array that store 4 four number that randomly generated with the date and time that review and rating is submitted.
shop_id	String	S002	Foreign key in the CusReview table and primary key for ShopInformation. Which is a constraint that ensure the data is manipulated correctly such as inserted, updated, and deleted.

review_date	String	2024-03-13	The date that review and rating is submitted by the user in the application.
review_time	String	09:22:24	The time that review and rating is submitted by the user in the application.
stars	String	5.0	Rating from 1 to 5 given by the user in the application
texts	String	Price is cheap, and good quality	Review or comment given by the user in the application.

Table 4-5-1-3 Customer Review

Lastly, another table is "SAResult" contains five attributes with string data type. Table 4-5-1-4, shown the attribute name and data types. This table will now be having its own unique primary key, but its primary key will be presenting by the review_id and shop_id. It contain the main information which is the result of sentiment analysis, review date, and time.

Attribute	**Data Type**	**Example Value**	**Description**
review_id	String	[0, 0, 8, 4]_2024-03-13_09:22:24	First primary key in the table.
shop_id	String	S002	Second primary key in the table.
review_date	String	2024-03-13	The date that review and rating is submitted by the user in the application.
review_time	String	09:22:24	The time that review and rating is submitted by the user in the application.
sentimentValue	Number	2	Sentiment value is generated by sentiment analysis system in the application. Which store either -1. 0, 1, and 2. Zero for negative, one for neutral, 2 for positive, and -1 for special cases.

Table 4-5-1-4 Sentiment Analysis Result

4.5.2 Python: Model Development (CRISP-DM)

Figure 4-1-2 and Figurw 4-2-2 show the block diagram and flow diagram with an implementation planning with brief elloboaration. This sub chapter introduce the implemantion planning for the system's model development. Sentiment analysis is one of the topics about machine learning. To perform modelling, there is a methodology that followed to ensure the model developed is in good quality known as Cross-Industry Standard Process for Data Mining (CRISP-DM) is used in this project. There are the steps or processes such as business understanding, data understanding, data preparation, modelling, evaluation, and deployment. In Table 4-5-2-1, it concluded the information about the implementation of each process.

Business understanding is first and important process. It can refer to the one of the project objectives which it is expected to achieve a minimum 80% of accuracy in the testing dataset to ensure the model’s reliability, and effectiveness. Next, in process of **data understanding**. The process is planned to determine the data souces and to understand the data structure and its information. Including the attribute name, data type of attribute, size of the dataset, and value of attributes containing. Hence during variable creation for model development, and application development, the data type and structure will be matched. After data understanding process, the data needed to be prepared and pre-processed for modeling.

This process known as **data preparation**. This process expected to use minimum 500,000 data for modelling. These data needed to be checked for the quality to ensure no null value contain. If there is null value of the attribtues, the data record will be deleted and replace by other. For the null value of review, it will delete the data record. As for, null value of rating, it will be replaced by using the mean of rating. The next planning for the data preparation is to create a new dataset that only contain needed attribute which is about the review and rating. Due to sentiment analysis if based on review and rating. The rating will be resturcure and store into a new attribute what restructure based on the value. The value of rating is from 1 to 5, hence, value of 1 and 2 will replace to 0, 3 repalce to 1, and 4 and 5 replace to 2. It is due to the sentiment analysis provided the result of either negative, neutral, and positive which represent 0, 1 and 2. After the process of restrucuting is complete, the dataset will split into three datasets and perform further pre-processing known as training, validating, and testing dataset. The next preprocessing implementation planning is aimed to perform tokenization based on the model chosen known as BERT. Before tokenization perform, it is expected to load the pre-train model and its resources such as tokenizer. Two data input of modelling is expected to use, which included token ID and attention mask. Both data input will create in data type of array. Token ID store the value of ID of the word based on the BERT tokenizer, and attention mask will repalve value of 1 to non-empty token, and 0 for unknown token. The size of them expects to be maximum 512. Lastly, the dataset will be well prepared for modelling after performing tokenization on the dataset created.

Moving on, the **modelling** known as model training and model validating which expected to have minimum 4 iterations. In each iteration, it is planned to calculate the accuracy and loss for the process of training and validating. Hence, it can provide useful information for evaluation. After the modelling is completed, the next process is known as **model testing or evaluation** by using testing dateset. In this process, it is aimed to calculate different metrics for evaluation purposes including accuracy, f1-socre, recall, precision, and confusion matrix. By using these evaluation metrics, it able to provide concise information for evaluation and comparison.

Lastly, **deployment** process divided into two parts. Firstly, model saving to achieve the purposes of integration into application developed. To ensure the model performance is performing well and effectively. The model is expected to save in H5 file type and convert into ONNX file type. Lastly, converting it to file type of TFLITE file type. As this file type provide a lighter version of model without heavily affecting the performance of model. It also eases the integration process. Next, after model is saved, it will integrade into the application developed and further explain in Chapter 4.4.3. Which loading the resources needed in the Shop Display Activity and perform the sentiment analysis function in a sentiment analysis class.

Process	**Description**
Business Understanding	To train a model, there is a data mining goal must be set. Therefore, in this project, the model is expected to achieve an average accuracy of 80%.
Data Understanding	In this process, the data sources are from Yelp Dataset which provide concrete attribute that perform sentiment analysis. This process is also to understand the nature structure of the data.
Data Preparation	This process prepares well the data before the model is trained according to its requirement. Which is performing text tokenization for the review and data conversion for the rating.
Modelling	The model is expected to have minimum 4 iteration for training and validating.

Evaluation	After the model is trained, there will be few aspects will be examined such as the accuracy, precision, F1-score, recall, and confusion matrix.
Deployment	The model will be saved into TensorFlow Lite. Which enable the model file size to be lighter and easier to integrate to the application from Android Studio.

Table 4-5-2-1 CRISP-DM of Sentiment Analysis Model

4.5.3 Android Studio: Application Development (Model Deployment)

Figure 4-1-1 and Figurw 4-2-1 show the block diagram and flow diagram with an implementation planning with brief elloboaration for the application development. By using Android Studio, it enables to develop an application that perform different function such as store and retrieve data from Firebase’s Firestore database and perform sentiment analysis with saving the result to Firebase’ Firestore database. To achieve the project objectives, there are total main five activity and one class is created. Which is main activity, shop list display activity, shop display activity, shop review display activity, filter activity, and sentiment analysis class. According to Table 4-5-3-1, provided a concluded description on the purpose of function of each activity and class. In every activity, there is a default method created that named “onCreate”. The developers can perform any calculation, setting, or other coding in this method. Within the activity code, the developer can create extra function or class inside the activity to ensure the functionalities of the pages.

The first interface of the application known as Main Activity which planned to display two contents. Where the Figure 4-3-1 show the draft design of the application interface for the main activity or known as main page. Firstly, displaying the categories of shop for user to select for recommending shops. Then, the interface will display the shops under the category selected. Next, the interface displays the top recommended shops according to two criteria set. The criteria such as the average rating is above 4, and the percentage of a shop getting positive review shall be more than or equal to 86%. Therefore, the shops fulfilled the requirement will be displayed according to the categories. The order of the categories will be set to healthcare, technology, stationary and mini market. There are no limits on the number of shops allow to fulfill the requirements. Hence, as long as the shop is having average rating of above 4, and more then or equal to 86% to get positive review. Therefore, the shop is added into the list for diplaying into the Main Activity interface. According to the shop displayed, the user is allowed to choose to view the shop information with different function.

Secondly, when a category of shop is chosen by the user. It will move to next activity known as Shop List Display Activity that displays the shops. According to the draft design shown in Figure 4-3-2, this activity or interface is planned to display all the shop under the category with firstly displaying the shop with having review and rating record, then display the new shop that did not have any record of review and rating. The logic of displaying the shop is expected to display according to the average rating. In this interface, there are three important information plan to display which are shop name, average rating of the shop, and total review and rating the shop received. Other than displaying the list of shops. The activity provided one function for the user, which is a filtering and ranking feature. When user click the button of filtering, it will move to another activity known as Filter Activity.

Shop Display Activity is an important activity. Figure 4-3-3 show the draft design of the page. It plays a vital role in the application. As it is planned to provide different function and information displaying. To access this activity, there are two method is planned. Firstly, the user can access the shop by directly choose the shop display in Main Activity. Next, a list of shops is displayed in Shop List Display Activity, and in this activity, the user can choose the shop to view its detail information in Shop Display Activity. The shop is plan and expected to provide 4 main elements. At first, the activity displayed the shop information including shop name, shop location, shop contact method, and description of the shop. Next, a function of viewing past review and rating is planned to develop which the suer can view all past reviews and ratings of the shop received. When the function is called, it is expected that the application access to another activity known as Shop Review Display Activity. Moving on, in this interface, it display the information of the sentiment analysis result. Including the percentage of the shop getting negative, neutral, and positive review. Lastly, the activity is planned to provide a function of providing review and rating. Where the user is allowed to provide their comment and rating. When submitting the review and rating, there are two functions running in background. Which are saving the review and rating into Firebase Firestore and performing sentiment analysis integrated. To implement the sentiment analysis, the value should be not null. Therefore, if user choose to provide rating only, the system will not perform sentiment analysis. However, the system will still record the rating data for average rating calculation. In this activity, it is planned to load the resources of the model developed and call the function from Sentiment Analysis Class with passing the model loaded.

The next two acvitities are allowed to access then the function is called from Shop Display Activity. These two activities known as Shop Review Display Activity and Filter Activity. Which referring to Figure 4-3-4 and Figure 4-3-5. For the former activity, it plans to display all reviews and ratings of the shop received in the order based on date and time. Few information expected to display including review, rating, date receiving, and time receiving. The later activity provides different criteria for user to set for performing filtering and ranking to recommend the shops that fulfilled requirements. First criteria are about the average rating. Second criteria is about the percentage of the shop getting positive review. Based on that, the user can choose and display the shop that fulfill the criteria.

Lastly, Sentiment Analysis Classs included different function for performing sentiment analysis when the review and rating is given. Firstly, the model loaded from the activity will pass into the function. By loading the model to perform analysis and generate result and saved into Firebase Firestore. Therefore, it is expected to have 4 functions in the class. First function is to perform analysis by using the model loaded in the activity. Second function is to perform data preprocessing by using the BERT and related library in Android Studio. Third and forth function is about the interpretation of the result generated by using softmax and avgmax calculation.

Class / Activity	**Descriptions**
Main Activity	This activity provides a clean and clear interface that show four categories of shop for user to choose types of shop they are looking for and display a few tops rating shops for each category.
Shop List Display Activity	This activity will be displaying the shop name with its average rating and total review received with in descending order of average rating.
Shop Display Activity	This activity consists of detail information displaying and two function provided. Where the information that will be displayed such as shop contact method, location, and so on. Most importantly, it displays the percentage of the shop getting positive, neutral, and

	negative review. It provides also two main functions for user which are viewing past review and submit review and rating.
Shop Review Display Activity	This activity displays the review collected according to the date and time.
Filter Activity	This activity provides a series of choice for user for filtering and ranking purposes. Which mainly based on average rating and the percentage of the shop getting positive review.
Sentiment Analysis Class	This class contain the function of analyze the sentiment of the review and rating that receive from Shop Display Activity. After analyzed, the result will be saved to Firebase.

Table 4-5-3-1 Description of Activities and Class in Android Studio

Table 4-5-3-2 show a list of basic function and library that frequently used in the application development.

Function / Library	**Description**
findViewByID	This function is to be linked and assigned the element such as layout, view, and others to the variables created in activity. Example: TextView shopNameTextView = findViewById(R.id.shopNameTextView);

setOnClickListener	It is mainly for button or view. When it is click, the function created will be called.
	Example:
	shopLayout.setOnClickListener(v -> {
	    Intent intent = new Intent(ShopListDisplay.this, ShopDisplay.class);
	    intent.putExtra("shopID", shopID);
	    startActivity(intent);
	});

Layout 1) setText 2) addView	This are the two basic function for layout. However there are many others. Example: LinearLayout shopLayout = new LinearLayout(ShopListDisplay.this);

	TextView ratingTextView = new TextView(ShopListDisplay.this);
	ratingTextView.setText(" Average Rating: " + averageRating + " Total review: " + reviewCount);
	shopLayout.addView(ratingTextView);

Intent
1) putExtra
2) getString Extra
	Start a new activity and passing data from one to another activity.
	Example:
	Activity A:
	Intent intent = new Intent(ShopListDisplay.this, Filter.class);
	intent.putExtra("shop_cat", shopCat);
	startActivity(intent);
	
	Activity B:
	Intent intent = getIntent();
	String shopCat = intent.getStringExtra("shop_cat");

Firebase
1) put
2) getString
3) getLong
	Example of saving data into Firebase:
	reviewDataFirestore.put("review_id", review_id);
	
	Example of retrieving data from Firebase:
	Long sentimentValue = resultDocument.getLong("sentimentValue");

Table 4-4-3-2 List of Basic Function and Library Used

Chapter 5

System Implementation

This chapter will be introducing the set ups, and system implementation under the process of CRISP-DM methodology.

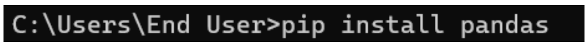
5.1 System Pre-Implementation

5.1.1 Software and Platform Setup

This project takes into account the software packages Android Studio Hedgehog 2023.1.1 Patch 1 and Python-IDLE Shell 3.10.8 with Java and Python. The usage can be divided into two parts. Firstly, the server's algorithm that supports the recommender system is to use python with the knowledge of machine learning to perform sentiment analysis, filtering functions, etc., which build by using Python programming language. And it will be used to connect with the user's mobile application by downloading it in TensorFlow lite model. Secondly, the software used for application development is Android Studio Hedgehog 2023.1.1 Patch 1. Android studio support two type of programming language which are Kolin and Java. In Android studio it is supporting coding to design as well as drag and drop designing. Next, there is a pre-implementation set up for the software used for Android Studio, Python IDLE and Firebase. Table 5-1-1 show a list of libraries that needed for the model development. Figure 5-1-1 to Figure 5-1-6 show the code of downloading and setting up the library in command prompt of the laptop for model development that stated in Table 5-1-1.

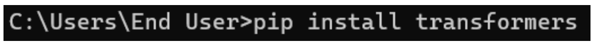
No	Library	Description
1	Pandas	To access the dataset to perform manipulation and analysis.
2	Transformer	To perform natural language processing by accessing BERT and its resources.
3	TensorFlow	To perform modelling by accessing the BERT and its resources.
4	Scikit-learn	To perform data mining and data analysis.
5	TF2ONNX	To convert the model from TensorFlow format to ONNX format.
6	ONNX	To access the model in ONNX format.

Table 5-1-1-1 List of Library Used for Model Development



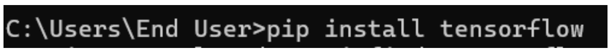
```
C:\Users\End User>pip install pandas
```

Figure 5-1-1 Code of Downloading Pandas Library



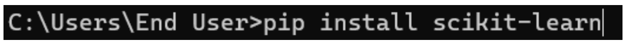
```
C:\Users\End User>pip install transformers
```

Figure 5-1-2 Code of Downloading Pandas Transformers Library



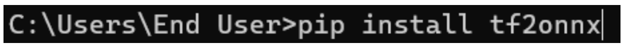
```
C:\Users\End User>pip install tensorflow
```

Figure 5-1-3 Code of Downloading Pandas TensorFlow Library



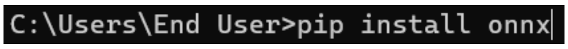
```
C:\Users\End User>pip install scikit-learn
```

Figure 5-1-4 Code of Downloading Pandas Scikit-learn Library



```
C:\Users\End User>pip install tf2onnx
```

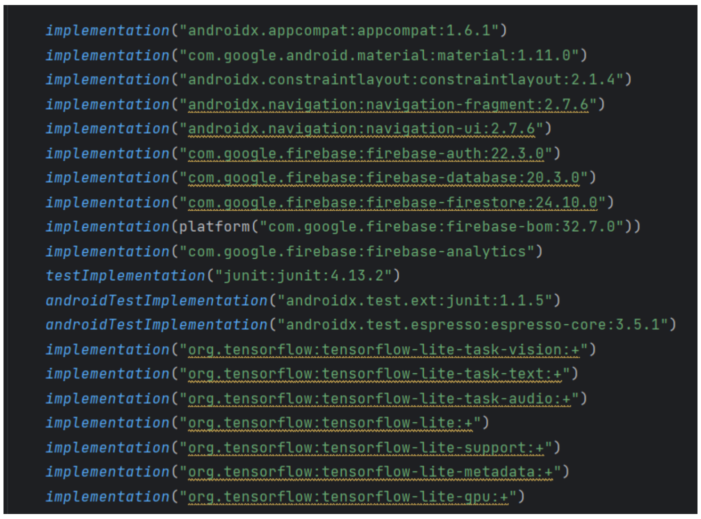
Figure 5-1-5 Code of Downloading tf2onnx Library



```
C:\Users\End User>pip install onnx
```

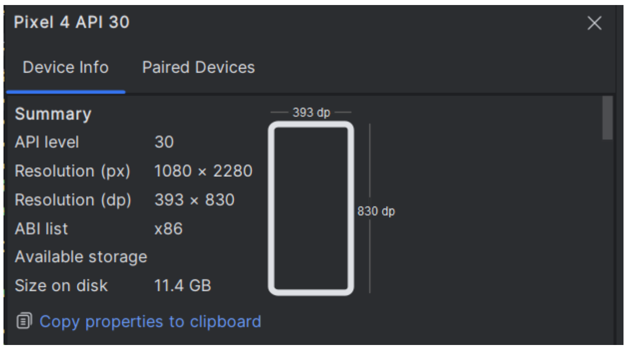
Figure 5-1-6 Code of Downloading onnx Library

To build an application that integrate with a model and Firebase. There are some set up is required. In Figure 5-1-7, it shows the dependencies of the application to allow the application access different library. Such as, TensorFlow, and Firebase. There are two programming language which are Java and Extensible Markup Language (XML) used. Java programming languages mainly support on building functionalities of the application, while XML mainly for interface design such as creating different layout, view, and so on. Next, a virtual device is created for interface design (Figure 5-1-8)



```
implementation("androidx.appcompat:appcompat:1.6.1")
implementation("com.google.android.material:material:1.11.0")
implementation("androidx.constraintlayout:constraintlayout:2.1.4")
implementation("androidx.navigation:navigation-fragment:2.7.6")
implementation("androidx.navigation:navigation-ui:2.7.6")
implementation("com.google.firebase:firebase-auth:22.3.0")
implementation("com.google.firebase:firebase-database:20.3.0")
implementation("com.google.firebase:firebase-firestore:24.10.0")
implementation(platform("com.google.firebase:firebase-bom:32.7.0"))
implementation("com.google.firebase:firebase-analytics")
testImplementation("junit:junit:4.13.2")
androidTestImplementation("androidx.test.ext:junit:1.1.5")
androidTestImplementation("androidx.test.espresso:espresso-core:3.5.1")
implementation("org.tensorflow:tensorflow-lite-task-vision:+")
implementation("org.tensorflow:tensorflow-lite-task-text:+")
implementation("org.tensorflow:tensorflow-lite-task-audio:+")
implementation("org.tensorflow:tensorflow-lite:+")
implementation("org.tensorflow:tensorflow-lite-support:+")
implementation("org.tensorflow:tensorflow-lite-metadata:+")
implementation("org.tensorflow:tensorflow-lite-gpu:+")
```

Figure 5-1-7 Code of Android Studio: Dependencies

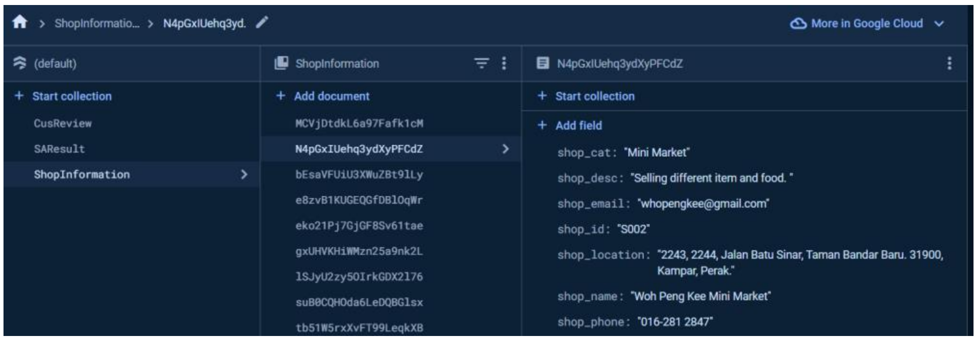


Summary

API level	30
Resolution (px)	1080 × 2280
Resolution (dp)	393 × 830
ABI list	x86
Available storage	
Size on disk	11.4 GB

Figure 5-1-8 Virtual Device Creation

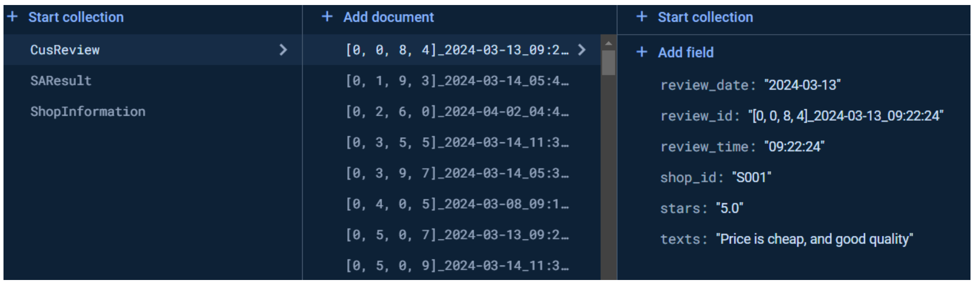
According to the Figure 5-1-9 to Figure 5-11 show the actualy table in the Firebase Firestore of Shop information, Customer Review, and Sentiment Analysis Result. According to Figure 5-1-12, it show the query rule set in the Firebase, which allow the application perform query with a deadline of until year of 2025 February.



```
MCVjDtdkL6a97Fafk1cM
N4pGxIUehq3ydXyPFCdZ
bEsaVFUiU3XWuZBt9lLy
e8zvB1KUGEQGfDBlOqWr
eko21Pj7GjGF8Sv61tae
gxUHVKHiWMzn25a9nk2L
1SJyU2zy5OIrkGDX2176
suB0CQHOda6LeDQBGlsx
tb51W5rxXvFT99LeqkXB
```

```
shop_cat: "Mini Market"
shop_desc: "Selling different item and food. "
shop_email: "whopengkee@gmail.com"
shop_id: "S002"
shop_location: "2243, 2244, Jalan Batu Sinar, Taman Bandar Baru. 31900,
                Kampar, Perak."
shop_name: "Woh Peng Kee Mini Market"
shop_phone: "016-281 2847"
```

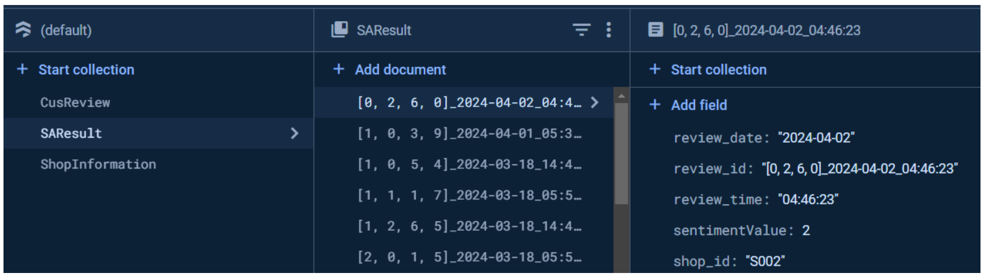
Figure 5-1-9 Shop Information in Firebase



```
[0, 0, 8, 4]_2024-03-13_09:2...
[0, 1, 9, 3]_2024-03-14_05:4...
[0, 2, 6, 0]_2024-04-02_04:4...
[0, 3, 5, 5]_2024-03-14_11:3...
[0, 3, 9, 7]_2024-03-14_05:3...
[0, 4, 0, 5]_2024-03-08_09:1...
[0, 5, 0, 7]_2024-03-13_09:2...
[0, 5, 0, 9]_2024-03-14_11:3...
```

```
review_date: "2024-03-13"
review_id: "[0, 0, 8, 4]_2024-03-13_09:22:24"
review_time: "09:22:24"
shop_id: "S001"
stars: "5.0"
texts: "Price is cheap, and good quality"
```

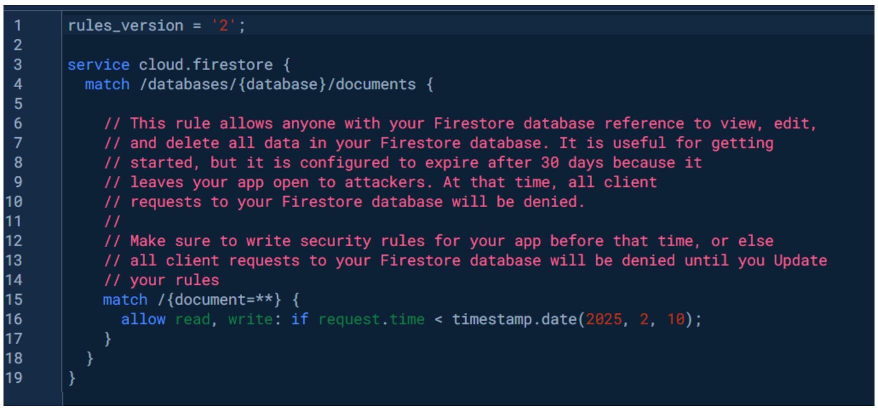
Figure 5-1-10 Customer Review in Firebase



```
[0, 2, 6, 0]_2024-04-02_04:4...
[1, 0, 3, 9]_2024-04-01_05:3...
[1, 0, 5, 4]_2024-03-18_14:4...
[1, 1, 1, 7]_2024-03-18_05:5...
[1, 2, 6, 5]_2024-03-18_14:4...
[2, 0, 1, 5]_2024-03-18_05:5...
```

```
review_date: "2024-04-02"
review_id: "[0, 2, 6, 0]_2024-04-02_04:46:23"
review_time: "04:46:23"
sentimentValue: 2
shop_id: "S002"
```

Figure 5-1-11 Sentiment Analysis Result in Firebase



```
rules_version = '2';

service cloud.firestore {
  match /databases/{database}/documents {

    // This rule allows anyone with your Firestore database reference to view, edit,
    // and delete all data in your Firestore database. It is useful for getting
    // started, but it is configured to expire after 30 days because it
    // leaves your app open to attackers. At that time, all client
    // requests to your Firestore database will be denied.
    //
    // Make sure to write security rules for your app before that time, or else
    // all client requests to your Firestore database will be denied until you Update
    // your rules
    match /{document=**} {
      allow read, write: if request.time < timestamp.date(2025, 2, 10);
    }
  }
}
```

Figure 5-1-12 Firebase Firestore Query Rule Setting

5.1.2 Hardware Setup

This project uses a computer and an android mobile device as its hardware. The software that was used to code the system is run on a computer. The internet application for the in-store shopping recommender is tested on a mobile device running any OS.

Description	**Specifications**
Model	HP Pavilion Laptop 15-cs3xxx
Processor	Intel(R) Core(TM) i5-1035G1 CPU @ 1.00GHz 1.19 GHz
Operating System	Windows 11
Graphic	NVIDIA GeForce GTX MX250 4GB
Memory	8GB Kingston 3200MHz + 4GB Samsung 2667MHz
Storage	500GB SSD

Table 5-1-2-1 Specifications of laptop

5.2 System Implementation: Model Development (CRISP-DM)

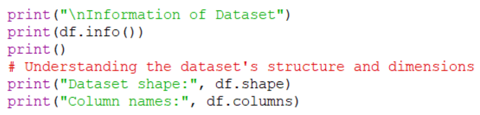
According to Chapter 3 and the implementation planning from Chaper 4. The machine learning: sentiment analysis model development will be using CRISP-DM which known as Cross-Industry Standard Process for Data Mining. Which contain 6 phases which are business understanding, data understanding, data preparation, modelling, evaluation, and deployment. In this part, I will be explaining in details about how the sentiment analysis model is being developed.

5.2.1 Business Understanding

The project contains two main important part which is model development and application development for shop recommendation system. The project's objectives are to create a sentiment analysis system which trained a model and use it in application that developed in Android Studio. The goal of the model is to achieve at least 80% percent of accuracy of model testing. As well as to evaluate the performance of mode in accuracy of model training and validating. The model is expected to have minimum 4 iteration for training and validating. Therefore, the evaluation of the model performance is expected to calculate and evaluate the average of accuracy of iteration for training, validating, and testing.

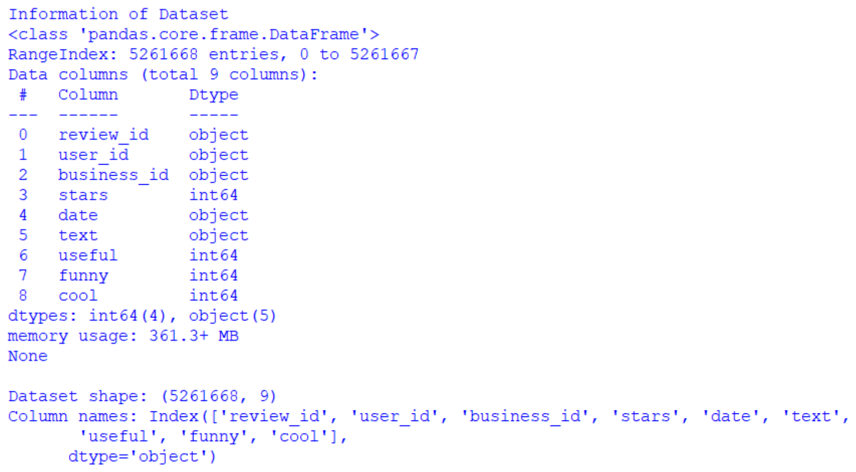
5.2.2 Data Understanding

The dataset used for model development was downloaded from Yelp dataset from Kaggle platform. Which contain the basis data and information needed for the analysis model. Data such as review, rating, and other. (Link:https://www.kaggle.com/datasets/yelp-dataset/yelp-dataset/versions/6?resource=download). This dataset contains information on 174,000 businesses from four country and 11 city such as Las Vegas and Phoenix. In this process, study the data and attributes in the CSV file which is for model training purposes. In this project, data understanding divided into two main parts. Firstly, basic information of the dataset such as size of dataset, attributes with its data type. Second, understand the format and structure of the attribute value that stored. Hence, Figure 5-2-2-1 refers to the code of data understanding for first part. Which enable me to understand the basic information of the dataset. And Figure 5-2-2-2 is the result of the code in Figure 5-2-2-1. According to Figure 5-2-2-2, this data set consist of 5,261,668 data with 9 different attributes. The attribute are "review_id", "user_id", "business_id", "date" in data type of object or known as string, and "text", "stars", "useful", "funny", and "cool" in data type of integer. In this project, there are total number of 5,261,668 data in the dataset. However, the model development used a minimum 100,000 data for dataset separation into training, validating, and testing dataset. To reduce the time consuming for the code running due to computational resources limitation. While the number of iterations for training data shall fall between 4 to 6. Number of iterations will increase when it is sufficient which cause low effectiveness of the model.



```
print("\nInformation of Dataset")
print(df.info())
print()
# Understanding the dataset's structure and dimensions
print("Dataset shape:", df.shape)
print("Column names:", df.columns)
```

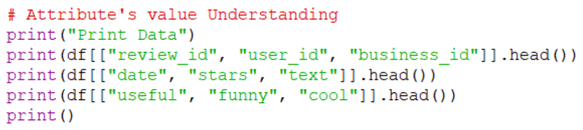
Figure 5-2-2-1 Code of Data Understanding: First Part_Basic Information



```
Information of Dataset
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 5261668 entries, 0 to 5261667
Data columns (total 9 columns):
 #   Column       Dtype 
---  ------       ----- 
 0   review_id    object
 1   user_id      object
 2   business_id  object
 3   stars        int64 
 4   date         object
 5   text         object
 6   useful       int64 
 7   funny        int64 
 8   cool         int64 
dtypes: int64(4), object(5)
memory usage: 361.3+ MB
None

Dataset shape: (5261668, 9)
Column names: Index(['review_id', 'user_id', 'business_id', 'stars', 'date', 'text',
       'useful', 'funny', 'cool'],
      dtype='object')
```

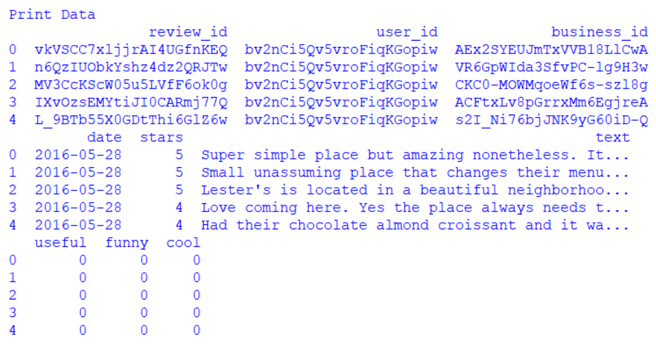
Figure 5-2-2-2 Result of Data Understanding: First Part_Basic Information

Next, Figure 5-2-2-3, show the value stored of the attribute in the dataset. However, the sentiment analysis model development in this project, there are only two main attributes used which are review and attribute name is "text", and rating with attribute name is "stars".



```
# Attribute's value Understanding
print("Print Data")
print(df[["review_id", "user_id", "business_id"]].head())
print(df[["date", "stars", "text"]].head())
print(df[["useful", "funny", "cool"]].head())
print()
```

Figure 5-2-2-3 Code of Data Understanding: Second Part_Value Stored



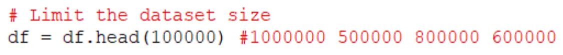
```
Print Data
                review_id                 user_id             business_id
0  vkVSCC7xljjrAI4UGfnKEQ  bv2nCi5Qv5vroFiqKGopiw  AEx2SYEUJmTxVVB18LlCwA
1  n6QzIUObkYshz4dz2QRJTw  bv2nCi5Qv5vroFiqKGopiw  VR6GpWIda3SfvPC-lg9H3w
2  MV3CcKScW05u5LVfF6ok0g  bv2nCi5Qv5vroFiqKGopiw  CKC0-MOWMqoeWf6s-szl8g
3  IXvOzsEMYtiJI0CARmj77Q  bv2nCi5Qv5vroFiqKGopiw  ACFtxLv8pGrrxMm6EgjreA
4  L_9BTb55X0GDtThi6GlZ6w  bv2nCi5Qv5vroFiqKGopiw  s2I_Ni76bjJNK9yG60iD-Q
         date  stars                                               text
0  2016-05-28      5  Super simple place but amazing nonetheless. It...
1  2016-05-28      5  Small unassuming place that changes their menu...
2  2016-05-28      5  Lester's is located in a beautiful neighborhoo...
3  2016-05-28      4  Love coming here. Yes the place always needs t...
4  2016-05-28      4  Had their chocolate almond croissant and it wa...
   useful  funny  cool
0       0      0     0
1       0      0     0
2       0      0     0
3       0      0     0
4       0      0     0
```

Figure 5-2-2-4 Result of Data Understanding: Second Part_Value Stored

5.2.3 Data Preparation

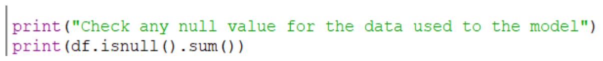
Data preparation also known as data preprocessing. It is a step that prepare the data well for model training. There are few activities that implemented in this process to ensure data quality by selecting and creating and constructing data. Total 100,000 data used for model development (Figure 5-2-3-1). Initially, based on the implementation planning stated in Chapter 4, the size of the dataset shall be minimum 500,000; however, due to the time and computational resources consume heavily and seriously. Hence, during the model development process, the size fo dataset was changed from 500,000 to 100,000.

After the size of dataset is selected. Firstly, is to ensure the data quality, there must no null value of the attributes for the model development. Hence, by performing a function of checking null value of attributes, I can examine the data have null value. If there is null value, the data needed to be performed further action such as removing the data from the dataset. In Figure 5-2-3-2, it is the code to check the total number of data having null value. Figure 5-2-3-3 show that the dataset is not having any null value, hence the quality of data is ensured and there is no further action required.



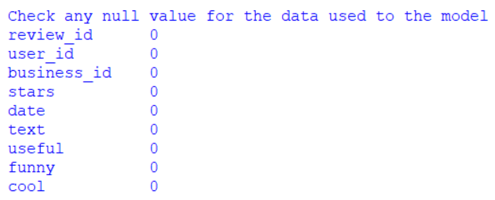
```
# Limit the dataset size
df = df.head(100000) #1000000 500000 800000 600000
```

Figure 5-2-3-1 Code of Data Preparation: Dataset Size Setup



```
print("Check any null value for the data used to the model")
print(df.isnull().sum())
```

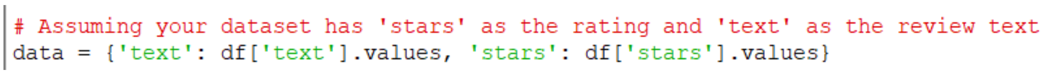
Figure 5-2-3-2 Code of Data Preparation: Null Value of Data Checking



```
Check any null value for the data used to the model
review_id      0
user_id        0
business_id    0
stars          0
date           0
text           0
useful         0
funny          0
cool           0
```

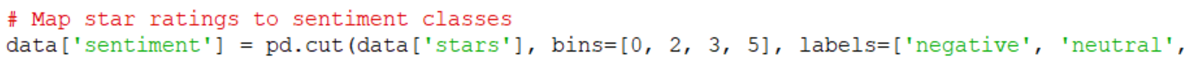
Figure 5-2-3-3 Result of Data Preparation: Null Value of Data Checking

Next, the sentiment analysis model required two data which is the rating and review. In the dataset, it refers to the text and stars. For that, according to the planning in Chapter 4, a new dataset is created with retrieving the attribute from initial dataset and named it as “data” which only contain of the attributes of “text” and “stars” (Figure 5-2-3-4). Next, the model requires the rating known as stars to restructure to 3 group which is value of 0, 1, and 2 from rating of 1 to 5. Which 0 stand for negative, 1 for neutral, and positive is 2. Where the stars of 1 and 2 out of 5 will categories into 0, 3 stars categories into 1, and 4 to 5 stars is value of 2. Figure 5-3-3-5 show the code of restructure the attribute by creating new attribute named as “sentiment”. By using “cut” function from Pandas Library.



```
# Assuming your dataset has 'stars' as the rating and 'text' as the review text
data = {'text': df['text'].values, 'stars': df['stars'].values}
```

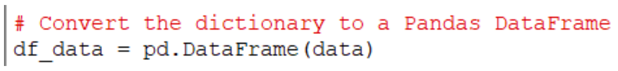
Figure 5-2-3-4 Code of Data Preparation: Creating New Dataset that Contain the Attribute Needed



```
# Map star ratings to sentiment classes
data['sentiment'] = pd.cut(data['stars'], bins=[0, 2, 3, 5], labels=['negative', 'neutral',
```

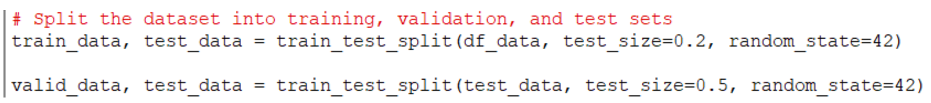
Figure 5-2-3-5 Code of Data Preparation: Creating New Attribute by Restructure the Value of Attribute Stars

There is total three dataset is needed to develop a sentiment analysis model. Which known as training data, validation data, and testing data. Before creating these data, the dataset that named in “data” converted into a Pandas Framework named as “df_data” that show in Figure 5-2-3-6. After that, Figure 5-2-3-7 show the code of creating the three datasets. Due to the size of dataset is reduce to 100,000. The total amount of data for training dataset was 80,000, and 10,000 for each validating and testing.



```
# Convert the dictionary to a Pandas DataFrame
df_data = pd.DataFrame(data)
```

Figure 5-2-3-6 Code of Data Preparation: Convert Dictionary to A Pandas Framework



```
# Split the dataset into training, validation, and test sets
train_data, test_data = train_test_split(df_data, test_size=0.2, random_state=42)

valid_data, test_data = train_test_split(test_data, test_size=0.5, random_state=42)
```

Figure 5-2-3-7 Code of Data Preparation: Creation of Training, Validating, and Testing Dataset

After creating the data, the date needs to be tokenized and structure well. To achieve that, a tokenizer and pre-trained model is loaded. It is a tokenizer from BERT. A tokenizer plays a important role for sentiment analysis model, as it include a certain number of vocabulary that can be defined into two types known as bert-base-cased and bert-base-uncased. The former refer to a text case-sensitive which treat lower case word and upper-case word different like. For example, “Apple” and “apple”. The later refer to word will be converted to lower case before it tokenized. Figure 5-2-3-8 show how the steps are coded. For the pre-trained model, it indicates a number of labels of three which refer to negative, neutral, and positive. In this project, TensorFlow is the library that used for accessing the pre-trained model and training the model.

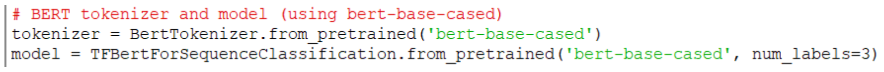
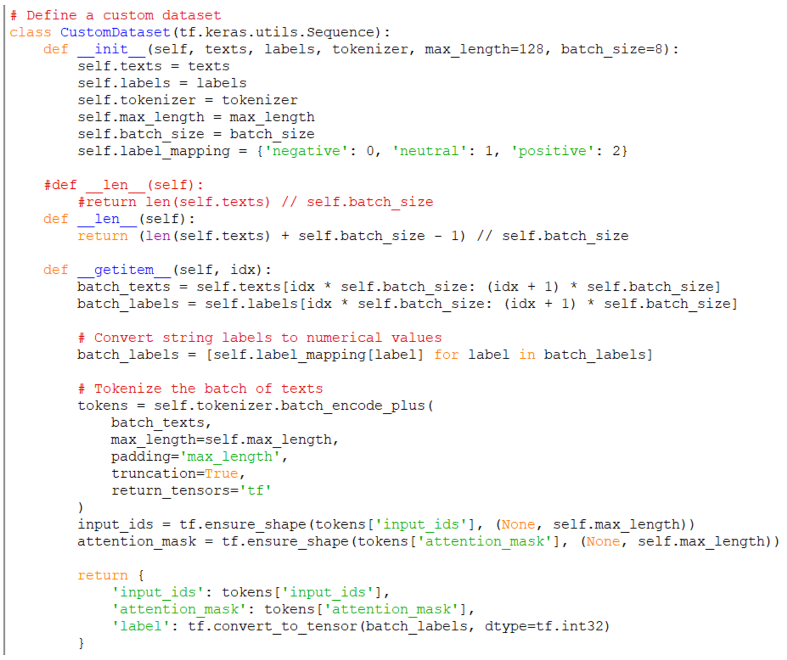

```
# BERT tokenizer and model (using bert-base-cased)
tokenizer = BertTokenizer.from_pretrained('bert-base-cased')
model = TFBertForSequenceClassification.from_pretrained('bert-base-cased', num_labels=3)
```

Figure 5-2-3-8 Code of Data Preparation: Resources Loaded

Based on the implementation planning in Chapter 4, to perform the tokenization to ensure the data is well prepared for modelling, a class and a method is created. The class named as “CustomDataset”. The purpose of this class is to create an object for each data by putting the require attributes with the requirement set. In Figure 5-2-3-9 show the code of the class. In the class, the first function known as constructor method named “__init__” that contain 6 variables. Which included the texts refer to the customer’s review, labels refer to the stars that restructured, tokenizer refer to the list of vocabulary, max length refer to the length that the text is tokenized, and batch size refer to the number of inputs is being processed together in one iteration. As for this project, the batch size is 8, therefore 8 input will be processed together for modelling. Second method of the class is to calculate the number of batches need for the iteration over the dataset, and this method named as “__len__”. Next, there is a method named as “__getitem__”. Which use for retrieving batch of data for training, validating, or evaluation. There are some components. Firstly, “batch_texts” refer to retrieve data from “texts” based on the current index. Next “batch_labels” retrieve the data of the labels for the batch of texts and convert them into numerical values using “label_mapping” dictionary. After retrieve all needed data, tokens are created with the requirement set. To perform tokenization, it needed the batch of text, within the max length set which are 128. After tokenization is completed, it will perform padding and truncation for the empty or null value. Lastly, return the value in “tf”.

During data preprocessing, the input will be tokenized by using the BERT tokenizer. Which the input will split including special token such as CLS and SEP. CLS stand for classification by adding a token in every beginning of a sentences. SEP means separator which is to add a token between two sentences. Moving on, the text will be lowercased and remove extra whitespace. Then for the known token will be extract the token ID based on the tokenizer. For example, the token ID of "good" word is 1234. Next, replacing a value 0 for remaining unknown token. Lastly, ensuring the maximum token size is 128. Example is given in Figure 5-2-3-11.

However, during the process, there is an issue faced. The initial value of batch size and max length is set to 32 and 512. However, the process taking unexpected heavily on computational resources and time. Therefore, the setting such as batch size of value 8 and max length of value 128 is to resolve the issues.



```
# Define a custom dataset
class CustomDataset(tf.keras.utils.Sequence):
    def __init__(self, texts, labels, tokenizer, max_length=128, batch_size=8):
        self.texts = texts
        self.labels = labels
        self.tokenizer = tokenizer
        self.max_length = max_length
        self.batch_size = batch_size
        self.label_mapping = {'negative': 0, 'neutral': 1, 'positive': 2}

    #def __len__(self):
        #return len(self.texts) // self.batch_size
    def __len__(self):
        return (len(self.texts) + self.batch_size - 1) // self.batch_size

    def __getitem__(self, idx):
        batch_texts = self.texts[idx * self.batch_size: (idx + 1) * self.batch_size]
        batch_labels = self.labels[idx * self.batch_size: (idx + 1) * self.batch_size]

        # Convert string labels to numerical values
        batch_labels = [self.label_mapping[label] for label in batch_labels]

        # Tokenize the batch of texts
        tokens = self.tokenizer.batch_encode_plus(
            batch_texts,
            max_length=self.max_length,
            padding='max_length',
            truncation=True,
            return_tensors='tf'
        )
        input_ids = tf.ensure_shape(tokens['input_ids'], (None, self.max_length))
        attention_mask = tf.ensure_shape(tokens['attention_mask'], (None, self.max_length))

        return {
            'input_ids': tokens['input_ids'],
            'attention_mask': tokens['attention_mask'],
            'label': tf.convert_to_tensor(batch_labels, dtype=tf.int32)
        }
```

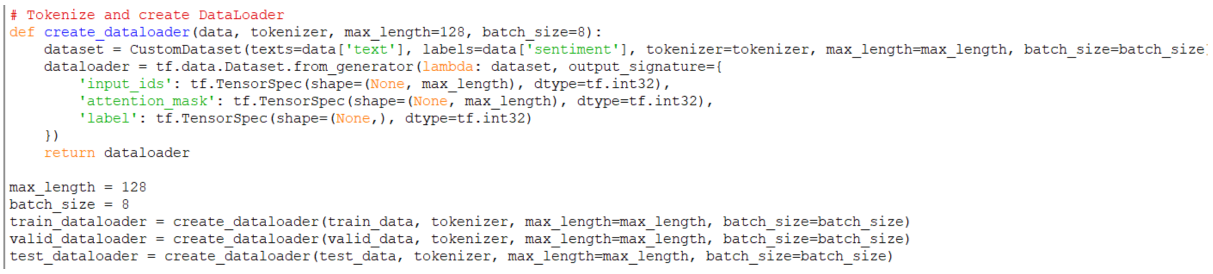
Figure 5-2-3-9 Code of Data Preparation: Custom Dataset Class

At the end of the code shown in Figure 5-2-3-9, three variable will be returned to the object. Which are "input_ids", "attention_mask", and "label". Table 5-2-3-1 show the description of these three variables. Input ID is referred to an ID for each word of the text, and adding special ID such as SEP and CLS.

Variables	**Description**
input_ids	After the sentences of the customer review is being tokenized. It will store the id matched according to the vocabulary from tokenizer of bert_based_case.
attention_mask	In this project, the max length is 128 which maximum 128 tokenized text will be stored. Therefore, attention_mask is an array that store either value 0 or 1. Value 1 stand for there is a text tokenized with an ID, value 0 stand for a null value or unknown token.
label	It contains three value which known as 0, 1, and 2. 0 stand for negative, 1 for neutral, and 2 for positive.

Table 5-2-3-1 Variables Description of Custom Dataset Class

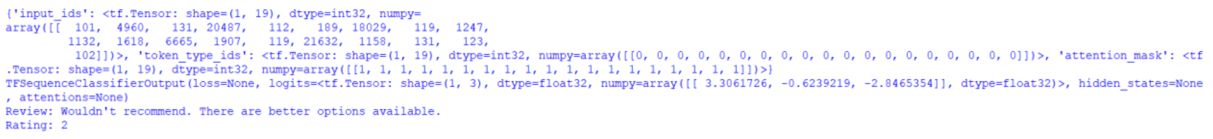
After the class is created, a function is created that can be applied to the dataset that created which is training set, validating set, and testing set. To perform this, a function named "create_dataloader" is created. After the function is called, train_dataloader, valid_dataloader, and test_dataloader is created. (Figure 5-2-3-10).



```
# Tokenize and create DataLoader
def create_dataloader(data, tokenizer, max_length=128, batch_size=8):
    dataset = CustomDataset(texts=data['text'], labels=data['sentiment'], tokenizer=tokenizer, max_length=max_length, batch_size=batch_size
    dataloader = tf.data.Dataset.from_generator(lambda: dataset, output_signature={
        'input_ids': tf.TensorSpec(shape=(None, max_length), dtype=tf.int32),
        'attention_mask': tf.TensorSpec(shape=(None, max_length), dtype=tf.int32),
        'label': tf.TensorSpec(shape=(None,), dtype=tf.int32)
    })
    return dataloader

max_length = 128
batch_size = 8
train_dataloader = create_dataloader(train_data, tokenizer, max_length=max_length, batch_size=batch_size)
valid_dataloader = create_dataloader(valid_data, tokenizer, max_length=max_length, batch_size=batch_size)
test_dataloader = create_dataloader(test_data, tokenizer, max_length=max_length, batch_size=batch_size)
```

Figure 5-2-3-10 Code of Data Preparation: Class Loader for Train, Validation, and Test Dataset

Figure 5-2-3-11 show an example of the input ID and attention mask after performing tokenization. Firstly, it determines the word by search and defining the token with eh vocabulary library from BERT and place with the ID. The size of the input shall be from 1 or none to 128. If the array is having empty token, it will replace them by value 0.



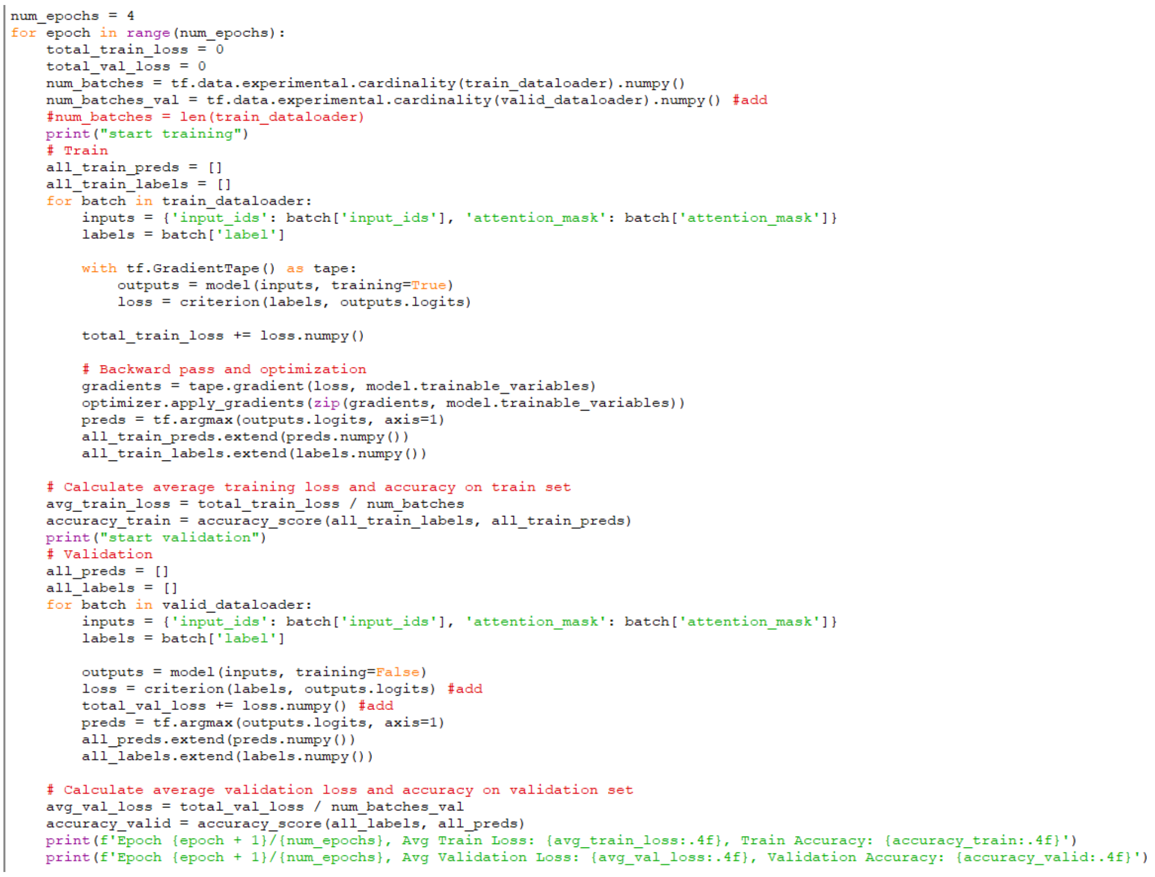
```
{'input_ids': <tf.Tensor: shape=(1, 19), dtype=int32, numpy=
array([[  101,  4960,   131, 20487,   112,   189, 18029,   119,  1247,
         1132,  1618,  6665,  1907,   119, 21632,  1158,   131,   123,
          102]])>, 'token_type_ids': <tf.Tensor: shape=(1, 19), dtype=int32, numpy=array([[0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0]])>, 'attention_mask': <tf
.Tensor: shape=(1, 19), dtype=int32, numpy=array([[1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1]])>}
TFSequenceClassifierOutput(loss=None, logits=<tf.Tensor: shape=(1, 3), dtype=float32, numpy=array([[ 3.3061726, -0.6239219, -2.8465354]], dtype=float32)>, hidden_states=None
, attentions=None)
Review: Wouldn't recommend. There are better options available.
Rating: 2
```

Figure 5-2-3-11 Example of Input ID and Attention Mask

After three datasets completed the function performed. It will proceed to next phases which is modelling, known as model training.

5.2.4 Modelling

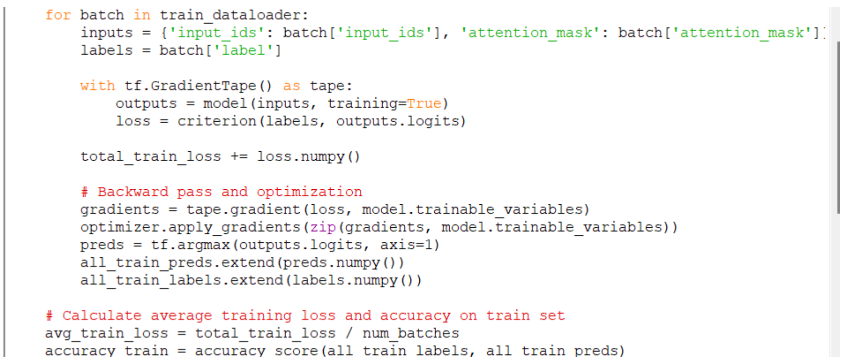
Figure 5-2-4-1 present the complete code of modelling. According to the implementation planning in Chapter 4, the model development shall have a minimum four iterations for its training. To achieve this, for loop is used for modelling process. However, the initial value of the iteration was set to 8, but the entire process of model development taking heavily on computational resources and time. Therefore, few testing is implemented to find the suitable value of iteration. Eventually, the value of 4 iteration is set which also the monimum iteration planned. In these four iterations, it will perform training and validating with its own dataset. To examine the goal achievement, the total training loss and total validation loss attribute is created and defined to a value of 0 which named it as “total_train_loss”, and “total_val_loss” The number of batches is defined by creating attribute as well. Which named it as “num_batches” and “num_batches_val” that use for training and validating. They are calculating the number of batches for training and validating dataset that initially is in TensorFlow and convert into a NumPy array. After that the training will started by creating two arrays for the training prediction and training labels. An attribute named as “inputs” will be created to store the “input_ids” and “attention_mask”, and labels is created to store the value.



```
num_epochs = 4
for epoch in range(num_epochs):
    total_train_loss = 0
    total_val_loss = 0
    num_batches = tf.data.experimental.cardinality(train_dataloader).numpy()
    num_batches_val = tf.data.experimental.cardinality(valid_dataloader).numpy() #add
    #num_batches = len(train_dataloader)
    print("start training")
    # Train
    all_train_preds = []
    all_train_labels = []
    for batch in train_dataloader:
        inputs = {'input_ids': batch['input_ids'], 'attention_mask': batch['attention_mask']}
        labels = batch['label']

        with tf.GradientTape() as tape:
            outputs = model(inputs, training=True)
            loss = criterion(labels, outputs.logits)

        total_train_loss += loss.numpy()

        # Backward pass and optimization
        gradients = tape.gradient(loss, model.trainable_variables)
        optimizer.apply_gradients(zip(gradients, model.trainable_variables))
        preds = tf.argmax(outputs.logits, axis=1)
        all_train_preds.extend(preds.numpy())
        all_train_labels.extend(labels.numpy())

    # Calculate average training loss and accuracy on train set
    avg_train_loss = total_train_loss / num_batches
    accuracy_train = accuracy_score(all_train_labels, all_train_preds)
    print("start validation")
    # Validation
    all_preds = []
    all_labels = []
    for batch in valid_dataloader:
        inputs = {'input_ids': batch['input_ids'], 'attention_mask': batch['attention_mask']}
        labels = batch['label']

        outputs = model(inputs, training=False)
        loss = criterion(labels, outputs.logits) #add
        total_val_loss += loss.numpy() #add
        preds = tf.argmax(outputs.logits, axis=1)
        all_preds.extend(preds.numpy())
        all_labels.extend(labels.numpy())

    # Calculate average validation loss and accuracy on validation set
    avg_val_loss = total_val_loss / num_batches_val
    accuracy_valid = accuracy_score(all_labels, all_preds)
    print(f'Epoch {epoch + 1}/{num_epochs}, Avg Train Loss: {avg_train_loss:.4f}, Train Accuracy: {accuracy_train:.4f}')
    print(f'Epoch {epoch + 1}/{num_epochs}, Avg Validation Loss: {avg_val_loss:.4f}, Validation Accuracy: {accuracy_valid:.4f}')
```

Figure 5-2-4-1 Code of Modelling: Complete Modelling

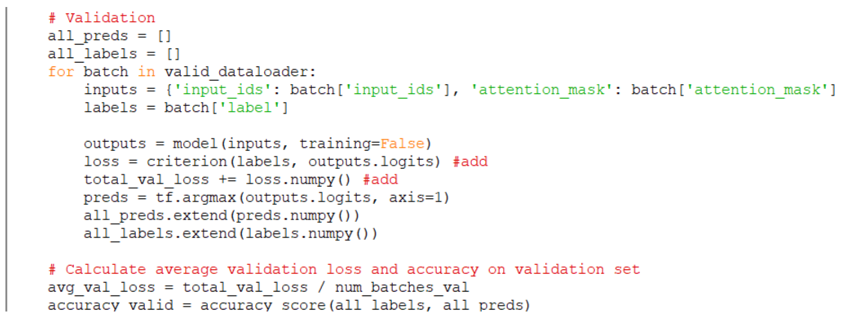
Each iteration will start from training, coming with validating. For training, figure 5-2-4-2 show the for loop in the code. In this for loop, it is focusing on train_dataloader at first. Two variables will be created named as “inputs” and “labels”. “inputs” is a variable that store the “input_ids” and “attention_mask”. As for the value or purpose of “label”, “input_ids”, and “attention_mask”, there is a description stated in Table 5-2-3-1. During training, tf.GradientTape() is used for automatic differentiation purposes which allow to track the operation for training. With this, the output will be predicted by passing the input to the model for training. Next, criterion is a function or method that used for loss calculation. By comparing two input which are the labels and output logits. The labels is part of the data input for model training that is restructured from the initial attribute named stars. It is not predicted by the model instead a true label. Hence, will compare them to calculate the loss. Next, the attribute “total_train_loss” will sum up all the loss for all batches of data. Next, “gradients” is created for ensuring the loss function if respect to the trainable variables of the model. And “optimizer.apply_gradients(zip(gradients, model.trainable_variables))” is code that ensuring the parameters of the model is updating to reduces the loss by adjusting it to let the model learn and perform better. To ensure the output is understandable for human, a TensorFlow function named as argmax is called for computes the value of output logits into a maximum values and store in a variable named as “preds” indicate prediction from the model. At last of the training, the prediction and labels will be store in the array variable corresponding which are “all_train_variable” and “all_train_extend”. Before start validating, there is a calculation for average training loss and accuracy on the training dataset. By dividing the total train loss with the number of batches can return the average of loss for the model training. For accuracy, a function is called from the sklearn metrics with data input of all train labels and all train prediction.



```
for batch in train_dataloader:
    inputs = {'input_ids': batch['input_ids'], 'attention_mask': batch['attention_mask']}
    labels = batch['label']

    with tf.GradientTape() as tape:
        outputs = model(inputs, training=True)
        loss = criterion(labels, outputs.logits)

    total_train_loss += loss.numpy()

    # Backward pass and optimization
    gradients = tape.gradient(loss, model.trainable_variables)
    optimizer.apply_gradients(zip(gradients, model.trainable_variables))
    preds = tf.argmax(outputs.logits, axis=1)
    all_train_preds.extend(preds.numpy())
    all_train_labels.extend(labels.numpy())

# Calculate average training loss and accuracy on train set
avg_train_loss = total_train_loss / num_batches
accuracy_train = accuracy_score(all_train_labels, all_train_preds)
```

Figure 5-2-4-2 Code of Modelling: Training

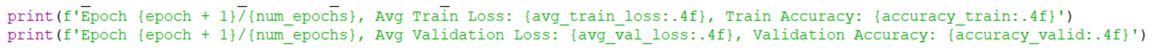
After model training is completed for one iteration. Validation will be implemented with overall same logic with model training. However, there are two main differences. Firstly, is the dataset used. There is a unique dataset for validation, hence there will be no duplication on the dataset. Next, during validating, tf.GradientTape() is not used at it only used during training.



```
# Validation
all_preds = []
all_labels = []
for batch in valid_dataloader:
    inputs = {'input_ids': batch['input_ids'], 'attention_mask': batch['attention_mask']
    labels = batch['label']

    outputs = model(inputs, training=False)
    loss = criterion(labels, outputs.logits) #add
    total_val_loss += loss.numpy() #add
    preds = tf.argmax(outputs.logits, axis=1)
    all_preds.extend(preds.numpy())
    all_labels.extend(labels.numpy())

# Calculate average validation loss and accuracy on validation set
avg_val_loss = total_val_loss / num_batches_val
accuracy_valid = accuracy_score(all_labels, all_preds)
```

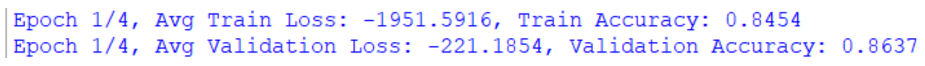
Figure 5-2-4-3 Code of Modelling: Validating

For one iteration is ended, it will print the outputs of the average training loss, validating loss, accuracy of training, and accuracy of validating. Therefore, there will be total 4 output per iteration, and 16 output for all iteration (Figure 5-2-4-4). Lastly, Figure 5-2-4-4 to Figure 5-2-4-8 show the results printed. However, initially there is a missing part of calculating the accuracy and loss for validation. Which cause the evaluation of the performance is not reliable.



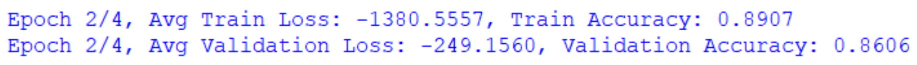
```
print(f'Epoch {epoch + 1}/{num_epochs}, Avg Train Loss: {avg_train_loss:.4f}, Train Accuracy: {accuracy_train:.4f}')
print(f'Epoch {epoch + 1}/{num_epochs}, Avg Validation Loss: {avg_val_loss:.4f}, Validation Accuracy: {accuracy_valid:.4f}')
```

Figure 5-2-4-4 Code of Modelling: Result Printing of Modelling



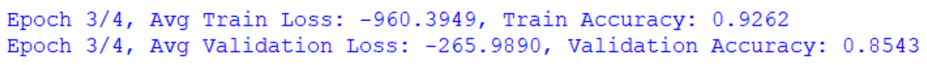
```
Epoch 1/4, Avg Train Loss: -1951.5916, Train Accuracy: 0.8454
Epoch 1/4, Avg Validation Loss: -221.1854, Validation Accuracy: 0.8637
```

Figure 5-2-4-5 Evaluation of 1st Iteration of Modelling



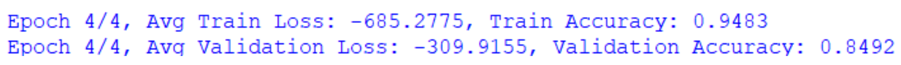
```
Epoch 2/4, Avg Train Loss: -1380.5557, Train Accuracy: 0.8907
Epoch 2/4, Avg Validation Loss: -249.1560, Validation Accuracy: 0.8606
```

Figure 5-2-4-6 Evaluation of 2nd Iteration of Modelling



```
Epoch 3/4, Avg Train Loss: -960.3949, Train Accuracy: 0.9262
Epoch 3/4, Avg Validation Loss: -265.9890, Validation Accuracy: 0.8543
```

Figure 5-2-4-7 Evaluation of 3rd Iteration of Modelling

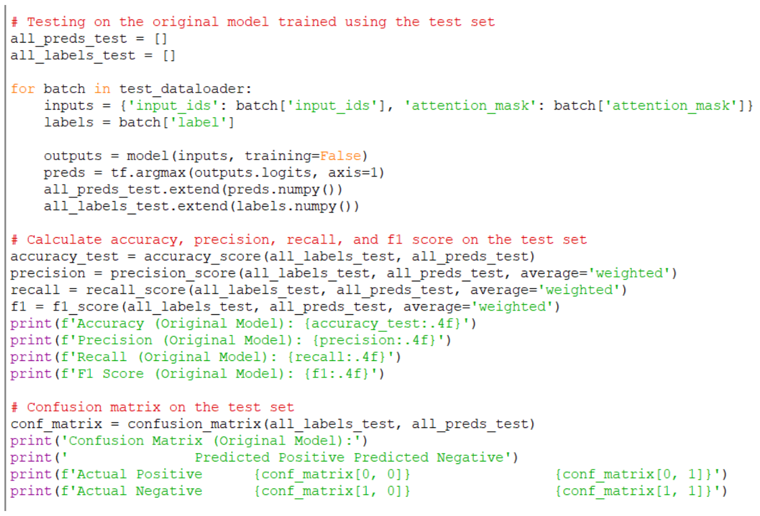


```
Epoch 4/4, Avg Train Loss: -685.2775, Train Accuracy: 0.9483
Epoch 4/4, Avg Validation Loss: -309.9155, Validation Accuracy: 0.8492
```

Figure 5-2-4-8 Evaluation of 4th Iteration of Modelling

5.2.5 Evaluation

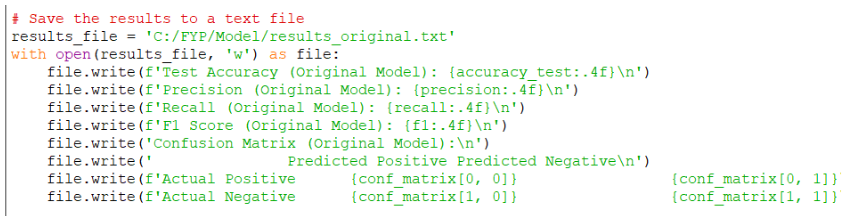
After modelling phase is ended. The model will be used for testing dataset for evaluation purposes. In this project, it is expected to achieve an average of 80% accuracy overall. Other than accuracy play as a role of evaluation, there are also precision, recall, F1-score, and confusion matrix for further evaluation. Figure 5-2-5-1 show the code of evaluation.



```
# Testing on the original model trained using the test set
all_preds_test = []
all_labels_test = []

for batch in test_dataloader:
    inputs = {'input_ids': batch['input_ids'], 'attention_mask': batch['attention_mask']}
    labels = batch['label']

    outputs = model(inputs, training=False)
    preds = tf.argmax(outputs.logits, axis=1)
    all_preds_test.extend(preds.numpy())
    all_labels_test.extend(labels.numpy())

# Calculate accuracy, precision, recall, and f1 score on the test set
accuracy_test = accuracy_score(all_labels_test, all_preds_test)
precision = precision_score(all_labels_test, all_preds_test, average='weighted')
recall = recall_score(all_labels_test, all_preds_test, average='weighted')
f1 = f1_score(all_labels_test, all_preds_test, average='weighted')
print(f'Accuracy (Original Model): {accuracy_test:.4f}')
print(f'Precision (Original Model): {precision:.4f}')
print(f'Recall (Original Model): {recall:.4f}')
print(f'F1 Score (Original Model): {f1:.4f}')

# Confusion matrix on the test set
conf_matrix = confusion_matrix(all_labels_test, all_preds_test)
print('Confusion Matrix (Original Model):')
print('                   Predicted Positive Predicted Negative')
print(f'Actual Positive      {conf_matrix[0, 0]}                 {conf_matrix[0, 1]}')
print(f'Actual Negative      {conf_matrix[1, 0]}                 {conf_matrix[1, 1]}')
```

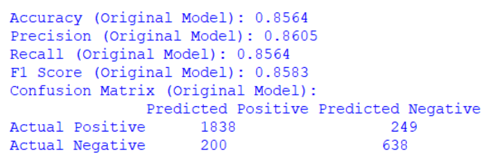
Figure 5-2-5-1 Code of Evaluation: Model Testing on Testing Dataset

After the Evaluation is completed, the result will save into a text file (Figure 5-2-5-2). The result printed by de code shown in Figure 5-2-5-3



```
# Save the results to a text file
results_file = 'C:/FYP/Model/results_original.txt'
with open(results_file, 'w') as file:
    file.write(f'Test Accuracy (Original Model): {accuracy_test:.4f}\n')
    file.write(f'Precision (Original Model): {precision:.4f}\n')
    file.write(f'Recall (Original Model): {recall:.4f}\n')
    file.write(f'F1 Score (Original Model): {f1:.4f}\n')
    file.write('Confusion Matrix (Original Model):\n')
    file.write('                   Predicted Positive Predicted Negative\n')
    file.write(f'Actual Positive      {conf_matrix[0, 0]}                 {conf_matrix[0, 1]}
    file.write(f'Actual Negative      {conf_matrix[1, 0]}                 {conf_matrix[1, 1]}
```

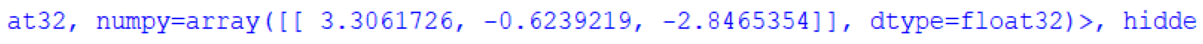
Figure 5-2-5-2 Code of Evaluation: Result of Evaluation Saving



```
Accuracy (Original Model): 0.8564
Precision (Original Model): 0.8605
Recall (Original Model): 0.8564
F1 Score (Original Model): 0.8583
Confusion Matrix (Original Model):
               Predicted Positive Predicted Negative
Actual Positive      1838                 249
Actual Negative      200                 638
```

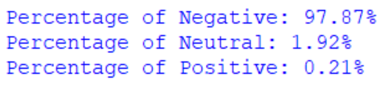
Figure 5-2-5-3 Result of Evaluation of The Model Towards Unseen Data

Next, Figure 5-2-5-4 show the result that the sentiment analysis can generate which is the percentage of particular review is belonging to negative, neutral, and positive. However, this result does not convert into a format that human understandable. Hence, to achieve that, the result needed to apply with softmax and avgmax function. And Figure 5-2-5-5 show the result of after applying softmax and avgmax function.



```
at32, numpy=array([[ 3.3061726, -0.6239219, -2.8465354]], dtype=float32)>, hidde
```

Figure 5-2-5-4 Result Before Applying Softmax and Avgmax Function



```
Percentage of Negative: 97.87%
Percentage of Neutral: 1.92%
Percentage of Positive: 0.21%
```

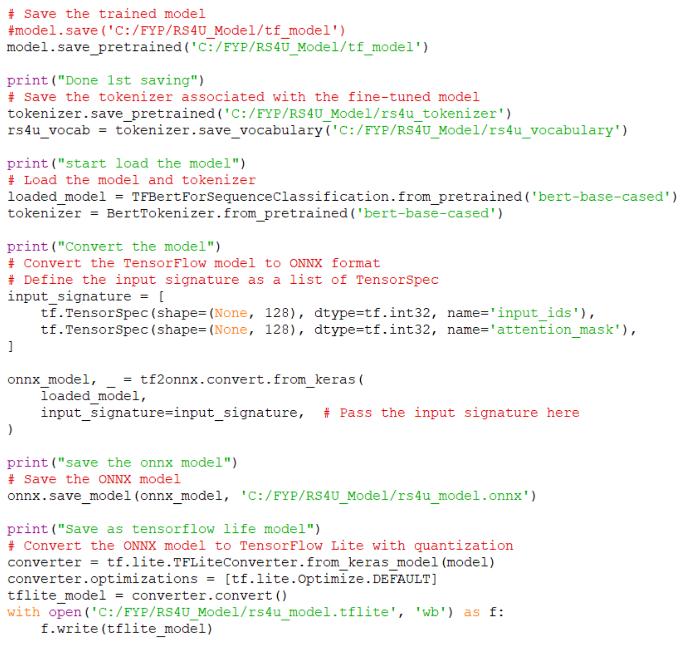
Figure 5-2-5-5 Result After Applying Softmax and Avgmax Function

5.2.6 Deployment: Model Saving and Application Development

5.2.6.1 Model Saving

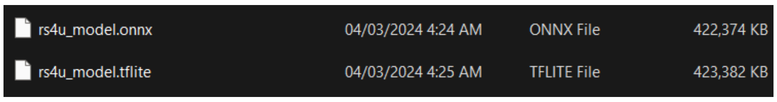
The project concist of front-end and back-end. For back-end, the most important element for the back-end is to apply the sentiment analysis model that trained into application created by using Android Studio. To achieve this, Figure 5-2-6-1-1 show the complete code on the model saving or known as model deployment. According to the planning in Chaper 4. The model is expected to save and integrated into the application with file type of TFLITE. Where before optimization is performed, the size of the model is about more than 400MB according ot Figure 5-2-6-1-2. However, there are size limitation for the TensorFlow Lite model from Android Studio (Figure 5-2-6-1-3). Where only a model less than 200MB is allowed to directly import. So, it is not able to direct integrate the model by its function. To ensure the model can be integrated into application in Android Studio, the model needed to be optimized. After optimization is performed the file size will be reduced but the performance of the model will slightly be affected. By saving the model in TenfsorFlow Lite with optimization, it enables the integration, or the deployment process more easy and lighter resources needed. Figure 5-2-6-1-4 show the model size after optimization which is about 100MB. Where Figure 5-2-6-1-1 show the process of model saving with optimization performed.

Firstly, the model will be saved using method named as “save_pretrained” which contain a file that have H5 file type. With saving the tokenizer and vocabulary file, it enables the developer to use it in Android Studio. To save the model, it is required to load the original pre-trained model and its tokenizer. Next, an input signature is important, as it affect the input collected from applications and used it in the model. Therefore, according to the model training, the input signature will be for input id and attention mask where both of them shall be in data type of integer and size within none to 128. To set this signature or specification, “TensorSpec” is used as the model is being trained by using TensorFlow library. Moving on, to save a model in TFLITE file type, it has to convert from pre-trained model with the customized input signature to ONNX file type. Then convert the model from ONNX file type to TFLITE file type. During converting process, optimization is performed to reduce the model file into a lighter version.



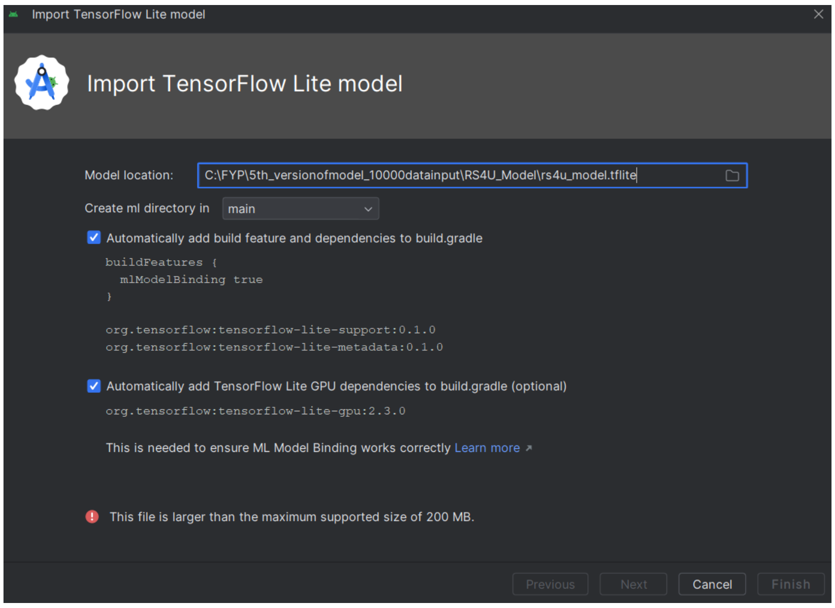
```
# Save the trained model
#model.save('C:/FYP/RS4U_Model/tf_model')
model.save_pretrained('C:/FYP/RS4U_Model/tf_model')

print("Done 1st saving")
# Save the tokenizer associated with the fine-tuned model
tokenizer.save_pretrained('C:/FYP/RS4U_Model/rs4u_tokenizer')
rs4u_vocab = tokenizer.save_vocabulary('C:/FYP/RS4U_Model/rs4u_vocabulary')

print("start load the model")
# Load the model and tokenizer
loaded_model = TFBertForSequenceClassification.from_pretrained('bert-base-cased')
tokenizer = BertTokenizer.from_pretrained('bert-base-cased')

print("Convert the model")
# Convert the TensorFlow model to ONNX format
# Define the input signature as a list of TensorSpec
input_signature = [
    tf.TensorSpec(shape=(None, 128), dtype=tf.int32, name='input_ids'),
    tf.TensorSpec(shape=(None, 128), dtype=tf.int32, name='attention_mask'),
]

onnx_model, _ = tf2onnx.convert.from_keras(
    loaded_model,
    input_signature=input_signature,  # Pass the input signature here
)

print("save the onnx model")
# Save the ONNX model
onnx.save_model(onnx_model, 'C:/FYP/RS4U_Model/rs4u_model.onnx')

print("Save as tensorflow life model")
# Convert the ONNX model to TensorFlow Lite with quantization
converter = tf.lite.TFLiteConverter.from_keras_model(model)
converter.optimizations = [tf.lite.Optimize.DEFAULT]
tflite_model = converter.convert()
with open('C:/FYP/RS4U_Model/rs4u_model.tflite', 'wb') as f:
    f.write(tflite_model)
```

Figure 5-2-6-1-1 Code of Deployment: Model Saving



```
rs4u_model.onnx      04/03/2024 4:24 AM    ONNX File      422,374 KB
rs4u_model.tflite    04/03/2024 4:25 AM    TFLITE File    423,382 KB
```

Figure 5-2-6-1-2 Size of Model Saved in TFLITE File Type


Import TensorFlow Lite model

Model location: C:\FYP\5th_versionofmodel_10000datainput\RS4U_Model\rs4u_model.tflite

Create ml directory in: main

☑ Automatically add build feature and dependencies to build.gradle

```
buildFeatures {
  mlModelBinding true
}

org.tensorflow:tensorflow-lite-support:0.1.0
org.tensorflow:tensorflow-lite-metadata:0.1.0
```

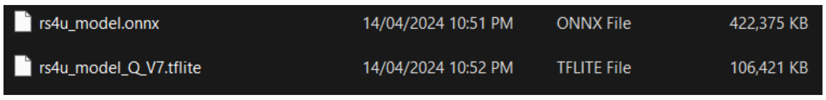
☑ Automatically add TensorFlow Lite GPU dependencies to build.gradle (optional)

```
org.tensorflow:tensorflow-lite-gpu:2.3.0
```

This is needed to ensure ML Model Binding works correctly Learn more

This file is larger than the maximum supported size of 200 MB.

Figure 5-2-6-1-3 Size Requirement of TensorFlow Model in Android Studio

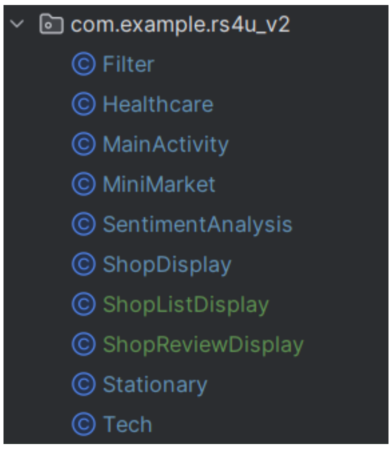


rs4u_model.onnx	14/04/2024 10:51 PM	ONNX File	422,375 KB
rs4u_model_Q_V7.tflite	14/04/2024 10:52 PM	TFLITE File	106,421 KB

Figure 5-2-6-1-4 Size of Model Saved in TFLITE File Type After Optimization

5.2.6.2 Application Development

The application development consists of 6 main parts which are main page, shop list display, shop display, shop review display, filter, and sentiment analysis. However, at first, all categories have its own activity named by its own category. However, to ensure the application performing well with less size, another activity is created which is Shop List Display.



```
com.example.rs4u_v2
    Filter
    Healthcare
    MainActivity
    MiniMarket
    SentimentAnalysis
    ShopDisplay
    ShopListDisplay
    ShopReviewDisplay
    Stationary
    Tech
```

Figure 5-2-6-2-1 Activities Created in Android Studio

5.2.6.2.1 Main Page

In the main page of the application, there are mainly two components displaying. Which are the categories of shop and list of shops recommended with criteria set. There is total four categories which are mini market, healthcare, stationary, and technology. In Figure 5-2-6-2-1-1, it shows the interface of the page developed. Which have some difference with the draft design of interface (Figure 4-3-1). In this page, the user is able to choose the category of shop for viewing the lists of shop. So, the user able to choose the shop to view its information or give review or rating. Other than this, the user is able to get recommendations of top shops that having outstanding result compared to most of other shops. For this shop recommended, the numbers is not limited. This means that, each categories of the shop in this page are having different number of shop get recommended.

To design the interface of this page, there is basic layout known as relative layout to contain other elements. For first part of displaying, there is a text view created in XML as a title coming with the boxes created in a grid layout with text view. After displaying the four categories of shop, there is a scroll layout that allow user to scroll when there are many shops is displayed. Within the scroll layout, there is a linear layout that contain four grid layouts. Purposes of having grid layout is to create text view to display the information of the shops. Figure 5-2-6-2-1-2 present a simplify version of the layout structure. However, due to lack of actual data in the application stored, and no shops are fulfilling the requirement set, hence, the figure did not show any shop is displayed.

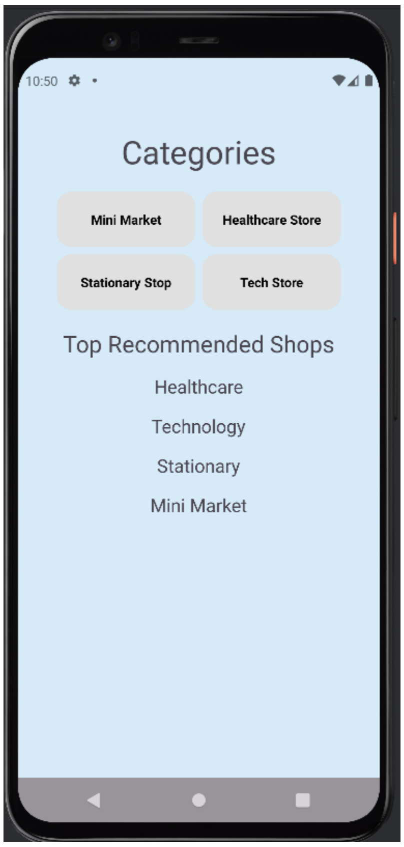
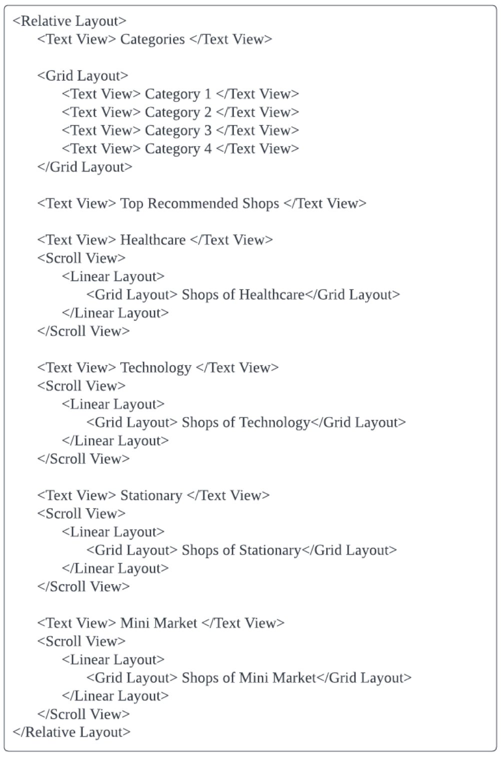


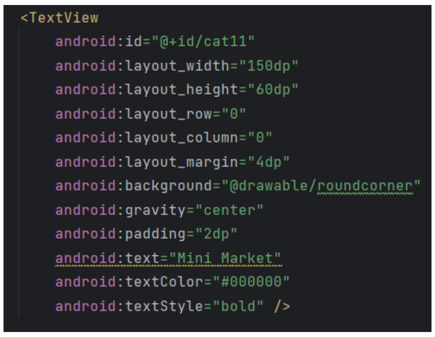
Figure 5-2-6-2-1-1 Application Interface: Main Activity



```
<Relative Layout>
    <Text View> Categories </Text View>

    <Grid Layout>
        <Text View> Category 1 </Text View>
        <Text View> Category 2 </Text View>
        <Text View> Category 3 </Text View>
        <Text View> Category 4 </Text View>
    </Grid Layout>

    <Text View> Top Recommended Shops </Text View>

    <Text View> Healthcare </Text View>
    <Scroll View>
        <Linear Layout>
            <Grid Layout> Shops of Healthcare</Grid Layout>
        </Linear Layout>
    </Scroll View>

    <Text View> Technology </Text View>
    <Scroll View>
        <Linear Layout>
            <Grid Layout> Shops of Technology</Grid Layout>
        </Linear Layout>
    </Scroll View>

    <Text View> Stationary </Text View>
    <Scroll View>
        <Linear Layout>
            <Grid Layout> Shops of Stationary</Grid Layout>
        </Linear Layout>
    </Scroll View>

    <Text View> Mini Market </Text View>
    <Scroll View>
        <Linear Layout>
            <Grid Layout> Shops of Mini Market</Grid Layout>
        </Linear Layout>
    </Scroll View>
</Relative Layout>
```

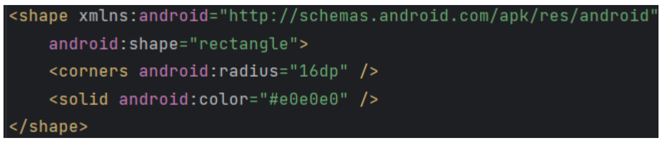
Figure 5-2-6-2-1-2 Simplified Version of Interface (XML) of Main Activity

Figure 5-2-6-2-1-3 show the example of design setting of the text view in the grid layout for the four categories. Where four of the text view are having size of width 150dp and height 60dp, margin of 4dp, padding of 2dp, text bolded and color with #000000 code which is black color, gravity set to center to ensure the text is place in center of the view, and a background setting that created in drawable (Figure 5-2-6-2-1-4).



```
<TextView
    android:id="@+id/cat11"
    android:layout_width="150dp"
    android:layout_height="60dp"
    android:layout_row="0"
    android:layout_column="0"
    android:layout_margin="4dp"
    android:background="@drawable/roundcorner"
    android:gravity="center"
    android:padding="2dp"
    android:text="Mini Market"
    android:textColor="#000000"
    android:textStyle="bold" />
```

Figure 5-2-6-2-1-3 Category Text View Design



```
<shape xmlns:android="http://schemas.android.com/apk/res/android"
    android:shape="rectangle">
    <corners android:radius="16dp" />
    <solid android:color="#e0e0e0" />
</shape>
```

Figure 5-2-6-2-1-4 Category Text View Background Design Setting

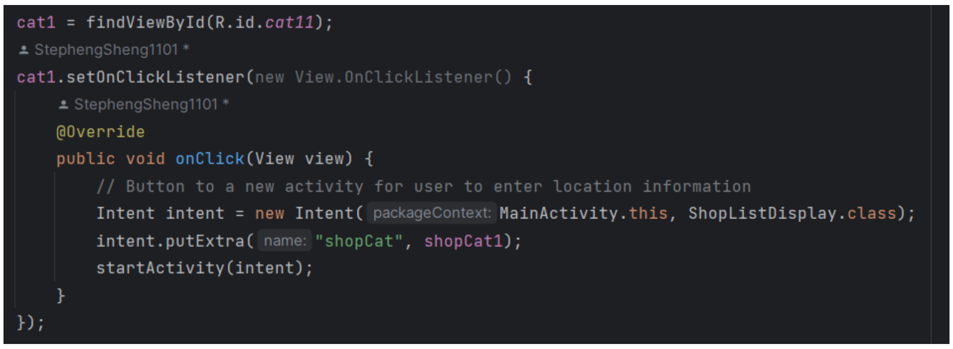
To access the element that created in XML from the activity code. Firstly, need to create variables with the variable type of text views. In Table 5-2-6-2-1-1, it shows all the variables that create initially. In this acvitiy, two groups of variables are created for different data type and purpose. To ensure the transition or interaction between two activities, such as from main activity to shop list display activity. The information passed between them are correct. For that, in the default method that create which is “onCreate”, the text view created in XML can be access and by writing code to ensure the functionality and performance of the text view. The text view is expected to be clickable by user, hence the page will go from main activity to shop list display activity.

Variable Name	**Variable Data Type**	**Purpose**
cat1	Text View	To linked with the elements created in XML from activity code.
cat2	Text View	
cat3	Text View	
cat4	Text View	

shopCat1	String with value "Mini Market"	To use as reference and pass to next activity then the category is chosen and clicked by user.
shopCat2	String with value "Healthcare"	
shopCat3	String with value "Stationary"	
shopCat4	String with value "Technology"	
gridLayout	Grid Layout	To linked with the layout created in XML and display the shops according to categories. Each grid layout is representing one category of shop.
gridLayout1	Grid Layout	
gridLayout11	Grid Layout	
gridLayout111	Grid Layout	
db	FirebaseFirestore	To access the Firebase Firestore and retrieve data saved.
shopList	ShopData	To save the shop that fulfilled recommendation criteria set. ShopData is a class create within the Main Activity.
totalRating	Float	To calculate the average rating of a shop
reviewCount	Float	
totalResults	Float	To calculate the percentage of a shop getting re positive review.
result2Count	Float	

Table 5-2-6-2-1-1 Main Activity Code: Variables Creation

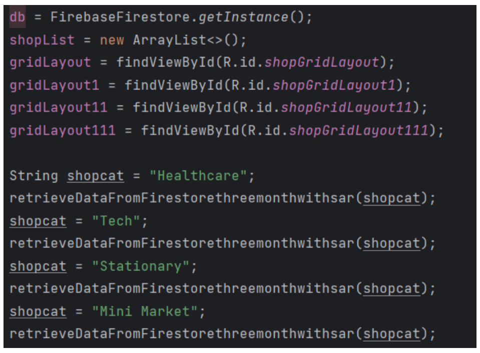
In Figure 5-2-6-2-1-5, it presents the example of code on accessing the element in XML and coding its functions. The function named as "findViewById" allow the variable created in the activity code which named as "cat1" linked with the element created in XML. For example, in Figure 5-2-6-2-1-3, the example of the text view is having an identification of "cat11". By setting the text view clickable, "setOnClickListener" is used. When the text view is clicked, it will pass the data which is named as "shopCat1" that assigned a value of "Mini Market" to next activity named as "Shop List Display" To achieve this, "Intent" is used for starting another activity when it is triggered and passing value.



```
cat1 = findViewById(R.id.cat11);
StephengSheng1101 *
cat1.setOnClickListener(new View.OnClickListener() {
    StephengSheng1101 *
    @Override
    public void onClick(View view) {
        // Button to a new activity for user to enter location information
        Intent intent = new Intent( packageContext: MainActivity.this, ShopListDisplay.class);
        intent.putExtra( name: "shopCat", shopCat1);
        startActivity(intent);
    }
});
```

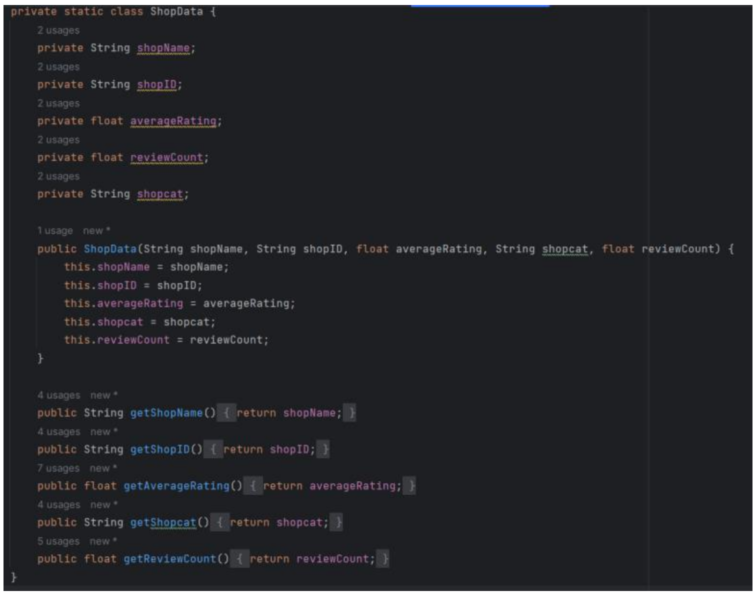
Figure 5-2-6-2-1-5 Main Activity Code: Onclick Listener of Four Category Text View

Next, based on the implementation planning stated in Chapter 4, the main page is expected to display the top recommended shops. Hence, there is a layout to display the shop that fulfilled the requirement and display as top recommended shops. In Figure 5-2-6-2-1-6, it shows the code of mainly four parts. Firstly, a variable in data type of" FirebaseFirestore" is created to access the database (Table 5-2-6-2-1-1). By using method named "getIntance" enable to access the Firebase. Next, an array list is created with variable named "shopList". It is created with using a class data type to store the shop basic information. Figure 5-2-6-2-1-7 show the code of the shop data class. Moving on, before displaying the shop. Need to ensure the layout is created and linked. In Figure 5-2-6-2-1-6, it show four grid layout that created in XML is assigned and linked with the variable created in main activity code (Table 5-2-6-2-1-1). Lastly, creating four string variables with different categories. It will called the function named as "retrieveDataFromFirestorethreemonthwithsar" with passing a variable of shop categories. To display the shops, the shops are required to fulfilled the requirements set, which are having average rating more than 3, and more than or equal to 86% percentage of a shop getting positive review.



```
db = FirebaseFirestore.getInstance();
shopList = new ArrayList<>();
gridLayout = findViewById(R.id.shopGridLayout);
gridLayout1 = findViewById(R.id.shopGridLayout1);
gridLayout11 = findViewById(R.id.shopGridLayout11);
gridLayout111 = findViewById(R.id.shopGridLayout111);

String shopcat = "Healthcare";
retrieveDataFromFirestorethreemonthwithsar(shopcat);
shopcat = "Tech";
retrieveDataFromFirestorethreemonthwithsar(shopcat);
shopcat = "Stationary";
retrieveDataFromFirestorethreemonthwithsar(shopcat);
shopcat = "Mini Market";
retrieveDataFromFirestorethreemonthwithsar(shopcat);
```

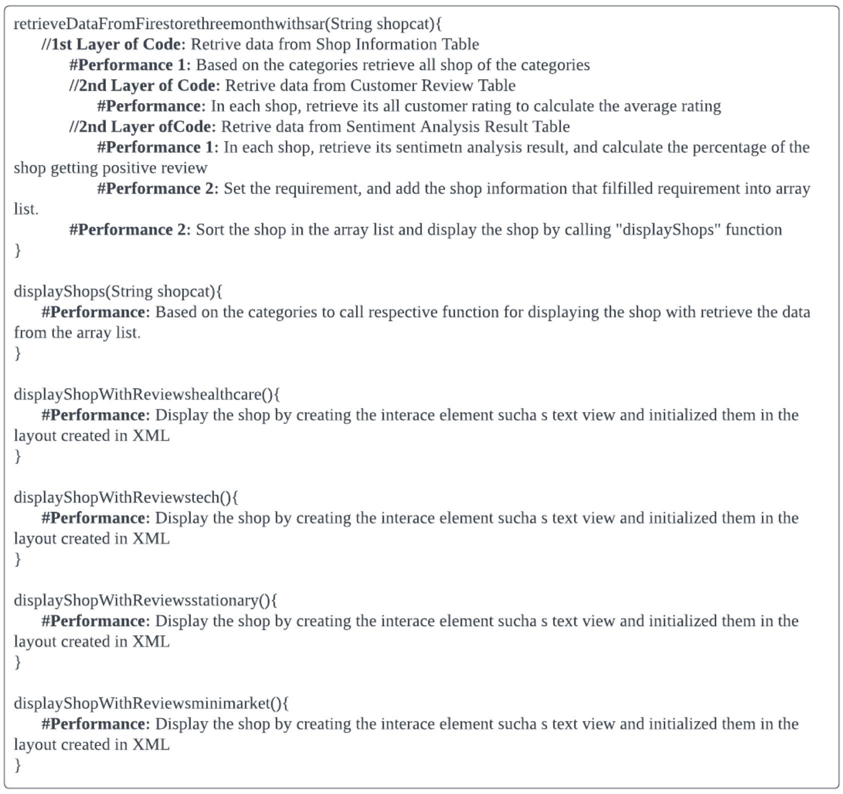
Figure 5-2-6-2-1-6 Main Activity Code: Shop Recommended Displaying



```
private static class ShopData {
    2 usages
    private String shopName;
    2 usages
    private String shopID;
    2 usages
    private float averageRating;
    2 usages
    private float reviewCount;
    2 usages
    private String shopcat;

    1 usage  new *
    public ShopData(String shopName, String shopID, float averageRating, String shopcat, float reviewCount) {
        this.shopName = shopName;
        this.shopID = shopID;
        this.averageRating = averageRating;
        this.shopcat = shopcat;
        this.reviewCount = reviewCount;
    }

    4 usages  new *
    public String getShopName() { return shopName; }
    4 usages  new *
    public String getShopID() { return shopID; }
    7 usages  new *
    public float getAverageRating() { return averageRating; }
    4 usages  new *
    public String getShopcat() { return shopcat; }
    5 usages  new *
    public float getReviewCount() { return reviewCount; }
}
```

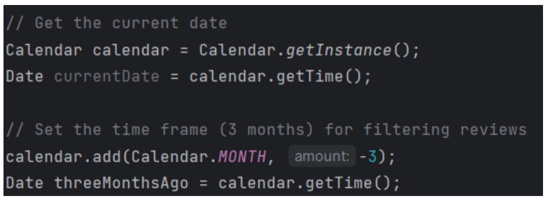
Figure 5-2-6-2-1-7 Main Activity Code: Shop Data Class

Figure 5-2-6-2-1-8 show the simplified code explanation in a diagram. The code implementation logic of retrieving data from Firebase and display into interface are similar to every page including Shop List Display Activity, Shop Display Activity, and Shop Review Display Activity in the sub chapter of Chapter 5.2.6.2.2 to Chapter 5.2.6.2.4. In the retrieving data function in Main Activity. It is design for retrieve data only within current three month. Hence, first element in the “retrieveDataFromFirestorethreemonthwithsar” function it so retrieves the current date and calculate the data of three month ago (Figure 5-2-6-2-1-9). The code of retrieving data from firebase is divided in to Figure 5-2-6-2-1-11 to Figure 5-2-6-2-1-15 with details explanation.



```
retrieveDataFromFirestorethreemonthwithsar(String shopcat){
    //1st Layer of Code: Retrive data from Shop Information Table
        #Performance 1: Based on the categories retrieve all shop of the categories
        //2nd Layer of Code: Retrive data from Customer Review Table
            #Performance: In each shop, retrieve its all customer rating to calculate the average rating
        //2nd Layer ofCode: Retrive data from Sentiment Analysis Result Table
            #Performance 1: In each shop, retrieve its sentimetn analysis result, and calculate the percentage of the
shop getting positive review
            #Performance 2: Set the requirement, and add the shop information that filfilled requirement into array
list.
        #Performance 2: Sort the shop in the array list and display the shop by calling "displayShops" function
}

displayShops(String shopcat){
    #Performance: Based on the categories to call respective function for displaying the shop with retrieve the data
from the array list.
}

displayShopWithReviewshealthcare(){
    #Performance: Display the shop by creating the interace element sucha s text view and initialized them in the
layout created in XML
}

displayShopWithReviewstech(){
    #Performance: Display the shop by creating the interace element sucha s text view and initialized them in the
layout created in XML
}

displayShopWithReviewsstationary(){
    #Performance: Display the shop by creating the interace element sucha s text view and initialized them in the
layout created in XML
}

displayShopWithReviewsminimarket(){
    #Performance: Display the shop by creating the interace element sucha s text view and initialized them in the
layout created in XML
}
```

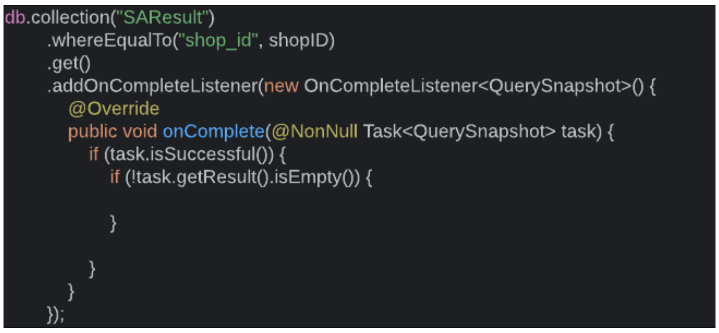
Figure 5-2-6-2-1-8 Main Activity Code: Simplified Code in Diagram of Retrieve Data From Firebase Function



```
// Get the current date
Calendar calendar = Calendar.getInstance();
Date currentDate = calendar.getTime();

// Set the time frame (3 months) for filtering reviews
calendar.add(Calendar.MONTH,  amount: -3);
Date threeMonthsAgo = calendar.getTime();
```

Figure 5-2-6-2-1-9 Main Activity Code: Retrieve Data From Firebase Function part 1

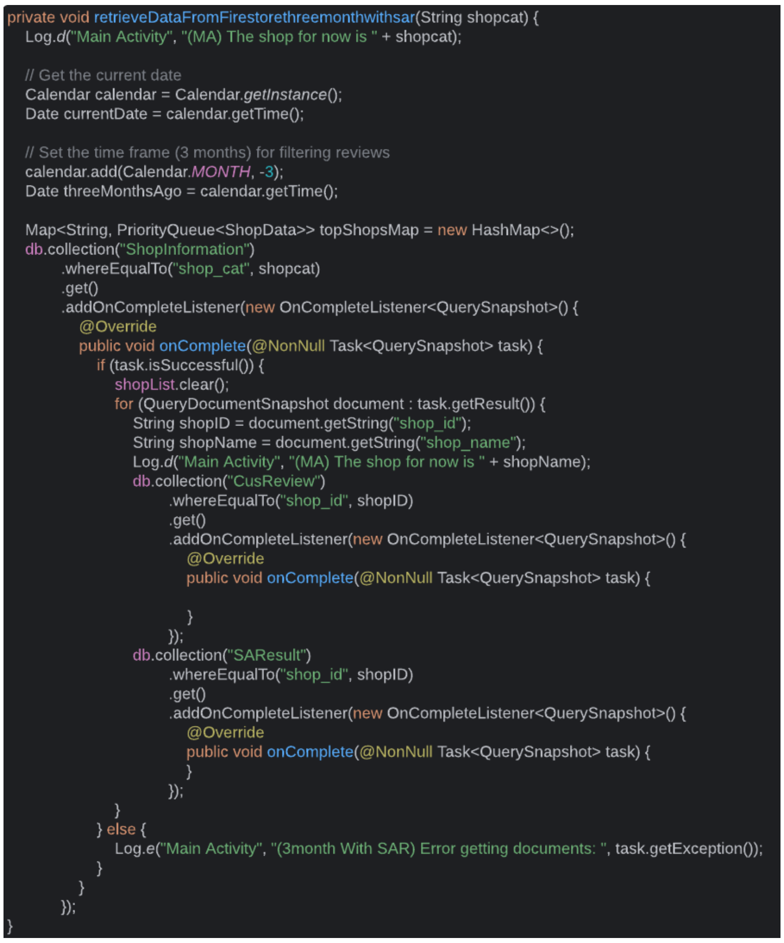
Figure 5-2-6-2-1-10 show the basic structure of code for retrieving data from Firebase Firestore. In this figure, it retrieves data from a table named "SAResult" and set a requirement that shop ID must matched. Next, to perform further functionalities such as calculation or retrieving data. These performances are placed with in the function named "onComplete"



```
db.collection("SAResult")
      .whereEqualTo("shop_id", shopID)
      .get()
      .addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
         @Override
         public void onComplete(@NonNull Task<QuerySnapshot> task) {
            if (task.isSuccessful()) {
               if (!task.getResult().isEmpty()) {

               }

            }
         }
      });
```

Figure 5-2-6-2-1-10 Code Structure of Retrieving Data from Firebase

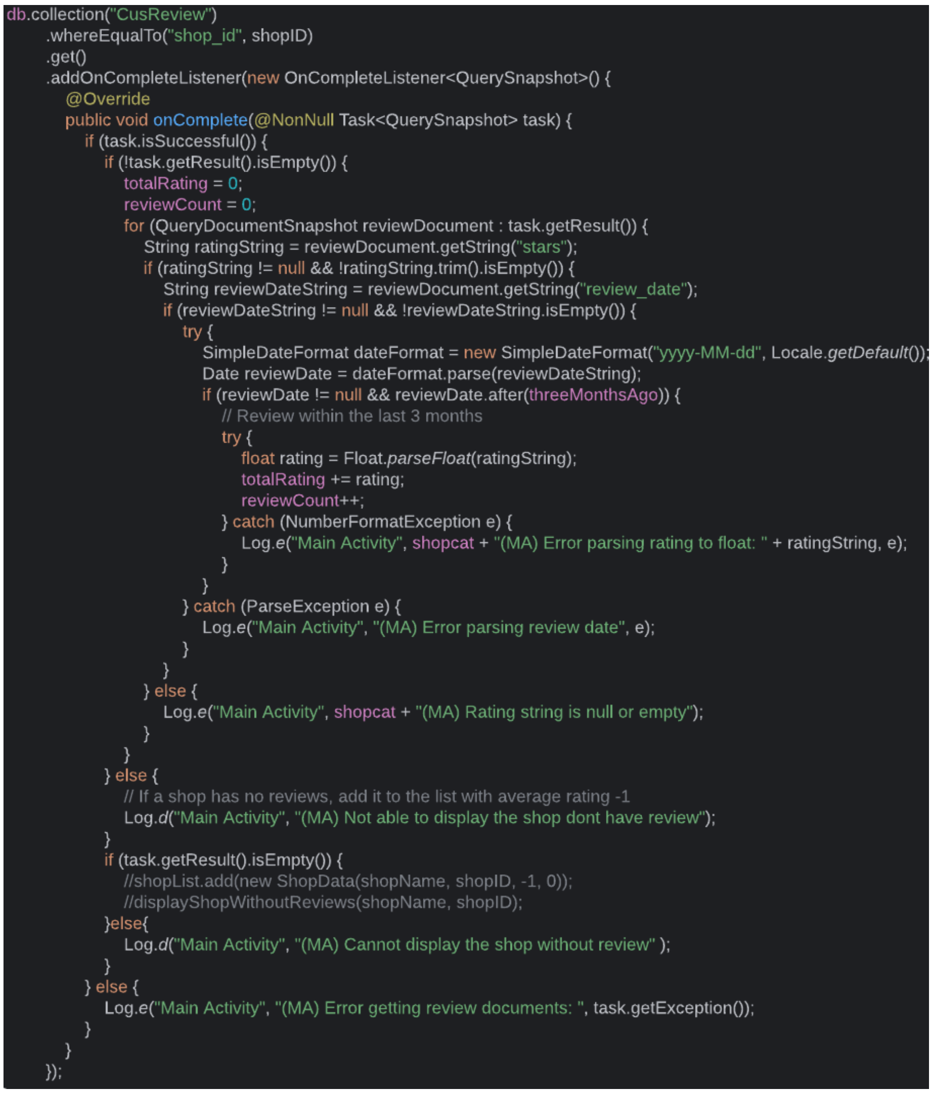
To display the shop at interface, the categories need to be confirmed when user chosen it. Then, it starts retrieving data from the list of shop from the respective table named “ShopInformation”. After a shop is confirmed, it will use the unique attribute which is the shop ID to perform further data retrieving from other table such as “SAResult” and “CusReview” to perform further calculation for each shop including calculation of average rating and percentage of a shop getting positive review. Which show in Figure 5-2-6-2-1-11. Figure 5-2-6-2-1-12 and Figure 5-2-6-2-1-13 show the code of retrieving data from Customer Review Table and Sentiment Analysis Result Table which these two codes are part of the code shown in Figure 5-2-6-2-1-11.



```java
private void retrieveDataFromFirestorethreemonthwithsar(String shopcat) {
    Log.d("Main Activity", "(MA) The shop for now is " + shopcat);

    // Get the current date
    Calendar calendar = Calendar.getInstance();
    Date currentDate = calendar.getTime();

    // Set the time frame (3 months) for filtering reviews
    calendar.add(Calendar.MONTH, -3);
    Date threeMonthsAgo = calendar.getTime();

    Map<String, PriorityQueue<ShopData>> topShopsMap = new HashMap<>();
    db.collection("ShopInformation")
            .whereEqualTo("shop_cat", shopcat)
            .get()
            .addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
                @Override
                public void onComplete(@NonNull Task<QuerySnapshot> task) {
                    if (task.isSuccessful()) {
                        shopList.clear();
                        for (QueryDocumentSnapshot document : task.getResult()) {
                            String shopID = document.getString("shop_id");
                            String shopName = document.getString("shop_name");
                            Log.d("Main Activity", "(MA) The shop for now is " + shopName);
                            db.collection("CusReview")
                                    .whereEqualTo("shop_id", shopID)
                                    .get()
                                    .addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
                                        @Override
                                        public void onComplete(@NonNull Task<QuerySnapshot> task) {

                                        }
                                    });
                            db.collection("SAResult")
                                    .whereEqualTo("shop_id", shopID)
                                    .get()
                                    .addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
                                        @Override
                                        public void onComplete(@NonNull Task<QuerySnapshot> task) {
                                        }
                                    });
                        }
                    } else {
                        Log.e("Main Activity", "(3month With SAR) Error getting documents: ", task.getException());
                    }
                }
            });
}
```

Figure 5-2-6-2-1-11 Main Activity Code: Retrieve Data From Firebase Function part 2_ShopInformation Table

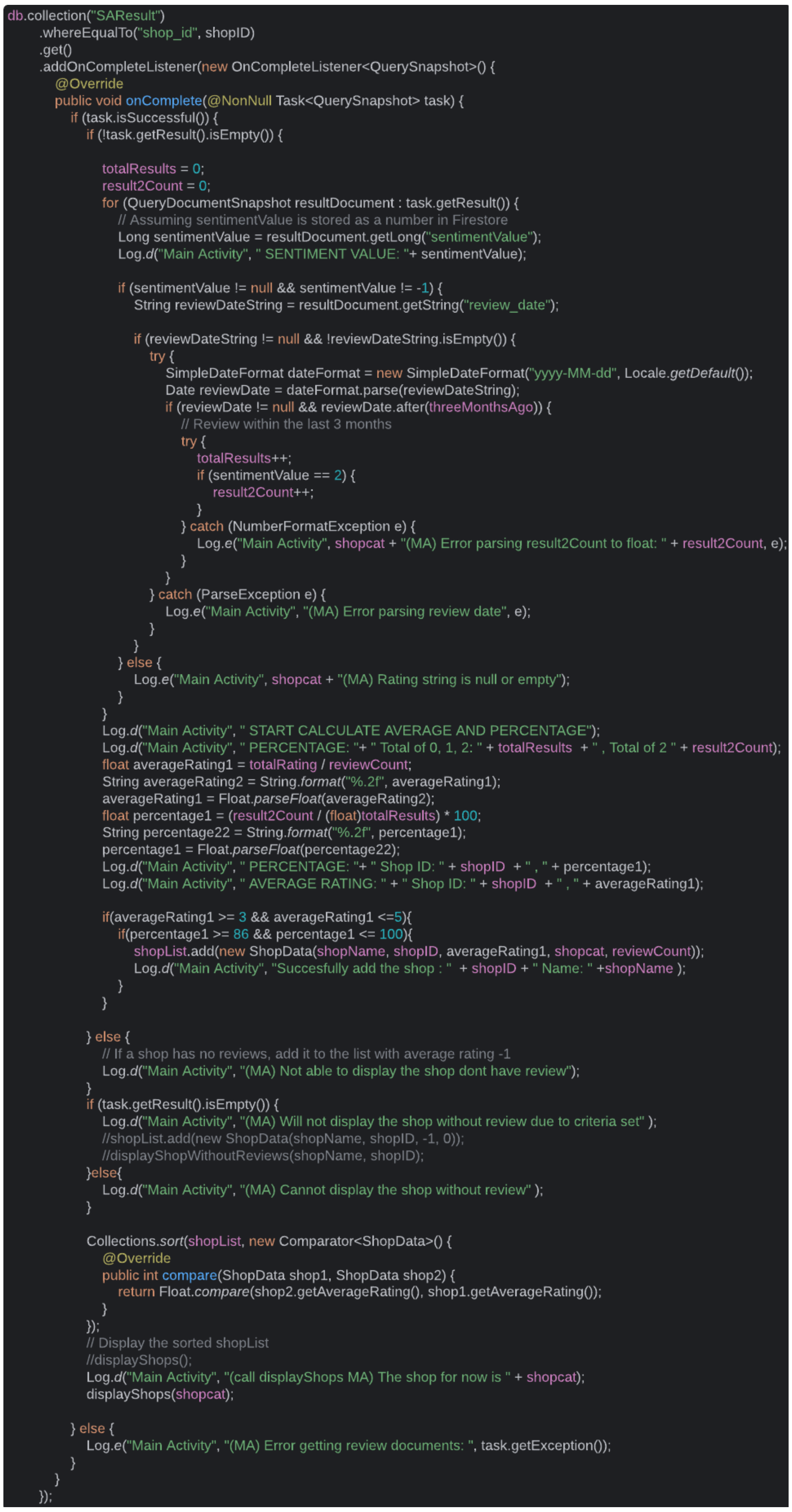
Figure 5-2-6-2-1-12 is part for the code inside the function of retrieving data from Firebase. In this figure, it mainly shows the code of retrieving from customer review table. Two main variables are created for retrieve data on customer review table (Table 5-2-6-2-1-1). Which are total rating and review count. For example, there are 4 rating of 4 and 3 rating of 4. Hence total rating will be 28 and review count is 7. When a shop is confirmed and prepare to retrieve data from customer review table. There is special case, where the shop many do not have any review and rating. To ensure the retrieving process successfully, an IF loop is used which is if the shop is not having empty result in the table, then it will start retrieve the rating and store in a variable created within the method. As the requirements set, the data retrieving only include the current three month. Therefore, it will retrieve the rating date to ensure it is within the three months.



```
db.collection("CusReview")
    .whereEqualTo("shop_id", shopID)
    .get()
    .addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
      @Override
      public void onComplete(@NonNull Task<QuerySnapshot> task) {
        if (task.isSuccessful()) {
          if (!task.getResult().isEmpty()) {
            totalRating = 0;
            reviewCount = 0;
            for (QueryDocumentSnapshot reviewDocument : task.getResult()) {
              String ratingString = reviewDocument.getString("stars");
              if (ratingString != null && !ratingString.trim().isEmpty()) {
                String reviewDateString = reviewDocument.getString("review_date");
                if (reviewDateString != null && !reviewDateString.isEmpty()) {
                  try {
                    SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd", Locale.getDefault());
                    Date reviewDate = dateFormat.parse(reviewDateString);
                    if (reviewDate != null && reviewDate.after(threeMonthsAgo)) {
                      // Review within the last 3 months
                      try {
                        float rating = Float.parseFloat(ratingString);
                        totalRating += rating;
                        reviewCount++;
                      } catch (NumberFormatException e) {
                        Log.e("Main Activity", shopcat + "(MA) Error parsing rating to float: " + ratingString, e);
                      }
                    }
                  } catch (ParseException e) {
                    Log.e("Main Activity", "(MA) Error parsing review date", e);
                  }
                }
              } else {
                Log.e("Main Activity", shopcat + "(MA) Rating string is null or empty");
              }
            }
          } else {
            // If a shop has no reviews, add it to the list with average rating -1
            Log.d("Main Activity", "(MA) Not able to display the shop dont have review");
          }
          if (task.getResult().isEmpty()) {
            //shopList.add(new ShopData(shopName, shopID, -1, 0));
            //displayShopWithoutReviews(shopName, shopID);
          }else{
            Log.d("Main Activity", "(MA) Cannot display the shop without review" );
          }
        } else {
          Log.e("Main Activity", "(MA) Error getting review documents: ", task.getException());
        }
      }
    });
```

Figure 5-2-6-2-1-12 Main Activity Code: Retrieve Data From Firebase Function part 3_Customer Review Table

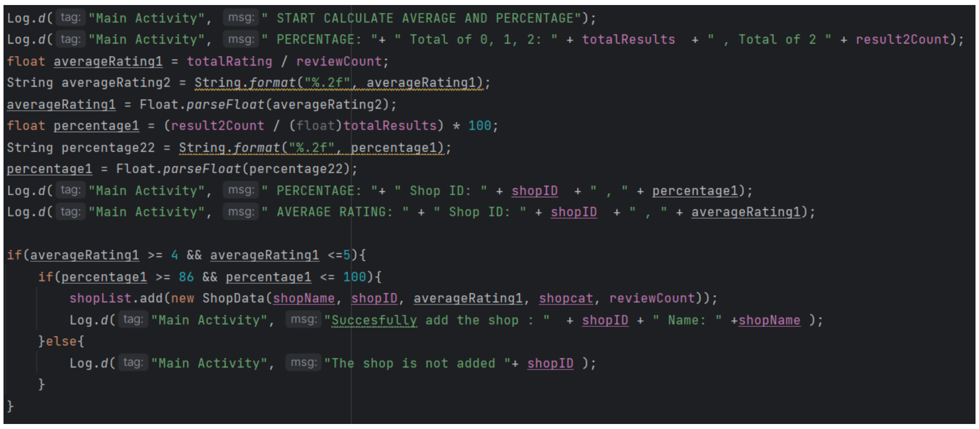
Figure 5-2-6-2-1-13 show another part of the code for the data retrieving function. It is to calculate the percentage of a shop getting positive review and according to requirement and criteria set to add the shop into a list and displaying them accordingly. According to Table 5-2-6-2-1-1, two variables is created for storing the value of total result of sentiment analysis, and total result of having value 2. The sentiment analysis model will only generate four result which are -1, 0, 1, and 2. Negative one for error or special case, 0 for negative, 1 for neutral, and positive is 2. Total result will exclude the result have having value of negative 1. Some of the shop may be new shop which may not having any sentiment analysis result yet as they did not receive any rating and review. Therefore, there is an IF loop for ensuring only the shop that not having empty data in Sentiment Analysis Result Table will continue the calculation and further performance. Next, the requirement stated that the data within three months will be retrieved. Hence, the date that store in the table will be retrieved to examine as well. Lastly, calculate the number of total result and total result of value 2 and store into the variables.



```
db.collection("SAResult")
        .whereEqualTo("shop_id", shopID)
        .get()
        .addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
            @Override
            public void onComplete(@NonNull Task<QuerySnapshot> task) {
                if (task.isSuccessful()) {
                    if (!task.getResult().isEmpty()) {

                        totalResults = 0;
                        result2Count = 0;
                        for (QueryDocumentSnapshot resultDocument : task.getResult()) {
                            // Assuming sentimentValue is stored as a number in Firestore
                            Long sentimentValue = resultDocument.getLong("sentimentValue");
                            Log.d("Main Activity", " SENTIMENT VALUE: "+ sentimentValue);

                            if (sentimentValue != null && sentimentValue != -1) {
                                String reviewDateString = resultDocument.getString("review_date");

                                if (reviewDateString != null && !reviewDateString.isEmpty()) {
                                    try {
                                        SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd", Locale.getDefault());
                                        Date reviewDate = dateFormat.parse(reviewDateString);
                                        if (reviewDate != null && reviewDate.after(threeMonthsAgo)) {
                                            // Review within the last 3 months
                                            try {
                                                totalResults++;
                                                if (sentimentValue == 2) {
                                                    result2Count++;
                                                }
                                            } catch (NumberFormatException e) {
                                                Log.e("Main Activity", shopcat + "(MA) Error parsing result2Count to float: " + result2Count, e);
                                            }
                                        }
                                    } catch (ParseException e) {
                                        Log.e("Main Activity", "(MA) Error parsing review date", e);
                                    }
                                }
                            } else {
                                Log.e("Main Activity", shopcat + "(MA) Rating string is null or empty");
                            }
                        }
                        Log.d("Main Activity", " START CALCULATE AVERAGE AND PERCENTAGE");
                        Log.d("Main Activity", " PERCENTAGE: "+ " Total of 0, 1, 2: " + totalResults  + " , Total of 2 " + result2Count);
                        float averageRating1 = totalRating / reviewCount;
                        String averageRating2 = String.format("%.2f", averageRating1);
                        averageRating1 = Float.parseFloat(averageRating2);
                        float percentage1 = (result2Count / (float)totalResults) * 100;
                        String percentage22 = String.format("%.2f", percentage1);
                        percentage1 = Float.parseFloat(percentage22);
                        Log.d("Main Activity", " PERCENTAGE: "+ " Shop ID: " + shopID  + " , " + percentage1);
                        Log.d("Main Activity", " AVERAGE RATING: " + " Shop ID: " + shopID  + " , " + averageRating1);

                        if(averageRating1 >= 3 && averageRating1 <=5){
                            if(percentage1 >= 86 && percentage1 <= 100){
                                shopList.add(new ShopData(shopName, shopID, averageRating1, shopcat, reviewCount));
                                Log.d("Main Activity", "Succesfully add the shop : "  + shopID + " Name: " +shopName );
                            }
                        }

                    } else {
                        // If a shop has no reviews, add it to the list with average rating -1
                        Log.d("Main Activity", "(MA) Not able to display the shop dont have review");
                    }
                    if (task.getResult().isEmpty()) {
                        Log.d("Main Activity", "(MA) Will not display the shop without review due to criteria set" );
                        //shopList.add(new ShopData(shopName, shopID, -1, 0));
                        //displayShopWithoutReviews(shopName, shopID);
                    }else{
                        Log.d("Main Activity", "(MA) Cannot display the shop without review" );
                    }

                    Collections.sort(shopList, new Comparator<ShopData>() {
                        @Override
                        public int compare(ShopData shop1, ShopData shop2) {
                            return Float.compare(shop2.getAverageRating(), shop1.getAverageRating());
                        }
                    });
                    // Display the sorted shopList
                    //displayShops();
                    Log.d("Main Activity", "(call displayShops MA) The shop for now is " + shopcat);
                    displayShops(shopcat);

                } else {
                    Log.e("Main Activity", "(MA) Error getting review documents: ", task.getException());
                }
            }
        });
```

Figure 5-2-6-2-1-13 Main Activity Code: Retrieve Data From Firebase Function part 4_Sentiment Analysis Result Table

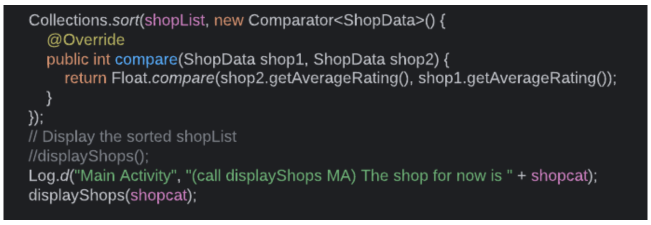
After all calculation is completed. Based on Figure 5-2-6-2-1-14, final calculation will be performed. Which is using the total rating and review count to calculate the average rating. As the example given previously, total rating is 28, and review count is 7. Hence, to get average rating the toral rating will be divided by review count, and the result will be average rating of 4. Considering, the result may have many decimal points. There is a conversion to ensure the value only have 2 decimal points. Next, to calculate the percentage of a shop getting positive review, it is using total result of value 2 divided by total results and multiply my 100 to get percentage. And convert them into a value that contain only two decimal points. Lastly, there is multi-layers IF loop play an important role on examine and determine fulfilment of the criteria and requirement on shop displaying. Only a shop will add into an array object list when it fulfilled the requirement and criteria by having average rating more than or equal to 4 and percentage of a shop getting positive review that more than or equal to 86 percent.



```
Log.d( tag: "Main Activity",  msg: " START CALCULATE AVERAGE AND PERCENTAGE");
Log.d( tag: "Main Activity",  msg: " PERCENTAGE: "+ " Total of 0, 1, 2: " + totalResults  + " , Total of 2 " + result2Count);
float averageRating1 = totalRating / reviewCount;
String averageRating2 = String.format("%.2f", averageRating1);
averageRating1 = Float.parseFloat(averageRating2);
float percentage1 = (result2Count / (float)totalResults) * 100;
String percentage22 = String.format("%.2f", percentage1);
percentage1 = Float.parseFloat(percentage22);
Log.d( tag: "Main Activity",  msg: " PERCENTAGE: "+ " Shop ID: " + shopID  + " , " + percentage1);
Log.d( tag: "Main Activity",  msg: " AVERAGE RATING: " + " Shop ID: " + shopID  + " , " + averageRating1);

if(averageRating1 >= 4 && averageRating1 <=5){
    if(percentage1 >= 86 && percentage1 <= 100){
        shopList.add(new ShopData(shopName, shopID, averageRating1, shopcat, reviewCount));
        Log.d( tag: "Main Activity",  msg: "Succesfully add the shop : "  + shopID + " Name: " +shopName );
    }else{
        Log.d( tag: "Main Activity",  msg: "The shop is not added "+ shopID );
    }
}
```

Figure 5-2-6-2-1-14 Main Activity Code: Retrieve Data From Firebase Function part 5_Average Rating and Percentage Calculation

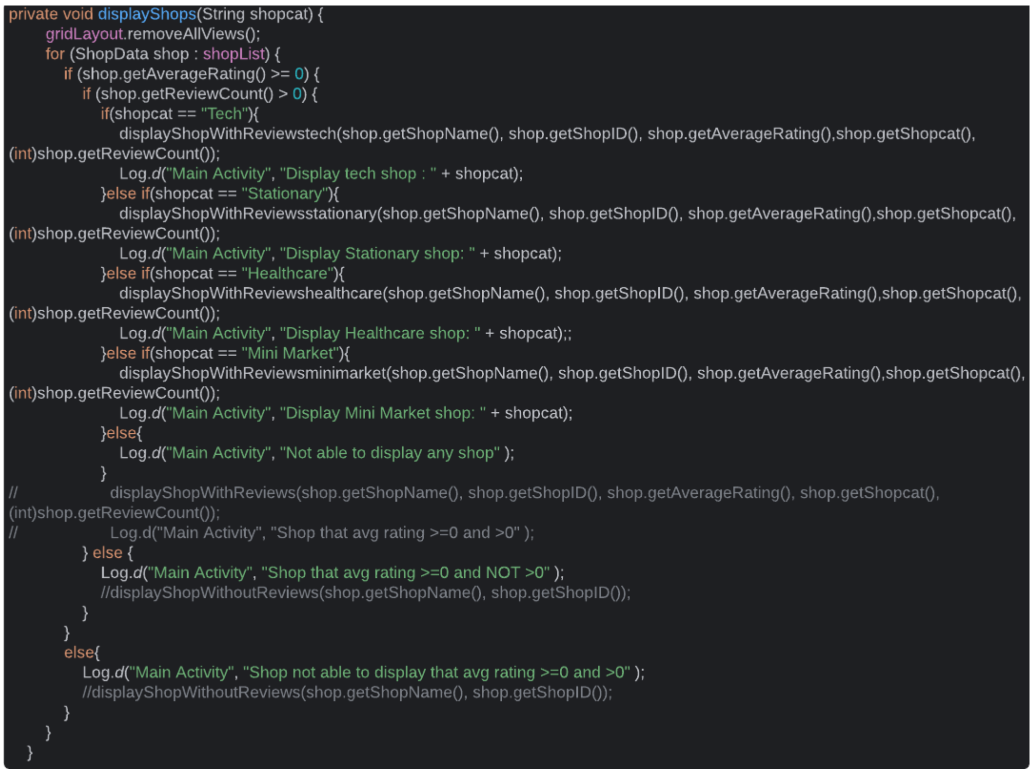
After all shops that fulfilled the requirement and added into array object list. A sorting function will be called to sort the shop based on average rating. Lastly, call the display shop function to display the shop on the layout with the shop categories value passed to next function (Figure 5-2-6-2-1-15).



```
Collections.sort(shopList, new Comparator<ShopData>() {
    @Override
    public int compare(ShopData shop1, ShopData shop2) {
        return Float.compare(shop2.getAverageRating(), shop1.getAverageRating());
    }
});
// Display the sorted shopList
//displayShops();
Log.d("Main Activity", "(call displayShops MA) The shop for now is " + shopcat);
displayShops(shopcat);
```

Figure 5-2-6-2-1-15 Main Activity Code: Retrieve Data From Firebase Function part 6_Shop Sorting and Display

Due to the data retrieve function will be called 4 times for each category to display together in a same layout that a fundamental layout created in XML. The fundamental layout is a linear layout. Next, the code implementation logic of displaying element at interface is similar for each category. However, there is one main difference is the layout that the elements created add into. This is because, each category has its own respective grid layout created inside the fundamental layout. Figure 5-2-6-2-1-16 show a function that ensuring correct shop categories displaying function will be called. To achieve that, there is a multiple layer IF loop, where only the average rating more than or equal to 0 and number of reviews more than 0, lastly according to the shop categories, different function will be called. However, the function is similar logic. When calling the function, it will retrieve the information and passing them into the next function accordingly.

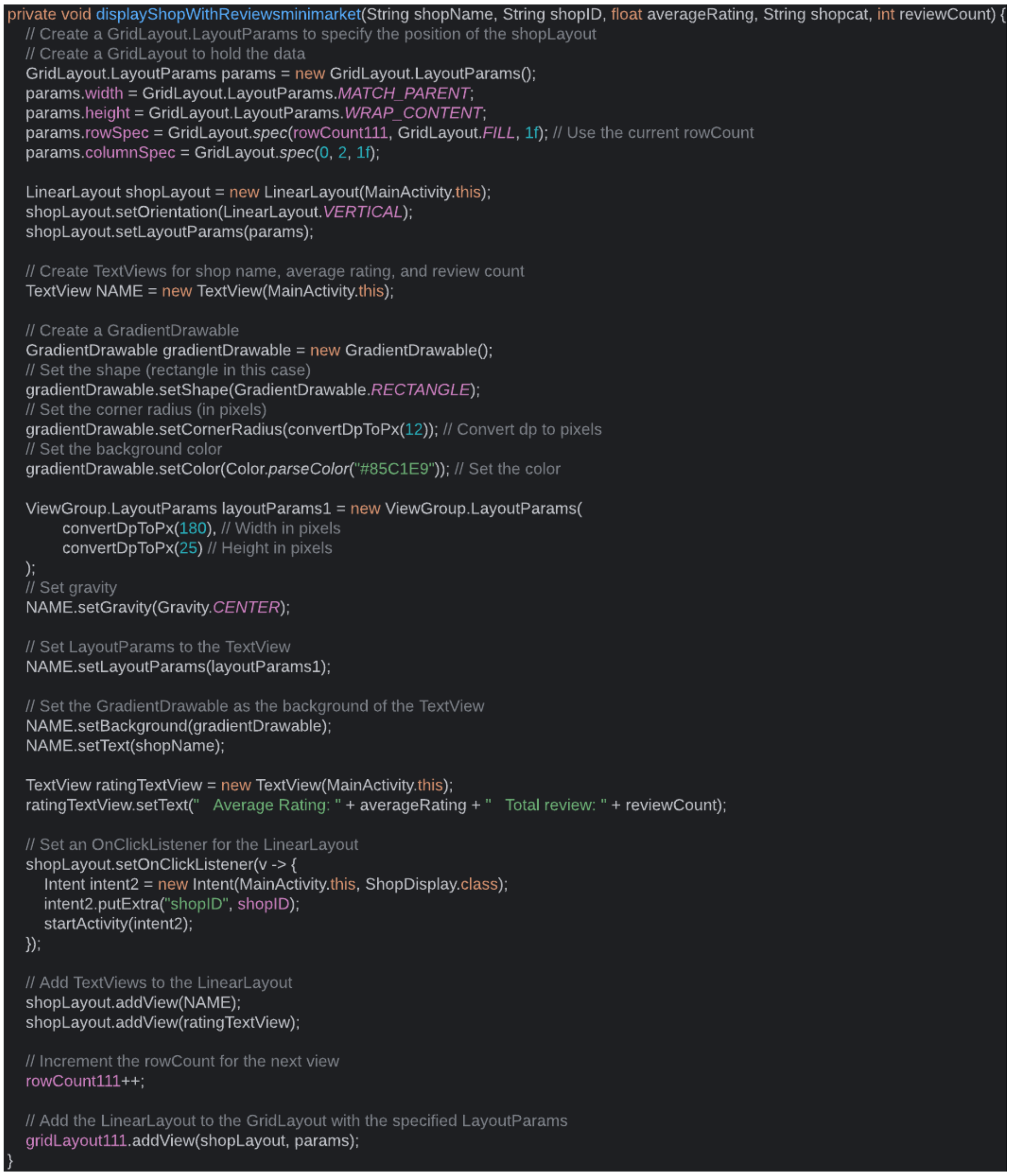


```
private void displayShops(String shopcat) {
    gridLayout.removeAllViews();
    for (ShopData shop : shopList) {
        if (shop.getAverageRating() >= 0) {
            if (shop.getReviewCount() > 0) {
                if(shopcat == "Tech"){
                    displayShopWithReviewstech(shop.getShopName(), shop.getShopID(), shop.getAverageRating(),shop.getShopcat(),
(int)shop.getReviewCount());
                    Log.d("Main Activity", "Display tech shop : " + shopcat);
                }else if(shopcat == "Stationary"){
                    displayShopWithReviewsstationary(shop.getShopName(), shop.getShopID(), shop.getAverageRating(),shop.getShopcat(),
(int)shop.getReviewCount());
                    Log.d("Main Activity", "Display Stationary shop: " + shopcat);
                }else if(shopcat == "Healthcare"){
                    displayShopWithReviewshealthcare(shop.getShopName(), shop.getShopID(), shop.getAverageRating(),shop.getShopcat(),
(int)shop.getReviewCount());
                    Log.d("Main Activity", "Display Healthcare shop: " + shopcat);;
                }else if(shopcat == "Mini Market"){
                    displayShopWithReviewsminimarket(shop.getShopName(), shop.getShopID(), shop.getAverageRating(),shop.getShopcat(),
(int)shop.getReviewCount());
                    Log.d("Main Activity", "Display Mini Market shop: " + shopcat);
                }else{
                    Log.d("Main Activity", "Not able to display any shop" );
                }
//                  displayShopWithReviews(shop.getShopName(), shop.getShopID(), shop.getAverageRating(), shop.getShopcat(),
(int)shop.getReviewCount());
//                  Log.d("Main Activity", "Shop that avg rating >=0 and >0" );
            } else {
                Log.d("Main Activity", "Shop that avg rating >=0 and NOT >0" );
                //displayShopWithoutReviews(shop.getShopName(), shop.getShopID());
            }
        }
        else{
            Log.d("Main Activity", "Shop not able to display that avg rating >=0 and >0" );
            //displayShopWithoutReviews(shop.getShopName(), shop.getShopID());
        }
    }
}
```

Figure 5-2-6-2-1-16 Main Activity Code: Shop Display Function

There will be four similar functions, but each function represents one category of the shop to display the element into own respective layout created in the fundamental layout. In Figure 5-2-6-2-1-17, it shows the complete code of displaying shop information into the layout. To display the shop, the elemtn create such as text view is not able to manually create via XML due to the unknown number of shops and information. Hence, these elements need to create by using Java code instead in the activity. And further explain in Figure 5-2-6-2-1-18 to Figure 5-

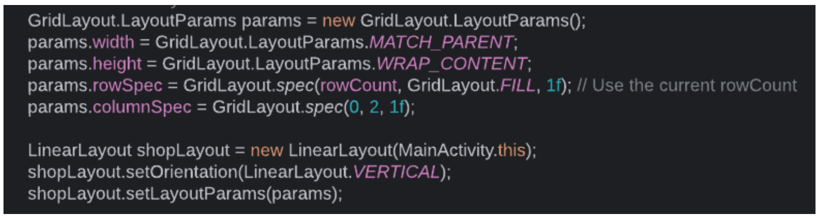
2-6-2-1-21. This logic of the code implementation is similar to the code in the Shop List Display Activity that display shops for the category.



```java
private void displayShopWithReviewsminimarket(String shopName, String shopID, float averageRating, String shopcat, int reviewCount) {
    // Create a GridLayout.LayoutParams to specify the position of the shopLayout
    // Create a GridLayout to hold the data
    GridLayout.LayoutParams params = new GridLayout.LayoutParams();
    params.width = GridLayout.LayoutParams.MATCH_PARENT;
    params.height = GridLayout.LayoutParams.WRAP_CONTENT;
    params.rowSpec = GridLayout.spec(rowCount111, GridLayout.FILL, 1f); // Use the current rowCount
    params.columnSpec = GridLayout.spec(0, 2, 1f);

    LinearLayout shopLayout = new LinearLayout(MainActivity.this);
    shopLayout.setOrientation(LinearLayout.VERTICAL);
    shopLayout.setLayoutParams(params);

    // Create TextViews for shop name, average rating, and review count
    TextView NAME = new TextView(MainActivity.this);

    // Create a GradientDrawable
    GradientDrawable gradientDrawable = new GradientDrawable();
    // Set the shape (rectangle in this case)
    gradientDrawable.setShape(GradientDrawable.RECTANGLE);
    // Set the corner radius (in pixels)
    gradientDrawable.setCornerRadius(convertDpToPx(12)); // Convert dp to pixels
    // Set the background color
    gradientDrawable.setColor(Color.parseColor("#85C1E9")); // Set the color

    ViewGroup.LayoutParams layoutParams1 = new ViewGroup.LayoutParams(
            convertDpToPx(180), // Width in pixels
            convertDpToPx(25) // Height in pixels
    );
    // Set gravity
    NAME.setGravity(Gravity.CENTER);

    // Set LayoutParams to the TextView
    NAME.setLayoutParams(layoutParams1);

    // Set the GradientDrawable as the background of the TextView
    NAME.setBackground(gradientDrawable);
    NAME.setText(shopName);

    TextView ratingTextView = new TextView(MainActivity.this);
    ratingTextView.setText("   Average Rating: " + averageRating + "   Total review: " + reviewCount);

    // Set an OnClickListener for the LinearLayout
    shopLayout.setOnClickListener(v -> {
        Intent intent2 = new Intent(MainActivity.this, ShopDisplay.class);
        intent2.putExtra("shopID", shopID);
        startActivity(intent2);
    });

    // Add TextViews to the LinearLayout
    shopLayout.addView(NAME);
    shopLayout.addView(ratingTextView);

    // Increment the rowCount for the next view
    rowCount111++;

    // Add the LinearLayout to the GridLayout with the specified LayoutParams
    gridLayout111.addView(shopLayout, params);
}
```

Figure 5-2-6-2-1-17 Main Activity Code: Shop Displaying Function

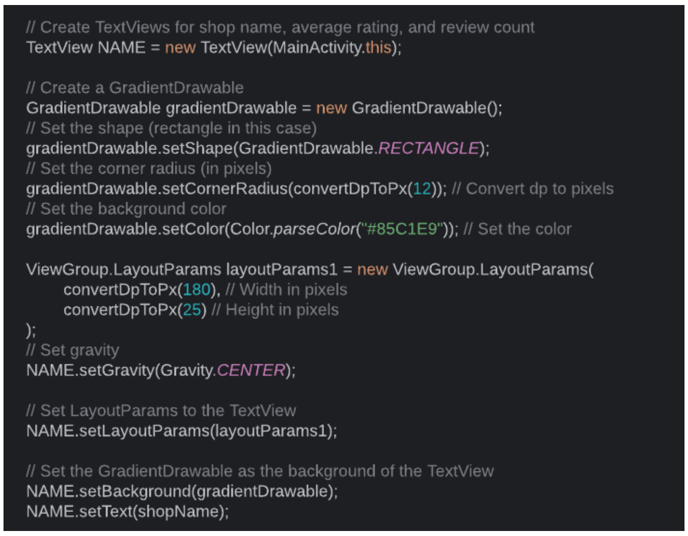
Moving on, this section explain about shop displaying code from Figure 5-2-6-2-1-18 to Figure 5-2-6-2-1-20. Which having similar code implementation logic for other activities as well. According to the code in Figure 5-2-6-2-1-18, a variable named "params" is created. There will be interface design of the layout and apply it at the end of the code. As the code present, it set the with match with parent layout, and heigh match with the content. Next, it set the numbers of rows and columns according to the total number of the shops. Next, it creates a new layout in types of linear layout and set with the parameters that created.



```
GridLayout.LayoutParams params = new GridLayout.LayoutParams();
params.width = GridLayout.LayoutParams.MATCH_PARENT;
params.height = GridLayout.LayoutParams.WRAP_CONTENT;
params.rowSpec = GridLayout.spec(rowCount, GridLayout.FILL, 1f); // Use the current rowCount
params.columnSpec = GridLayout.spec(0, 2, 1f);

LinearLayout shopLayout = new LinearLayout(MainActivity.this);
shopLayout.setOrientation(LinearLayout.VERTICAL);
shopLayout.setLayoutParams(params);
```

Figure 5-2-6-2-1-18 Main Activity Code: Shop Displaying Function Detail Part 1

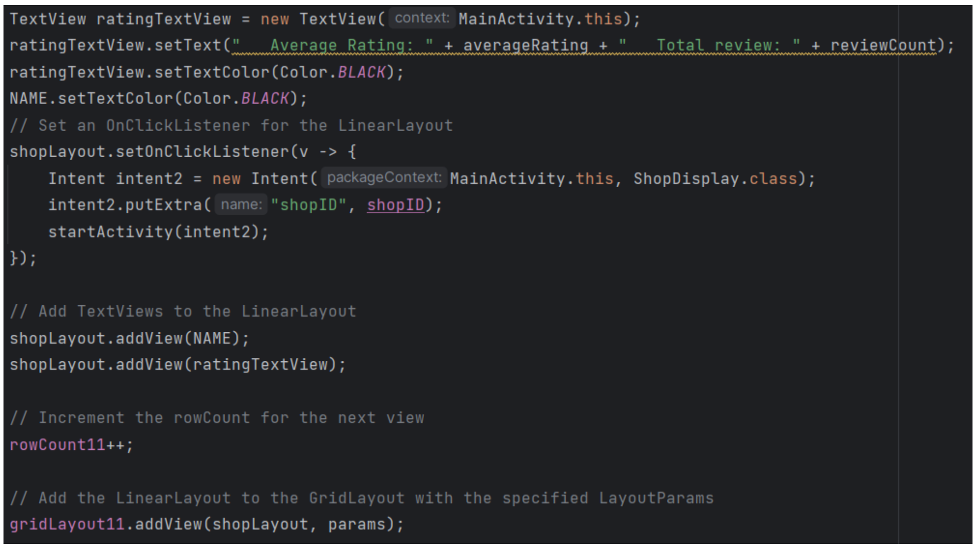
Figure 5-2-6-2-1-19 show a code that creating a text view and design it to have a shop for attracting user. Firstly, text view named as “NAME” is created. Gradient drawable is used for drawing shops, setting color, setting size to the elements itself. In these cases, design setting such as shape, size, color, and radius are applied. And there is method named “convertDptoPx” is created manually to convert the size type. This text view is set to gravity center, hence the content of the text view will be center of the shop regardless of the size of the shop is created. Lastly, the “NAME” using a function named “setText” to set the value of shop name into the text view.



```
// Create TextViews for shop name, average rating, and review count
TextView NAME = new TextView(MainActivity.this);

// Create a GradientDrawable
GradientDrawable gradientDrawable = new GradientDrawable();
// Set the shape (rectangle in this case)
gradientDrawable.setShape(GradientDrawable.RECTANGLE);
// Set the corner radius (in pixels)
gradientDrawable.setCornerRadius(convertDpToPx(12)); // Convert dp to pixels
// Set the background color
gradientDrawable.setColor(Color.parseColor("#85C1E9")); // Set the color

ViewGroup.LayoutParams layoutParams1 = new ViewGroup.LayoutParams(
        convertDpToPx(180), // Width in pixels
        convertDpToPx(25) // Height in pixels
);
// Set gravity
NAME.setGravity(Gravity.CENTER);

// Set LayoutParams to the TextView
NAME.setLayoutParams(layoutParams1);

// Set the GradientDrawable as the background of the TextView
NAME.setBackground(gradientDrawable);
NAME.setText(shopName);
```

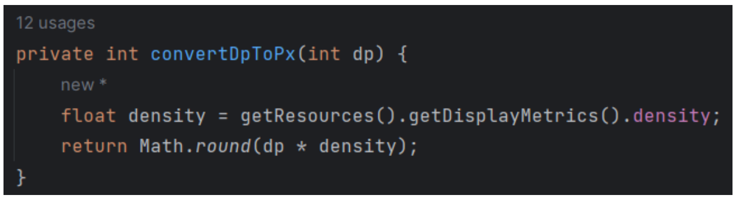
Figure 5-2-6-2-1-19 Main Activity Code: Shop Displaying Function Detail Part 2

In Figure 5-2-6-2-1-20, a text view named “ratingTextView” created and set the text that contain average rating and the number of reviews with a font color of black. Next, there is function preparing to be call for passing the value of shop ID into next activity known as Shop Display Activity to display detail information and functions. Lastly, the element such as text view created will add to the layout using the function named “addView”. The value of row count will be increase as well, so it can be added up into the grid layout created. Figure 5-2-6-1-21 present the function of converting the size types.



```java
TextView ratingTextView = new TextView( context: MainActivity.this);
ratingTextView.setText("    Average Rating:  " + averageRating + "    Total review:  " + reviewCount);
ratingTextView.setTextColor(Color.BLACK);
NAME.setTextColor(Color.BLACK);
// Set an OnClickListener for the LinearLayout
shopLayout.setOnClickListener(v -> {
    Intent intent2 = new Intent( packageContext: MainActivity.this, ShopDisplay.class);
    intent2.putExtra( name: "shopID", shopID);
    startActivity(intent2);
});

// Add TextViews to the LinearLayout
shopLayout.addView(NAME);
shopLayout.addView(ratingTextView);

// Increment the rowCount for the next view
rowCount11++;

// Add the LinearLayout to the GridLayout with the specified LayoutParams
gridLayout11.addView(shopLayout, params);
```

Figure 5-2-6-2-1-20 Main Activity Code: Shop Displaying Function Detail Part 3

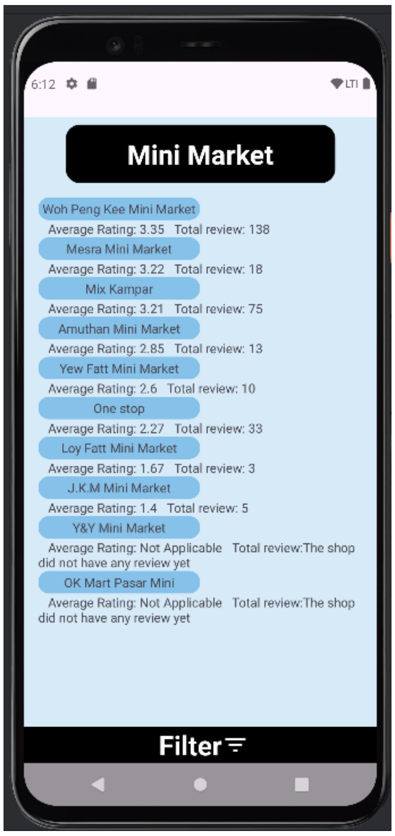


```java
12 usages
private int convertDpToPx(int dp) {
    new *
    float density = getResources().getDisplayMetrics().density;
    return Math.round(dp * density);
}
```

Figure 5-2-6-2-1-21 Main Activity Code: Conversion Function for Size Type

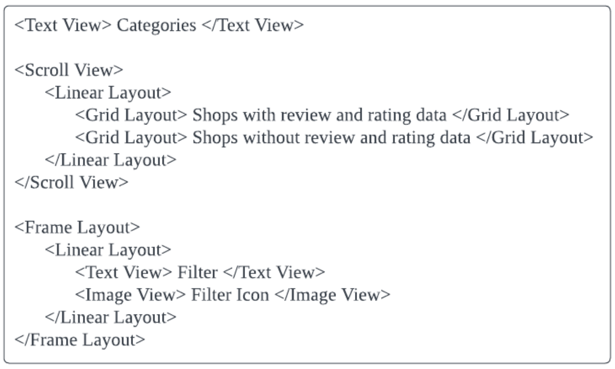
5.2.6.2.2 Shop List Display

If user choose one of the categories, it will display the shops with descending order of average rating. With a filter function that display at bottom of the screen. The main information that displayed are the shop name, average rating, and total review received. In Figure 5-2-6-2-2-1, it shows the interface of the page developed. Which have some difference with the draft design of interface (Figure 4-3-2). This page is able to let user access by only one way which is from the main page. This page will display all of the shops for the particular category chosen. As the example shown in thie figure, the category was mini market; hence, it will display all the shops with category mini market. The displaying logic is according to the ordering of average rating. Other than displaying the shops, the page provided a filter function for the user to perform further filtering and ranking to get reommendations based on the criteria and requirement set by the user.



Woh Peng Kee Mini Market
Average Rating: 3.35 Total review: 138
Mesra Mini Market
Average Rating: 3.22 Total review: 18
Mix Kampar
Average Rating: 3.21 Total review: 75
Amuthan Mini Market
Average Rating: 2.85 Total review: 13
Yew Fatt Mini Market
Average Rating: 2.6 Total review: 10
One stop
Average Rating: 2.27 Total review: 33
Loy Fatt Mini Market
Average Rating: 1.67 Total review: 3
J.K.M Mini Market
Average Rating: 1.4 Total review: 5
Y&Y Mini Market
Average Rating: Not Applicable Total review:The shop did not have any review yet
OK Mart Pasar Mini
Average Rating: Not Applicable Total review:The shop did not have any review yet

Figure 5-2-6-2-2-1 Application Interface: Shop List Display Activity



```
<Text View> Categories </Text View>

<Scroll View>
     <Linear Layout>
          <Grid Layout> Shops with review and rating data </Grid Layout>
          <Grid Layout> Shops without review and rating data </Grid Layout>
     </Linear Layout>
</Scroll View>

<Frame Layout>
     <Linear Layout>
          <Text View> Filter </Text View>
          <Image View> Filter Icon </Image View>
     </Linear Layout>
</Frame Layout>
```

Figure 5-2-6-2-2-2 Simplified Version of Interface (XML) of Shop List Display Activity

Variable Name	**Variable Data Type**	**Purpose**
selectedValue	Integer	To score the choice that user choose from two radio button group
selectedOption1	Integer	
selectedUOption2	Integer	
shopCat	String	To save the category of shop.
filter	Frame Layout	To linked with the layout created in XML and display the shops according to categories, and the filter function. Each grid layout is representing one category of shop.
gridLayout	Grid Layout	
gridLayout2	Grid Layout	
db	Firebase Firestrore	To access the Firebase Firestore and retrieve data saved.
rowCount	Integer	To create new grid layout for each shops by tracking the number.
rouwCount2	Integer	
shopCat	String	To retrieve the value passed from previous activity and saved.
shopdonthavereview	Integer	To determine the recommendation criteria chosen by user.
shopdonthavesar	Integer	
totalRating	Float	To calculate the average rating of a shop
reviewCount	Float	
totalResults	Float	To calculate the percentage of a shop getting re positive review.
result2Count	Float	
averageRating	Float	To save the result of average rating and percentage.
percentage	Float	

shopList	Shop Data Class	To save the shop that fulfilled recommendation criteria set. ShopData is a class create within the Main Activity.

Table 5-2-6-2-2-1 Shop list Display Code: Variables Creation

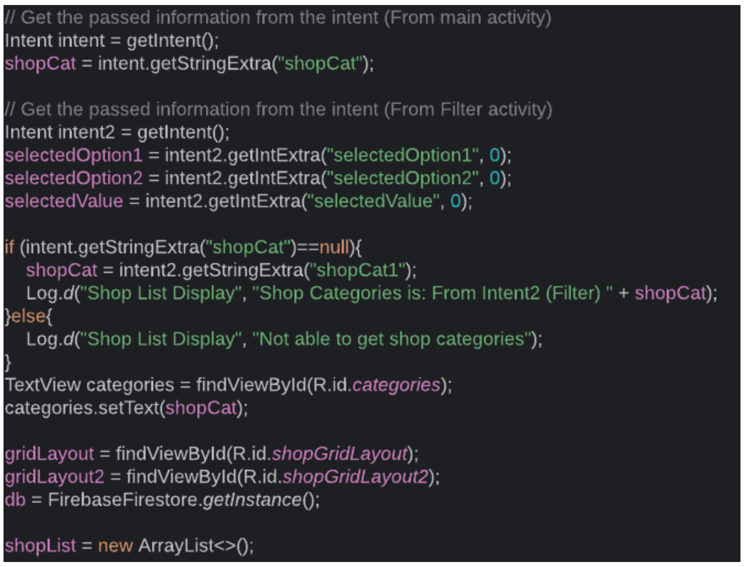
The class that shown in Figure 5-2-6-2-1-7 is created in Shop List Display Activity as well. Overall, the logic of implementation for retrieving data from Firebase Forestore about Shop Information table, Customer Review table, and Sentiment Analysis Result table is same. The logic and coding of the shop displaying to the layout is also the same. Which explained in Chapter 5-2-6-1-1, Figure 5-2-6-2-1-8, and Figure 2-2-6-2-1-11 to Figure 5-2-6-2-1-15. However, there is slightly different when performing adding shop into the array list created by a class as object. The requirement is different according to the criteria set by the user in Filter Activity. To determine the criteria, there are main three variable that passed from Filter Activity to Shop List Display Activity (Table 5-2-6-2-2-1). Which are "selectedValue", "selectedOption1", and "selectedoption2". In Table 5-2-6-2-2-2 explain in more detail elaborations. Where "selectedOption1" refer to either using all data or data within three months. "selectedValue" refer to the range of average rating that classified into 5 level. Lastly, "selectedOption2" refer to the range of percentage of a shop getting positive review, and it divided into 5 level.

Variables	**Elaborations**
selectedOption1	It determines how many review is used for calculating average rating. Value 1 for use all review. Value 2 for use only the review within current three month
selectedValue	It determines the range of average rating that user choose. Value 1 for rating between 0 to 0.99 Value 2 for rating between 1 to 1.99 Value 3 for rating between 2 to 2.99 Value 4 for rating between 3 to 3.99 Value 5 for rating between 4 to 5
selectedOption2	It determines the range of percentage of a shop receiving positive reviews based on sentiment analysis result. Value 1 for range in 26% to 40% Value 2 for range in 41% to 55%

	Value 3 for range in 56% to 70% Value 4 for range in 71% to 85% Value 5 for range in 86% to 100%

Table 5-2-6-2-2-2 Shop list Display Code: Description of Variable Criteria Determination

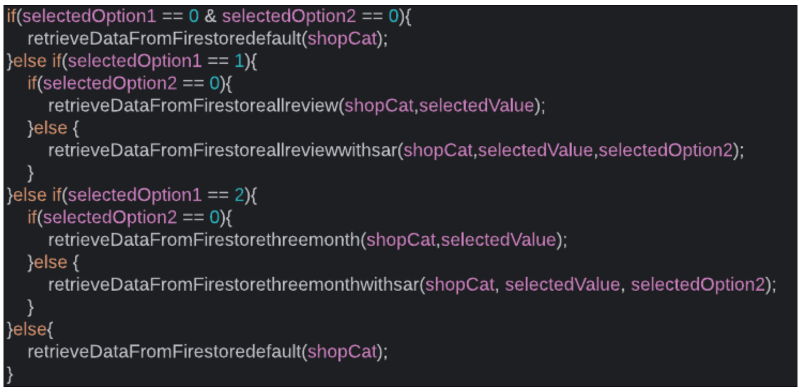
In this activity, it displays the shop either from default or based on requirements set by user from Filter Activity. There are few elements of the code to perform different functions. According to Figure 5-2-6-2-2-3, there are few steps to perform. Firstly, retrieve the data from filter to determine the recommendation requirement. Next, ensuring the shop categories by retrieve and set data to the text view. Before displaying the shop, the layout created in XML must linked and assigned to the variables created in the activity. To retrieve data, initialization of the firebase is performed. An array list of class object is created to prepare add the shop into it to display.



```
// Get the passed information from the intent (From main activity)
Intent intent = getIntent();
shopCat = intent.getStringExtra("shopCat");

// Get the passed information from the intent (From Filter activity)
Intent intent2 = getIntent();
selectedOption1 = intent2.getIntExtra("selectedOption1", 0);
selectedOption2 = intent2.getIntExtra("selectedOption2", 0);
selectedValue = intent2.getIntExtra("selectedValue", 0);

if (intent.getStringExtra("shopCat")==null){
   shopCat = intent2.getStringExtra("shopCat1");
   Log.d("Shop List Display", "Shop Categories is: From Intent2 (Filter) " + shopCat);
}else{
   Log.d("Shop List Display", "Not able to get shop categories");
}
TextView categories = findViewById(R.id.categories);
categories.setText(shopCat);

gridLayout = findViewById(R.id.shopGridLayout);
gridLayout2 = findViewById(R.id.shopGridLayout2);
db = FirebaseFirestore.getInstance();

shopList = new ArrayList<>();
```

Figure 5-2-6-2-2-3 Code of Shop List Display Activity _ onCreate Method Part 1

Moving on, using an IF loop to call the correct function based on the requirement set and pass the needed variables to the function. The default value of "selectedOption1" and "selectedOption2" is 0. Hence, if there is no requirement set from filter, it will display the shop only with ordering of the average rating. Table 5-2-6-2-2-3 show all possibilities of the requirement set.



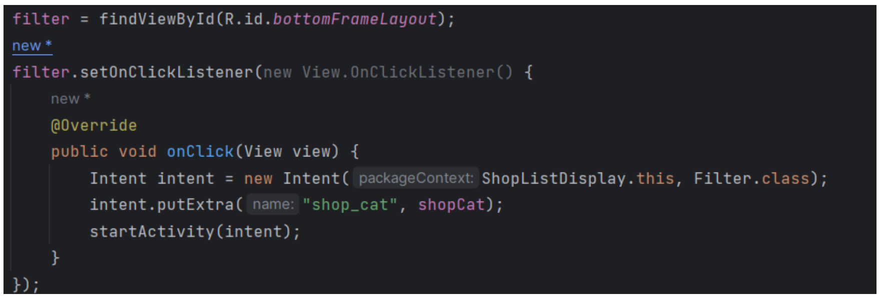
```
if(selectedOption1 == 0 & selectedOption2 == 0){
    retrieveDataFromFirestoredefault(shopCat);
}else if(selectedOption1 == 1){
    if(selectedOption2 == 0){
        retrieveDataFromFirestoreallreview(shopCat,selectedValue);
    }else {
        retrieveDataFromFirestoreallreviewwithsar(shopCat,selectedValue,selectedOption2);
    }
}else if(selectedOption1 == 2){
    if(selectedOption2 == 0){
        retrieveDataFromFirestorethreemonth(shopCat,selectedValue);
    }else {
        retrieveDataFromFirestorethreemonthwithsar(shopCat, selectedValue, selectedOption2);
    }
}else{
    retrieveDataFromFirestoredefault(shopCat);
}
```

Figure 5-2-6-2-2-4 Code of Shop List Display Activity _ onCreate Method Part 2

No Cases	Possibilities	Page
1	Default: Display All Shops Based on Average Rating Ordering with All Data	A-59
2	Display Shops Based on Average Rating Ordering with All Data and Range of Rating is Selected	A-25
3	Display Shops Based on Average Rating with Three Month Data and Range of Rating is Selected	A-29
4	Default: Display Shops Based on Average Rating Ordering with All Data and Range of Rating is Selected, and Sentiment Analysis Result	A-46
5	Display Shops Based on Average Rating with Three Month Data and Range of Rating is Selected, and Sentiment Analysis Result	A-33

Table 5-2-6-2-2-3 Shop list Display Code: Shop Displaying Criteria Possibilities and Its Code Placement in Appendix A.2

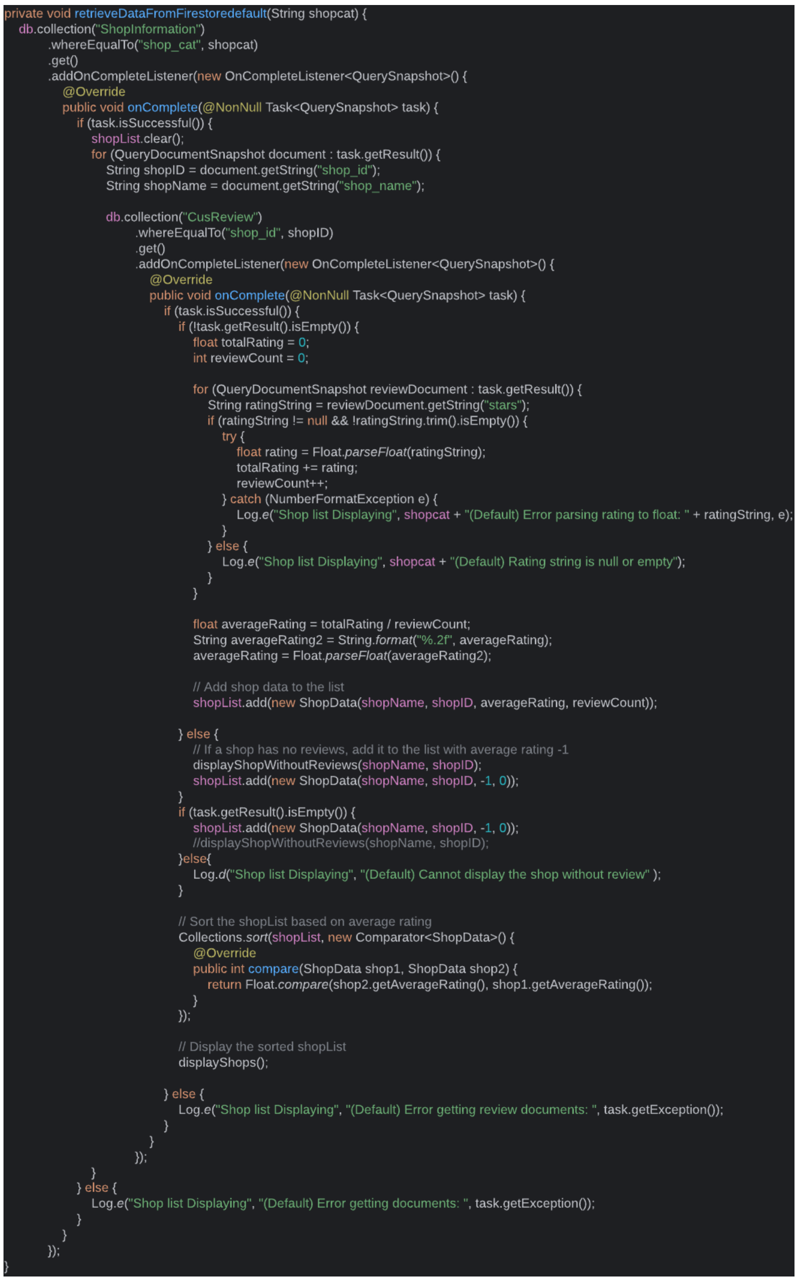
Lastly, Figure 5-2-6-2-2-5 show the code of setting on click listener for the filter function that proceed to next pages.



```
filter = findViewById(R.id.bottomFrameLayout);
new *
filter.setOnClickListener(new View.OnClickListener() {
    new *
    @Override
    public void onClick(View view) {
        Intent intent = new Intent( packageContext: ShopListDisplay.this, Filter.class);
        intent.putExtra( name: "shop_cat", shopCat);
        startActivity(intent);
    }
});
```

Figure 5-2-6-2-2-5 Code of Shop List Display Activity _ onCreate Method Part 3

Next, the logic of implementation for data retrieving and displaying is mostly same for all five cases with the code explanation of Chapter 5.2.6.2.1 of the Figure 5-2-6-2-1-9 to Figure 5-2-6-2-1-21. However, the main difference is the method of adding the shop into the object arrays based on the requirements and criteria. Another different is the data retrieving. Where case 1, 2 and 4 retrieve all data, and case 3 and 5 retrieve data within three months. As for retrieving data within three months, it is same with the Chapter 5-2-6-2-1 Figure 5-2-6-2-1-9 to Figure 5-2-6-2-1-21. Figure 5-2-6-2-2-6 show the example code of retrieving all data in case one and simplified into Figure 5-2-6-2-2-7. The code directly retrieves the data without restricting the month into 3 months.



```java
private void retrieveDataFromFirestoredefault(String shopcat) {
    db.collection("ShopInformation")
            .whereEqualTo("shop_cat", shopcat)
            .get()
            .addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
                @Override
                public void onComplete(@NonNull Task<QuerySnapshot> task) {
                    if (task.isSuccessful()) {
                        shopList.clear();
                        for (QueryDocumentSnapshot document : task.getResult()) {
                            String shopID = document.getString("shop_id");
                            String shopName = document.getString("shop_name");

                            db.collection("CusReview")
                                    .whereEqualTo("shop_id", shopID)
                                    .get()
                                    .addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
                                        @Override
                                        public void onComplete(@NonNull Task<QuerySnapshot> task) {
                                            if (task.isSuccessful()) {
                                                if (!task.getResult().isEmpty()) {
                                                    float totalRating = 0;
                                                    int reviewCount = 0;

                                                    for (QueryDocumentSnapshot reviewDocument : task.getResult()) {
                                                        String ratingString = reviewDocument.getString("stars");
                                                        if (ratingString != null && !ratingString.trim().isEmpty()) {
                                                            try {
                                                                float rating = Float.parseFloat(ratingString);
                                                                totalRating += rating;
                                                                reviewCount++;
                                                            } catch (NumberFormatException e) {
                                                                Log.e("Shop list Displaying", shopcat + "(Default) Error parsing rating to float: " + ratingString, e);
                                                            }
                                                        } else {
                                                            Log.e("Shop list Displaying", shopcat + "(Default) Rating string is null or empty");
                                                        }
                                                    }

                                                    float averageRating = totalRating / reviewCount;
                                                    String averageRating2 = String.format("%.2f", averageRating);
                                                    averageRating = Float.parseFloat(averageRating2);

                                                    // Add shop data to the list
                                                    shopList.add(new ShopData(shopName, shopID, averageRating, reviewCount));

                                                } else {
                                                    // If a shop has no reviews, add it to the list with average rating -1
                                                    displayShopWithoutReviews(shopName, shopID);
                                                    shopList.add(new ShopData(shopName, shopID, -1, 0));
                                                }
                                                if (task.getResult().isEmpty()) {
                                                    shopList.add(new ShopData(shopName, shopID, -1, 0));
                                                    //displayShopWithoutReviews(shopName, shopID);
                                                }else{
                                                    Log.d("Shop list Displaying", "(Default) Cannot display the shop without review" );
                                                }

                                                // Sort the shopList based on average rating
                                                Collections.sort(shopList, new Comparator<ShopData>() {
                                                    @Override
                                                    public int compare(ShopData shop1, ShopData shop2) {
                                                        return Float.compare(shop2.getAverageRating(), shop1.getAverageRating());
                                                    }
                                                });

                                                // Display the sorted shopList
                                                displayShops();

                                            } else {
                                                Log.e("Shop list Displaying", "(Default) Error getting review documents: ", task.getException());
                                            }
                                        }
                                    });
                        }
                    } else {
                        Log.e("Shop list Displaying", "(Default) Error getting documents: ", task.getException());
                    }
                }
            });
}
```

Figure 5-2-6-2-2-6 Code of Shop List Display Activity _ Example of All Data Retrieving

To determine the criteria and requirement set by the user is a difficulty when perform coding that can achive the requirements. This is due to the complexity of the requirement choices given. The method used initially was switch method, however due to the complexity, it was changed to use IF loop. Hence, multi-layer IF loop is created to achieve the shop displaying according to the criteria and requirement set. According to the Table 5-2-6-2-2-3, there are total 5 cases with its own requirements and criteria. Figure 5-2-6-2-2-7 and Figure 5-2-6-2-2-8 show two diagrams that the difference between 5 cases. There are three variables created for setting the requirements (Table 5-2-6-2-2-2). Which are “selectedOption1”, selectedValue”, and “selectedOption2”. Hence, “selectedOption1” indicate that the case is either retrieve all data or data within three months. If it is case 1, case 2 or case 4, it only retrieves and process all data for average rating calculation from the Customer Review Table. Eventually, perform examination and determination of the fulfillment of requirements and criteria. According to the diagram, “avgrating” refers to the “selectedValue” in Table 5-2-6-2-2-2. For this variable, there are only 5 values contain which are 1 to 5. Each value represents different range of average rating from minimum 0 to maximum 5 of average rating.

For example, a user would like to have case 2 with an average rating of 3 to 3.99. In this case, it indicates a value 1 for “selectedOption1” and value 4 of “selectedValue”. Therefore, the application will display the shops by using all data for average rating calculations and display the shop with average rating of 3 to 3.99 only. Second example, a user would like to have case 5 with an average rating of 4 to 5, and percentage between 71% to 85%. For that, the value of “selectedOption1” is 1, value of “selectedValue” and “avgrating” will be 5, and value of 4 for “sar” and “selectedOption2”. According to these variables value, the system will perform the examination and determination after the calculations of percentage and average rating. Hence, in the application system, there are multi-layer IF loop used for examination and determination of the fulfillment of requirements and criteria. The complete code placed in appendix A.2 from page A-23 to A-63 with a detail of pages stated in Table 5-2-6-2-2-3 for each case’s function code. Table 5-2-6-2-2-4 show the possible values for three variables for 5 cases.

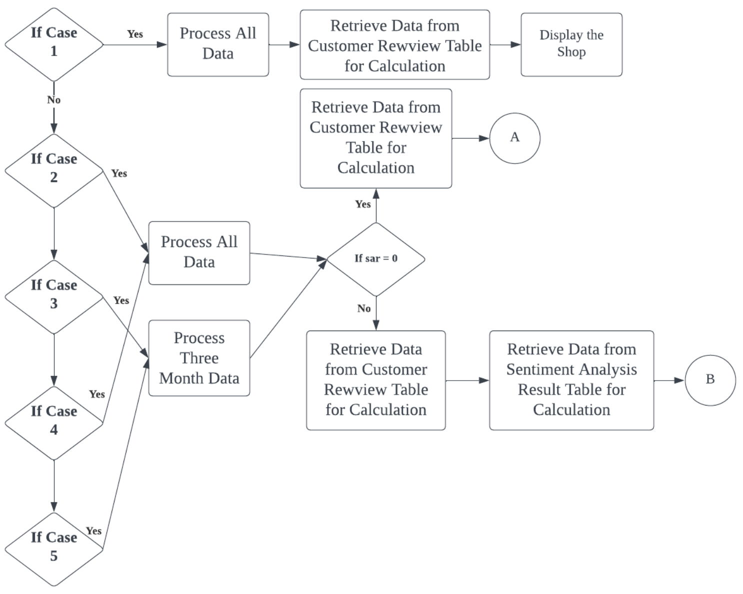

If Case 1
Yes
Process All Data
Retrieve Data from Customer Rewview Table for Calculation
Display the Shop
No
If Case 2
If Case 3
If Case 4
If Case 5
Process All Data
Process Three Month Data
If sar = 0
Retrieve Data from Customer Rewview Table for Calculation
A
Retrieve Data from Customer Rewview Table for Calculation
Retrieve Data from Sentiment Analysis Result Table for Calculation
B

Figure 5-2-6-2-2-7 Data Retrieving and Displaying for 5 Cases with Its Requirements and Criteria Part 1

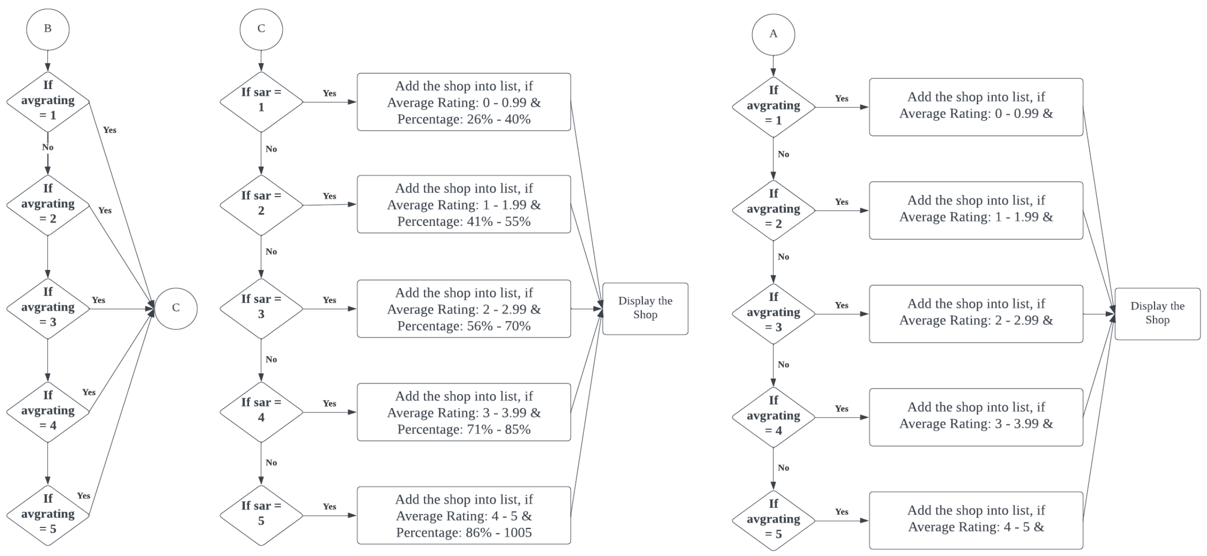
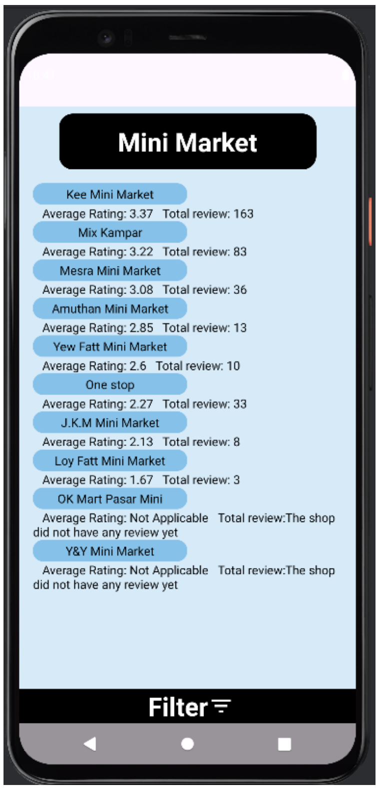

B
If avgrating = 1
If avgrating = 2
If avgrating = 3
If avgrating = 4
If avgrating = 5
C
If sar = 1
Add the shop into list, if Average Rating: 0 - 0.99 & Percentage: 26% - 40%
If sar = 2
Add the shop into list, if Average Rating: 1 - 1.99 & Percentage: 41% - 55%
If sar = 3
Add the shop into list, if Average Rating: 2 - 2.99 & Percentage: 56% - 70%
If sar = 4
Add the shop into list, if Average Rating: 3 - 3.99 & Percentage: 71% - 85%
If sar = 5
Add the shop into list, if Average Rating: 4 - 5 & Percentage: 86% - 1005
Display the Shop
A
Add the shop into list, if Average Rating: 0 - 0.99 &
Add the shop into list, if Average Rating: 1 - 1.99 &
Add the shop into list, if Average Rating: 2 - 2.99 &
Add the shop into list, if Average Rating: 3 - 3.99 &
Add the shop into list, if Average Rating: 4 - 5 &
Display the Shop
Yes
No

Figure 5-2-6-2-2-8 Data Retrieving and Displaying for 5 Cases with Its Requirements and Criteria Part 2

No	Value			Cases
	selectionOption1	selectedOption2	selectedValue	
1	0	0	N/A	Case 1
2	1	0	Value = 1,2,3,4, or 5	Case 2
3	1	Not 0; Value = 1,2,3,4, or 5	Value = 1,2,3,4, or 5	Case 4
4	2	0	Value = 1,2,3,4, or 5	Case 3
5	2	Not 0; Value = 1,2,3,4, or 5	Value = 1,2,3,4, or 5	Case 5

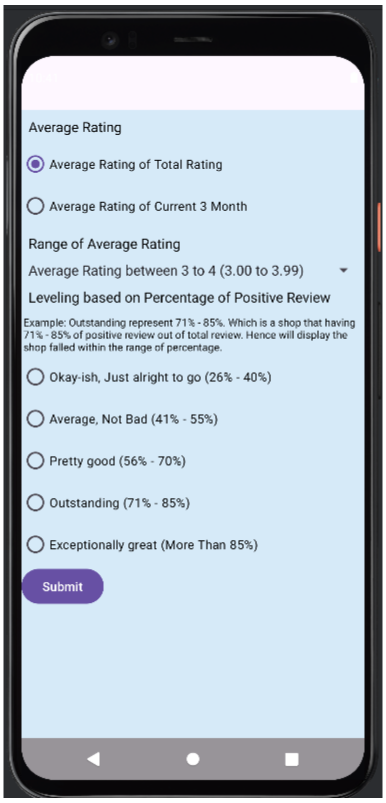
Table 5-2-6-2-2-4 Possible Value of Three Variables for 5 cases

Next, below will show two different case examples. According to Figure 5-2-6-2-2-9, it shows default displaying which only based on average rating which is case 1. Where there are having three shop is having more than 3 average rating, 4 shop having average rating between 2 to 2.99, one shop is having average rating between 1 to 1.99, and two shop do not have any review and rating receive, hence it does not have value of average rating. Next, for first example case known as case 2, Figure 5-2-6-2-2-10 show the Filter page of the example criteria set to filter and get recommendation on shop of mini market. These critetias are using all data, average rating between 3 to 3.99. And Figure 5-2-6-2-2-11 show the result after setting the requirement and criteria. Which only display the three shop that having average rating between 3 to 3.99.


Mini Market

Kee Mini Market
Average Rating: 3.37 Total review: 163
Mix Kampar
Average Rating: 3.22 Total review: 83
Mesra Mini Market
Average Rating: 3.08 Total review: 36
Amuthan Mini Market
Average Rating: 2.85 Total review: 13
Yew Fatt Mini Market
Average Rating: 2.6 Total review: 10
One stop
Average Rating: 2.27 Total review: 33
J.K.M Mini Market
Average Rating: 2.13 Total review: 8
Loy Fatt Mini Market
Average Rating: 1.67 Total review: 3
OK Mart Pasar Mini
Average Rating: Not Applicable Total review:The shop did not have any review yet
Y&Y Mini Market
Average Rating: Not Applicable Total review:The shop did not have any review yet

Table 5-2-6-2-2-9 Mini Market Shops Default Display


Average Rating

[✓ Average Rating of Total Rating]

Average Rating of Current 3 Month

Range of Average Rating

Average Rating between 3 to 4 (3.00 to 3.99)

Leveling based on Percentage of Positive Review

Example: Outstanding represent 71% - 85%. Which is a shop that having 71% - 85% of positive review out of total review. Hence will display the shop failed within the range of percentage.

Okay-ish, Just alright to go (26% - 40%)

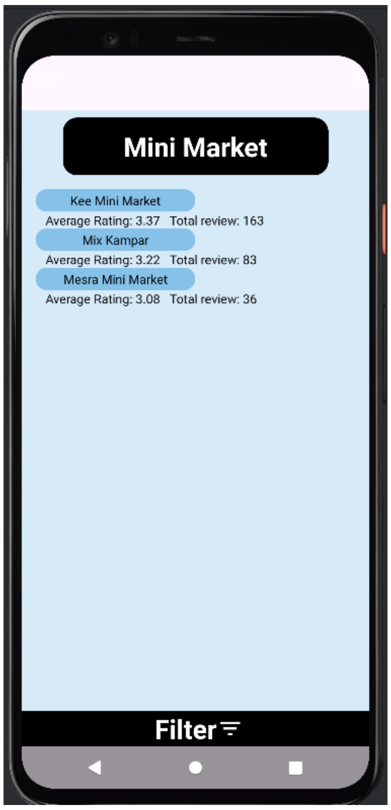
Average, Not Bad (41% - 55%)

Pretty good (56% - 70%)

Outstanding (71% - 85%)

Exceptionally great (More Than 85%)

Table 5-2-6-2-2-10 First Example Criteria Setting


Mini Market

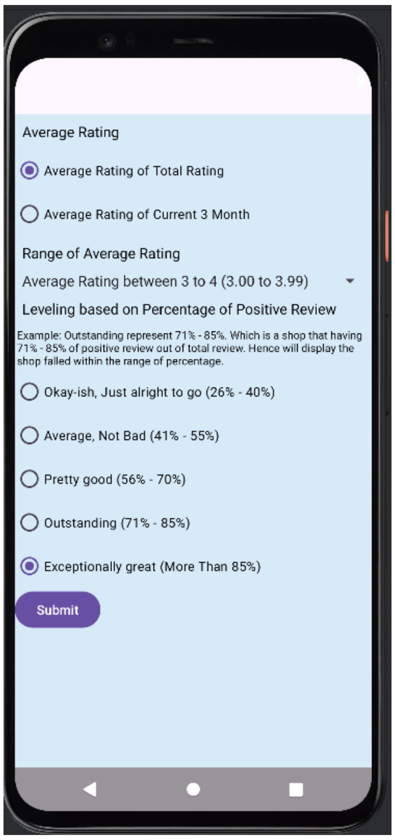
Kee Mini Market
Average Rating: 3.37 Total review: 163

Mix Kampar
Average Rating: 3.22 Total review: 83

Mesra Mini Market
Average Rating: 3.08 Total review: 36

Table 5-2-6-2-2-11 Mini Market Shops Display of First Example

Second example kownn as case 4, the criteria is to get recommendations of mini market with using all data, average rating between 4 to 5, and percentage more than and equal to 86% (Figure 5-2-6-2-2-12). Figure 5-2-6-2-2-13 show the result of shop displaying after filtering. According to the Figure5-2-6-2-2-9, there are total three shop that having average rating between 3 to 3.99, however not all of them is having more than or equal to 86% on getting positive review. According to Figure 5-2-6-2-2-14 to Figure 5-2-6-2-2-16, only shop of Figure5-2-6-2-2-15 is having the percentage more than and equal to 86%. Hence, the result shown only one shop in the Figure 5-2-6-2-2-13.


Average Rating

[✓ Average Rating of Total Rating]

Average Rating of Current 3 Month

Range of Average Rating

Average Rating between 3 to 4 (3.00 to 3.99)

Leveling based on Percentage of Positive Review

Example: Outstanding represent 71% - 85%. Which is a shop that having 71% - 85% of positive review out of total review. Hence will display the shop falled within the range of percentage.

Okay-ish, Just alright to go (26% - 40%)

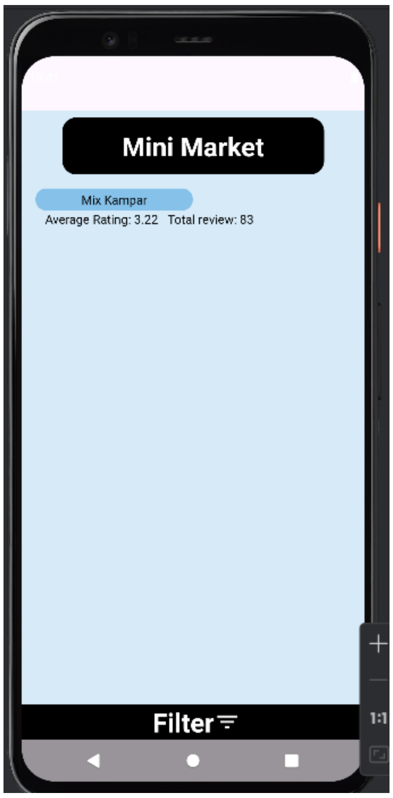
Average, Not Bad (41% - 55%)

Pretty good (56% - 70%)

Outstanding (71% - 85%)

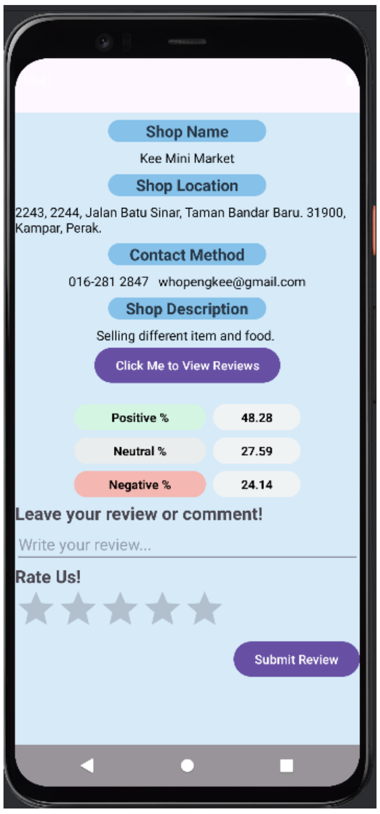
[✓ Exceptionally great (More Than 85%)]

Table 5-2-6-2-2-12 Second Example Criteria Setting


Mini Market

Mix Kampar
Average Rating: 3.22 Total review: 83

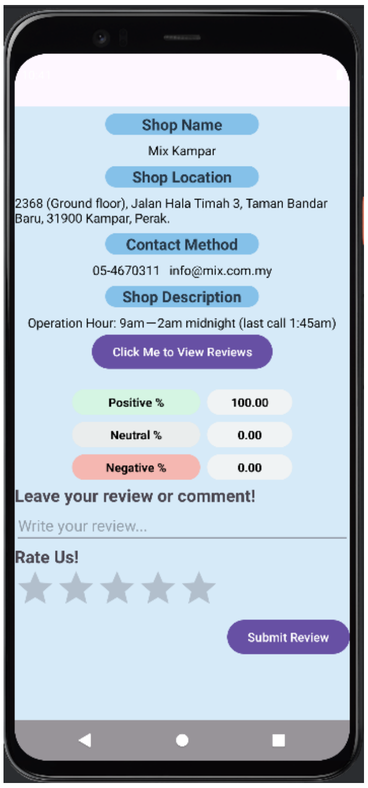
Table 5-2-6-2-2-13 Mini Market Shops Display of Second Example


Shop Name
Kee Mini Market
Shop Location
2243, 2244, Jalan Batu Sinar, Taman Bandar Baru. 31900, Kampar, Perak.
Contact Method
016-281 2847 whopengkee@gmail.com
Shop Description
Selling different item and food.

Positive %	48.28
Neutral %	27.59
Negative %	24.14

Leave your review or comment!
Write your review...
Rate Us!

Table 5-2-6-2-2-14 First Shop with Average Rating between 3 to 3.99



Shop Name

Mix Kampar

Shop Location

2368 (Ground floor), Jalan Hala Timah 3, Taman Bandar Baru, 31900 Kampar, Perak.

Contact Method

05-4670311 info@mix.com.my

Shop Description

Operation Hour: 9am—2am midnight (last call 1:45am)

Click Me to View Reviews

Positive %	100.00
Neutral %	0.00
Negative %	0.00

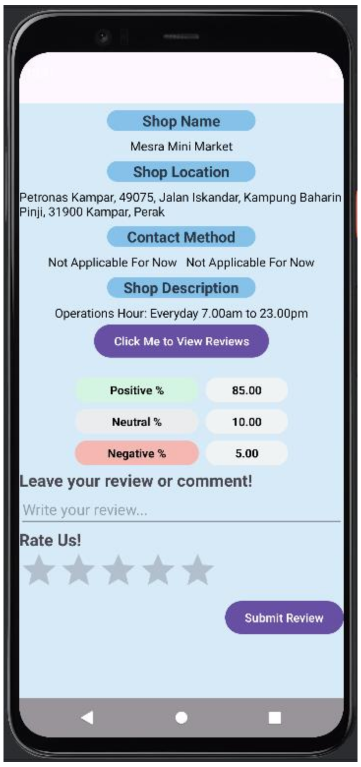
Leave your review or comment!

Write your review...

Rate Us!

Submit Review

Table 5-2-6-2-2-15 Second Shop with Average Rating between 3 to 3.99



Shop Name

Mesra Mini Market

Shop Location

Petronas Kampar, 49075, Jalan Iskandar, Kampung Baharin Pinji, 31900 Kampar, Perak

Contact Method

Not Applicable For Now Not Applicable For Now

Shop Description

Operations Hour: Everyday 7.00am to 23.00pm

Click Me to View Reviews

Positive %	85.00
Neutral %	10.00
Negative %	5.00

Leave your review or comment!

Write your review...

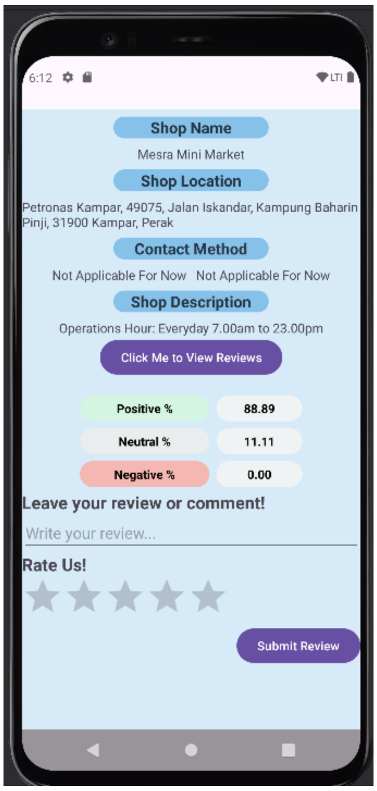
Rate Us!

Submit Review

Table 5-2-6-2-2-16 Third Shop with Average Rating between 3 to 3.99

5.2.6.2.3 Shop Display

In Figure 5-3-6-2-3-1, it shows the interface of the page developed. Which have some difference with the initial planning in terms of draft design (Figure 4-3-3). This activity page will have three ways to access. Firstly, in the main page of the application, if user directly choose one of the shops that displayed, then it will direct the user to this page for displaying the shop information with some function. Second, after user choose a category, and select one of the shops to view. It will direct the user to this page as well. Last, is from itself. Where after user submit a review and rating. It will rerun the activity to ensure the information is updated. In this activity, there are mainly four layout that contain different contents. These four layouts placed under a scroll view. Therefore, the user can scroll the screen of the content is exceeding the screen. First layout displayed the shop information such as location, description, contact method, and others (Referring to Table 4-2-1-1). Second layout displayed button for user to review the past review and rating. These review and rating sorted and displayed in an order based on data and time in another activity. Next, there is a layout displayed the percentage of a shop getting negative, neutral, and positive review which is the result of sentiment analysis. The last layout is a section for user to provide their comments or review with a rating from 1 to 5 stars.



Mesra Mini Market

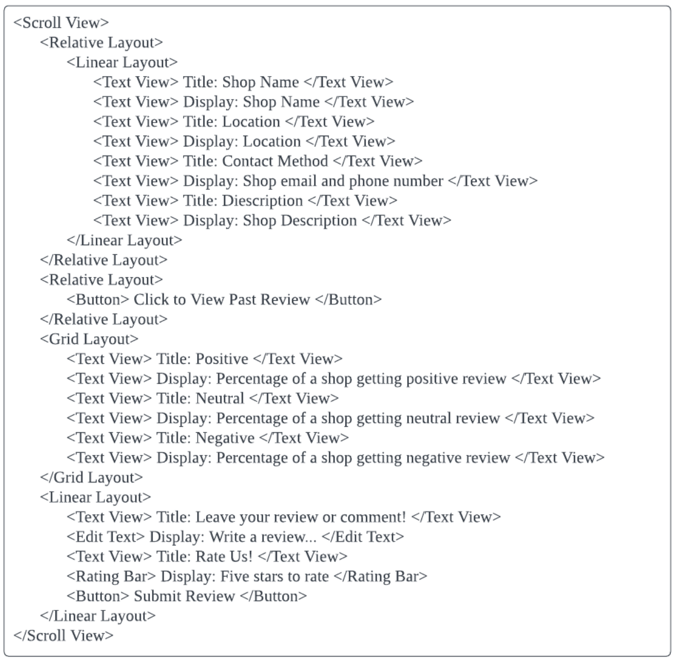
Petronas Kampar, 49075, Jalan Iskandar, Kampung Baharin Pinji, 31900 Kampar, Perak

Not Applicable For Now Not Applicable For Now

Operations Hour: Everyday 7.00am to 23.00pm

Positive %	88.89
Neutral %	11.11
Negative %	0.00

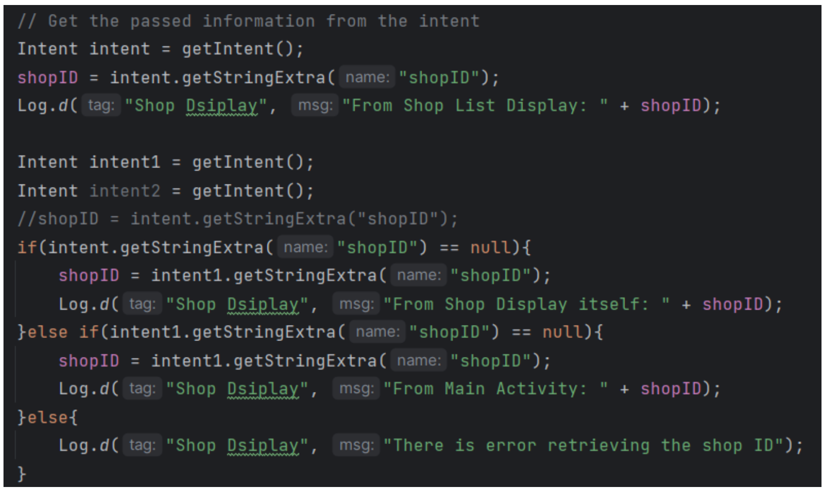
Figure 5-2-6-2-3-1 Application Interface: Shop Display Activity



```
<Scroll View>
    <Relative Layout>
        <Linear Layout>
            <Text View> Title: Shop Name </Text View>
            <Text View> Display: Shop Name </Text View>
            <Text View> Title: Location </Text View>
            <Text View> Display: Location </Text View>
            <Text View> Title: Contact Method </Text View>
            <Text View> Display: Shop email and phone number </Text View>
            <Text View> Title: Diescription </Text View>
            <Text View> Display: Shop Description </Text View>
        </Linear Layout>
    </Relative Layout>
    <Relative Layout>
        <Button> Click to View Past Review </Button>
    </Relative Layout>
    <Grid Layout>
        <Text View> Title: Positive </Text View>
        <Text View> Display: Percentage of a shop getting positive review </Text View>
        <Text View> Title: Neutral </Text View>
        <Text View> Display: Percentage of a shop getting neutral review </Text View>
        <Text View> Title: Negative </Text View>
        <Text View> Display: Percentage of a shop getting negative review </Text View>
    </Grid Layout>
    <Linear Layout>
        <Text View> Title: Leave your review or comment! </Text View>
        <Edit Text> Display: Write a review... </Edit Text>
        <Text View> Title: Rate Us! </Text View>
        <Rating Bar> Display: Five stars to rate </Rating Bar>
        <Button> Submit Review </Button>
    </Linear Layout>
</Scroll View>
```

Figure 5-2-6-2-3-2 Simplified Version of Interface (XML) of Shop Display Activity

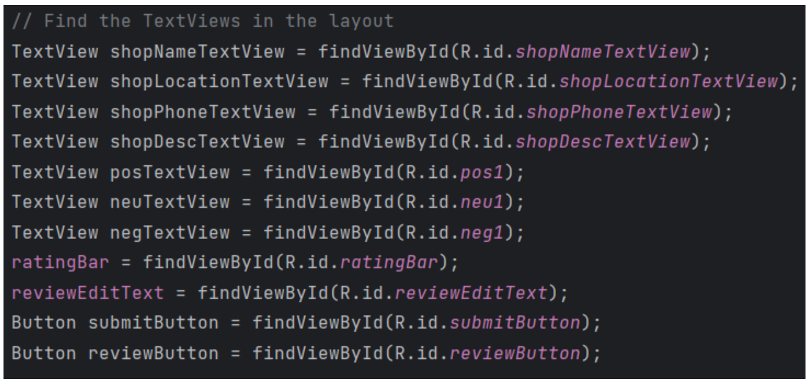
Firstly, to display the shop information. First step is to retrieve the data that passed from previous activity which is shop ID. Which shown in Figure 5-2-6-2-3-3. At first, it will retrieve data from the Shop List Display Activity as default, if it is null, it will retrieve from Shop Display Activity which is itself. If it is null value again, it will retrieve from Main Activity. As there is only three ways to access to this activity.



```
// Get the passed information from the intent
Intent intent = getIntent();
shopID = intent.getStringExtra( name: "shopID");
Log.d( tag: "Shop Dsiplay",  msg: "From Shop List Display: " + shopID);

Intent intent1 = getIntent();
Intent intent2 = getIntent();
//shopID = intent.getStringExtra("shopID");
if(intent.getStringExtra( name: "shopID") == null){
    shopID = intent1.getStringExtra( name: "shopID");
    Log.d( tag: "Shop Dsiplay",  msg: "From Shop Display itself: " + shopID);
}else if(intent1.getStringExtra( name: "shopID") == null){
    shopID = intent1.getStringExtra( name: "shopID");
    Log.d( tag: "Shop Dsiplay",  msg: "From Main Activity: " + shopID);
}else{
    Log.d( tag: "Shop Dsiplay",  msg: "There is error retrieving the shop ID");
}
```

Figure 5-2-6-2-3-3 Code of Shop Display Activity _ onCreate Method Part 1

Next, the element that created in the XML will be linked with the variable created in the activity which shown in Figure 5-2-6-2-3-4. Some variables created before the default method “onCreated” which present in Table 5-2-6-2-3-1.



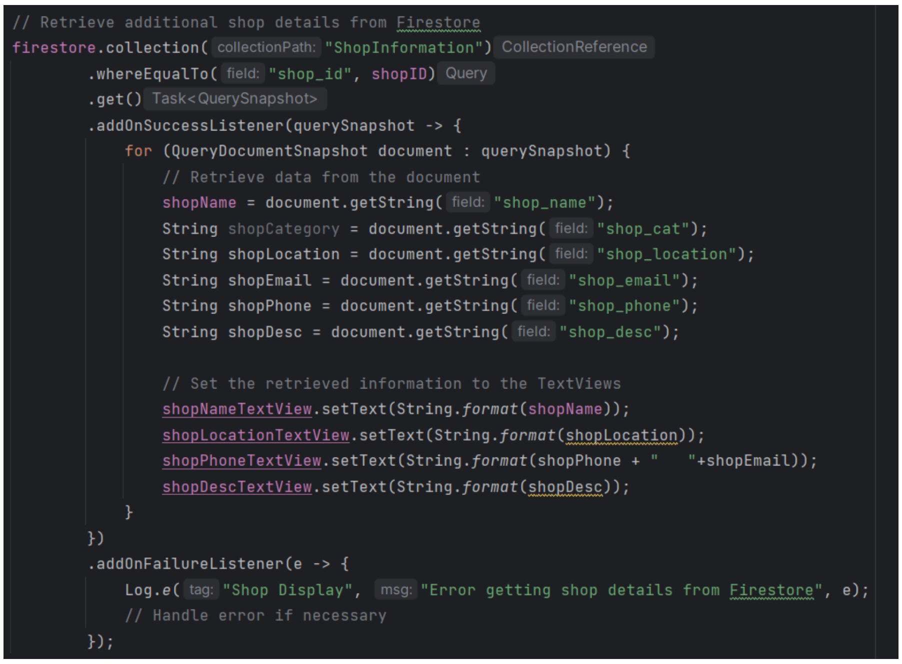
```
// Find the TextViews in the layout
TextView shopNameTextView = findViewById(R.id.shopNameTextView);
TextView shopLocationTextView = findViewById(R.id.shopLocationTextView);
TextView shopPhoneTextView = findViewById(R.id.shopPhoneTextView);
TextView shopDescTextView = findViewById(R.id.shopDescTextView);
TextView posTextView = findViewById(R.id.pos1);
TextView neuTextView = findViewById(R.id.neu1);
TextView negTextView = findViewById(R.id.neg1);
ratingBar = findViewById(R.id.ratingBar);
reviewEditText = findViewById(R.id.reviewEditText);
Button submitButton = findViewById(R.id.submitButton);
Button reviewButton = findViewById(R.id.reviewButton);
```

Figure 5-2-6-2-3-4 Code of Shop Display Activity _ onCreate Method Part 2

Variable Name	**Variable Data Type**	**Purpose**
tflite	Interpreter	To access the model saved in TensorFlow Lite file type
firestore	FirebaseFirestore	To access the Firebase Firestore and retrieve data saved.
shopList	ShopData	To save the shop that fulfilled recommendation criteria set. ShopData is a class create within the Main Activity.
totalRating	Float	To calculate the average rating of a shop
reviewCount	Float	
totalResults	Float	To calculate the percentage of a shop getting a negative, neutral and positive review.
result2Count	Float	
result1Count	Float	
eesult0Count	Float	
ratingBar	RatingBar	To let user to rate from 1 to 5.
reviewEditText	EditText	To let user write their review or comment towards the shop or store.
shopID	String	To retrieve the data that passed from other activity and saved into the variable created.

Table 5-2-6-2-3-1 Shop Display Code: Variables Creation

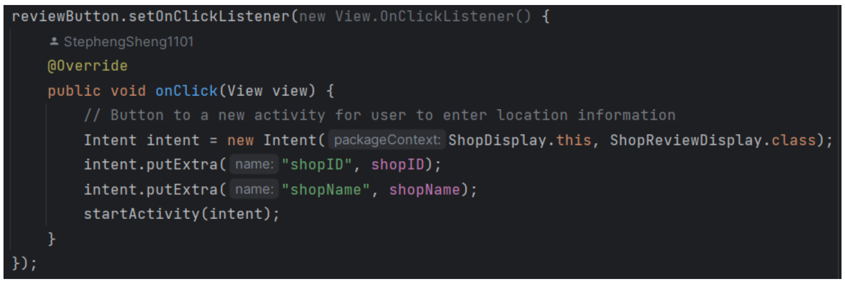
Moving on, started displaying the shop information in the first layout. In Figure 5-2-6-2-3-5, show the complete code from retrieve data from Firebase Firestore to set the value into the user interface element created in XML. It retrieves the shop information according the the shop ID that retrieved from other activity by using "getString" method. And providing the same attribute name in Firestore into the method as field for data retrieving. After retrieving the data, the code will set the value into the text view that created in XML by using "setText" method.



```
// Retrieve additional shop details from Firestore
firestore.collection( collectionPath: "ShopInformation") CollectionReference
        .whereEqualTo( field: "shop_id", shopID) Query
        .get() Task<QuerySnapshot>
        .addOnSuccessListener(querySnapshot -> {
            for (QueryDocumentSnapshot document : querySnapshot) {
                // Retrieve data from the document
                shopName = document.getString( field: "shop_name");
                String shopCategory = document.getString( field: "shop_cat");
                String shopLocation = document.getString( field: "shop_location");
                String shopEmail = document.getString( field: "shop_email");
                String shopPhone = document.getString( field: "shop_phone");
                String shopDesc = document.getString( field: "shop_desc");

                // Set the retrieved information to the TextViews
                shopNameTextView.setText(String.format(shopName));
                shopLocationTextView.setText(String.format(shopLocation));
                shopPhoneTextView.setText(String.format(shopPhone + "    "+shopEmail));
                shopDescTextView.setText(String.format(shopDesc));
            }
        })
        .addOnFailureListener(e -> {
            Log.e( tag: "Shop Display",  msg: "Error getting shop details from Firestore", e);
            // Handle error if necessary
        });
```

Figure 5-2-6-2-3-5 Code of Shop Display Activity _ onCreate Method Part 3

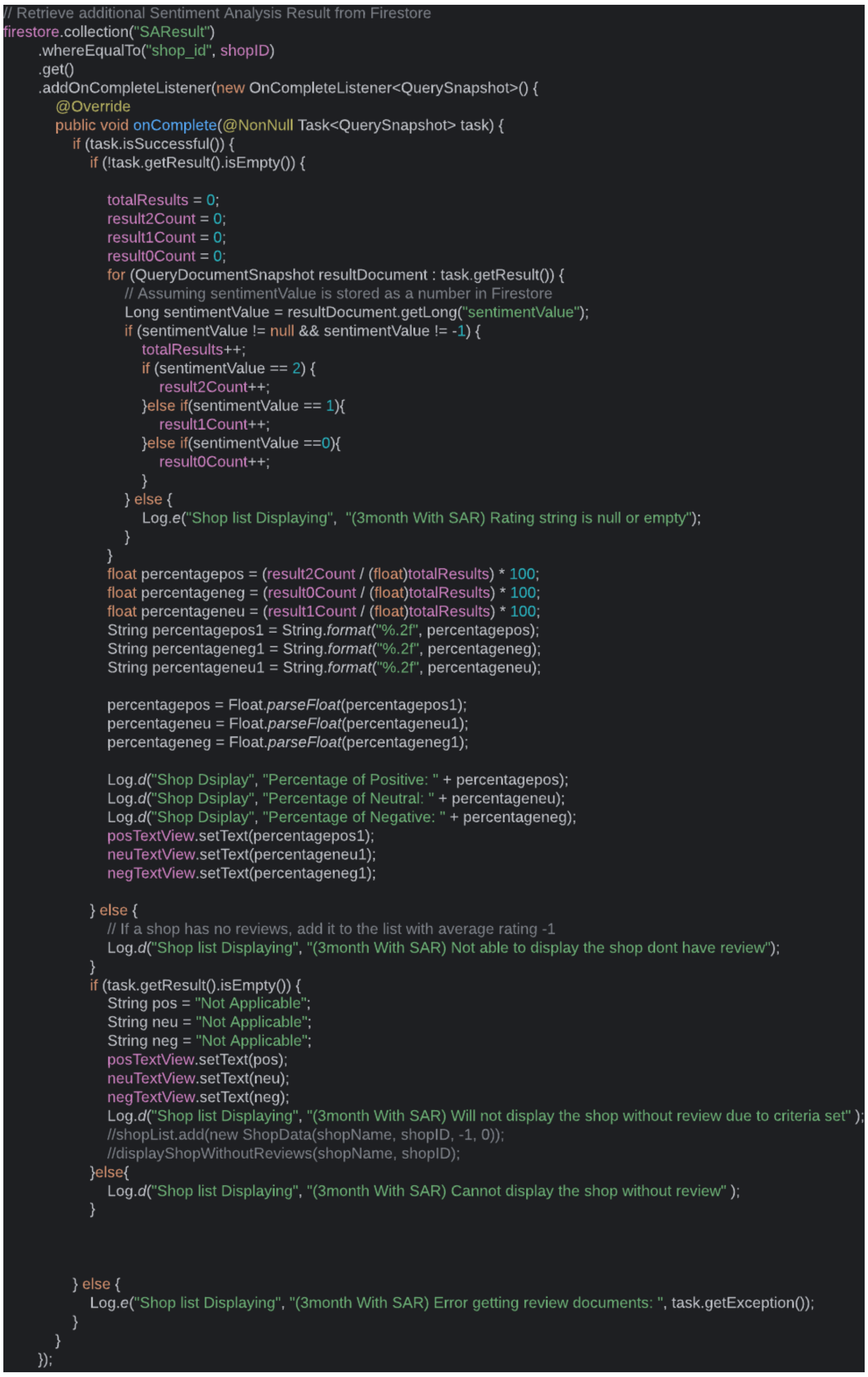
After display the shop information. A button click listener will be set to prepare for activity transition and data passing. Figure 5-2-6-2-3-6 show the code of it, where two data is passed which are the shop ID and name.



```
reviewButton.setOnClickListener(new View.OnClickListener() {
    StephengSheng1101
    @Override
    public void onClick(View view) {
        // Button to a new activity for user to enter location information
        Intent intent = new Intent( packageContext: ShopDisplay.this, ShopReviewDisplay.class);
        intent.putExtra( name: "shopID", shopID);
        intent.putExtra( name: "shopName", shopName);
        startActivity(intent);
    }
});
```

Figure 5-2-6-2-3-6 Code of Shop Display Activity _ onCreate Method Part 4

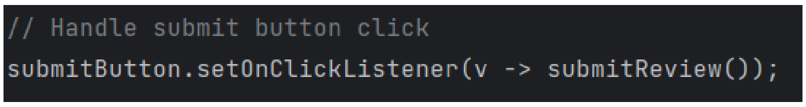
Next, the page is design to display the percentage of a shop getting negative, neutral, and positive review. Figure 5-2-6-2-3-7 show the code on achieving this function. According to the Table 5-2-6-2-3-1, there are four variables created for calculating these percentage. As the sentiment analysis result have 4 types of value which are -1, 0, 1 and 2. For percentage calculation, value of -1 will be excluded. Therefore, IF loop is used to ensure only value of 0, 1 and 2 is included in the percentage calculation. After the calculation is completed, it will set the value into the text view accordingly. If a shop does not have any sentiment analysis result due to no review and rating is received, the value od the three percentage will be "Not Applicable".



```
// Retrieve additional Sentiment Analysis Result from Firestore
firestore.collection("SAResult")
        .whereEqualTo("shop_id", shopID)
        .get()
        .addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
            @Override
            public void onComplete(@NonNull Task<QuerySnapshot> task) {
                if (task.isSuccessful()) {
                    if (!task.getResult().isEmpty()) {

                        totalResults = 0;
                        result2Count = 0;
                        result1Count = 0;
                        result0Count = 0;
                        for (QueryDocumentSnapshot resultDocument : task.getResult()) {
                            // Assuming sentimentValue is stored as a number in Firestore
                            Long sentimentValue = resultDocument.getLong("sentimentValue");
                            if (sentimentValue != null && sentimentValue != -1) {
                                totalResults++;
                                if (sentimentValue == 2) {
                                    result2Count++;
                                }else if(sentimentValue == 1){
                                    result1Count++;
                                }else if(sentimentValue ==0){
                                    result0Count++;
                                }
                            } else {
                                Log.e("Shop list Displaying",  "(3month With SAR) Rating string is null or empty");
                            }
                        }
                        float percentagepos = (result2Count / (float)totalResults) * 100;
                        float percentageneg = (result0Count / (float)totalResults) * 100;
                        float percentageneu = (result1Count / (float)totalResults) * 100;
                        String percentagepos1 = String.format("%.2f", percentagepos);
                        String percentageneg1 = String.format("%.2f", percentageneg);
                        String percentageneu1 = String.format("%.2f", percentageneu);

                        percentagepos = Float.parseFloat(percentagepos1);
                        percentageneu = Float.parseFloat(percentageneu1);
                        percentageneg = Float.parseFloat(percentageneg1);

                        Log.d("Shop Dsiplay", "Percentage of Positive: " + percentagepos);
                        Log.d("Shop Dsiplay", "Percentage of Neutral: " + percentageneu);
                        Log.d("Shop Dsiplay", "Percentage of Negative: " + percentageneg);
                        posTextView.setText(percentagepos1);
                        neuTextView.setText(percentageneu1);
                        negTextView.setText(percentageneg1);

                    } else {
                        // If a shop has no reviews, add it to the list with average rating -1
                        Log.d("Shop list Displaying", "(3month With SAR) Not able to display the shop dont have review");
                    }
                    if (task.getResult().isEmpty()) {
                        String pos = "Not Applicable";
                        String neu = "Not Applicable";
                        String neg = "Not Applicable";
                        posTextView.setText(pos);
                        neuTextView.setText(neu);
                        negTextView.setText(neg);
                        Log.d("Shop list Displaying", "(3month With SAR) Will not display the shop without review due to criteria set" );
                        //shopList.add(new ShopData(shopName, shopID, -1, 0));
                        //displayShopWithoutReviews(shopName, shopID);
                    }else{
                        Log.d("Shop list Displaying", "(3month With SAR) Cannot display the shop without review" );
                    }



                } else {
                    Log.e("Shop list Displaying", "(3month With SAR) Error getting review documents: ", task.getException());
                }
            }
        });
```

Figure 5-2-6-2-3-7 Code of Shop Display Activity _ onCreate Method Part 5

Next, the last part of the code in the on create function is to set a button click listener. When the button is clicked it will call the method which for two purposes. First, it is to save the review and rating into Firebase Firestore. Second, it is to perform sentiment analysis (Figure 5-2-6-2-3-8).

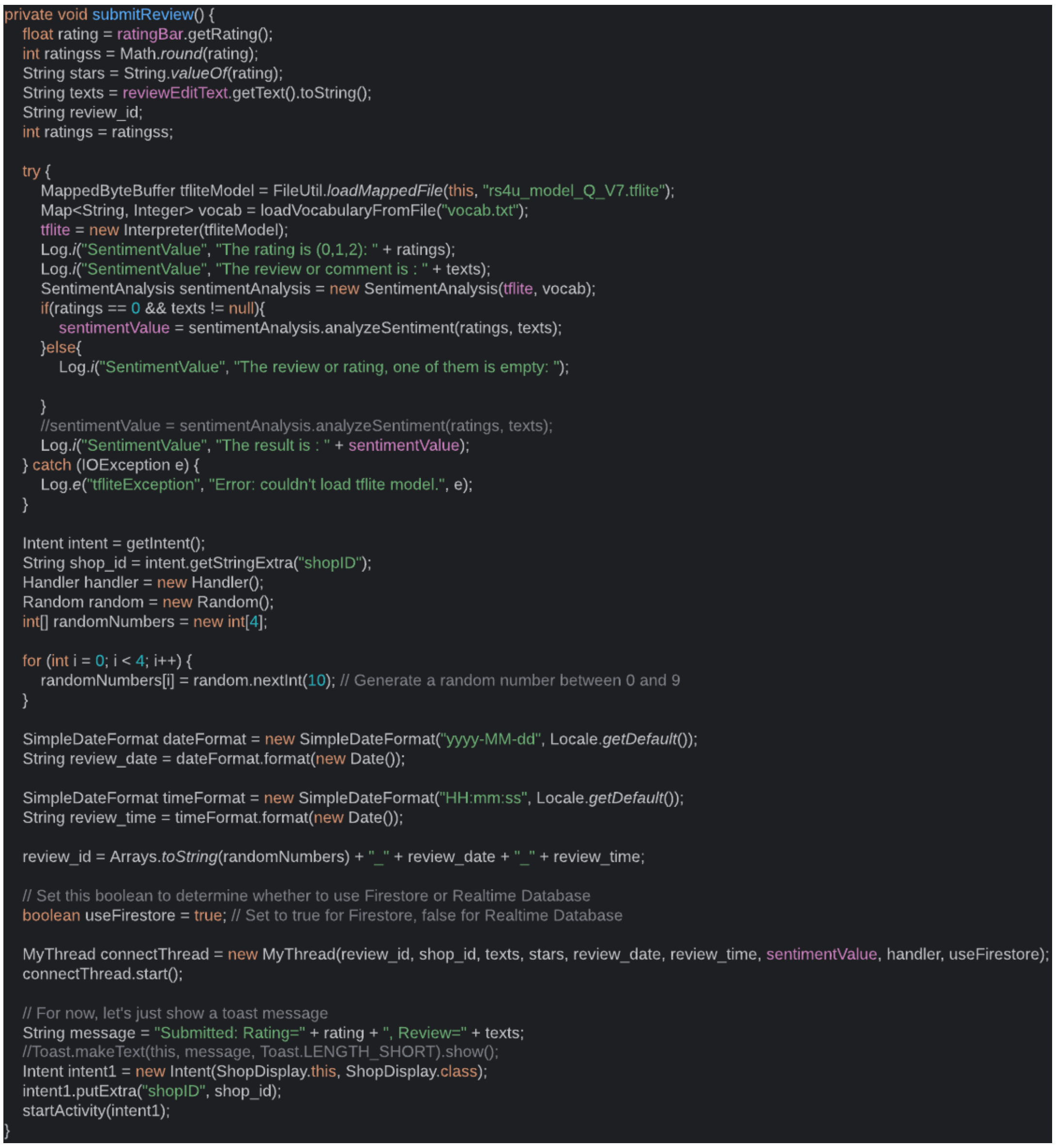


```
// Handle submit button click
submitButton.setOnClickListener(v -> submitReview());
```

Figure 5-2-6-2-3-8 Code of Shop Display Activity _ onCreate Method Part 6

When the submit button of review and rating is clicked, the submit function will be called. Figure 5-2-6-2-3-9 show the complete code. Firstly, is to assign the value of rating and review from the variable created in on create function which are the rating bar and edit text view (Table 5-2-6-2-3-1) and ensuring that the value data type is align with the model input signature which are integer for rating and string for review. Next, try and catch function is used for loading the model trained and its resource such as vocabulary file by using a function created in Figure 5-2-6-2-3-10. This figure shows the vocabulary file loading. After the model and resources file is loaded. This review and rating submit function will create a review ID by combining three components. Firstly, randomly generated 4 number in array coming with the submission date and time. To save the review, rating, and review ID, a thread class is used which show in Figure 5-2-6-2-3-11. After the data is saved into Firebase Firestore, the submit function will rerun and redirect the activity to ensure the data displayed is updated by using intent.

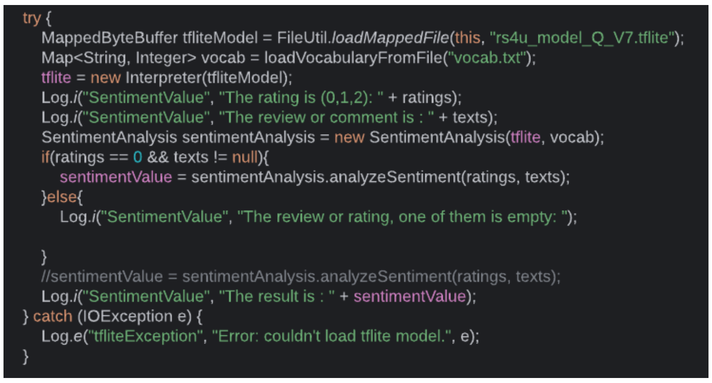
According to the initial system implementation plan stated in Chapter 4, the vocabulary resources is not compulsory and necessary to use in the application. As it is expected to directly import the needed library and use its function to perform preprocessing. However, due to unexpected issue which are the software unable to import the needed resources in term of BERT library and its tokenizer. Hence, the vocabulary resources are downloaded and loaded in the Android Studio to manually perform data preprocessing, especially tokenization process.



```java
private void submitReview() {
    float rating = ratingBar.getRating();
    int ratingss = Math.round(rating);
    String stars = String.valueOf(rating);
    String texts = reviewEditText.getText().toString();
    String review_id;
    int ratings = ratingss;

    try {
        MappedByteBuffer tfliteModel = FileUtil.loadMappedFile(this, "rs4u_model_Q_V7.tflite");
        Map<String, Integer> vocab = loadVocabularyFromFile("vocab.txt");
        tflite = new Interpreter(tfliteModel);
        Log.i("SentimentValue", "The rating is (0,1,2): " + ratings);
        Log.i("SentimentValue", "The review or comment is : " + texts);
        SentimentAnalysis sentimentAnalysis = new SentimentAnalysis(tflite, vocab);
        if(ratings == 0 && texts != null){
            sentimentValue = sentimentAnalysis.analyzeSentiment(ratings, texts);
        }else{
            Log.i("SentimentValue", "The review or rating, one of them is empty: ");

        }
        //sentimentValue = sentimentAnalysis.analyzeSentiment(ratings, texts);
        Log.i("SentimentValue", "The result is : " + sentimentValue);
    } catch (IOException e) {
        Log.e("tfliteException", "Error: couldn't load tflite model.", e);
    }

    Intent intent = getIntent();
    String shop_id = intent.getStringExtra("shopID");
    Handler handler = new Handler();
    Random random = new Random();
    int[] randomNumbers = new int[4];

    for (int i = 0; i < 4; i++) {
        randomNumbers[i] = random.nextInt(10); // Generate a random number between 0 and 9
    }

    SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy-MM-dd", Locale.getDefault());
    String review_date = dateFormat.format(new Date());

    SimpleDateFormat timeFormat = new SimpleDateFormat("HH:mm:ss", Locale.getDefault());
    String review_time = timeFormat.format(new Date());

    review_id = Arrays.toString(randomNumbers) + "_" + review_date + "_" + review_time;

    // Set this boolean to determine whether to use Firestore or Realtime Database
    boolean useFirestore = true; // Set to true for Firestore, false for Realtime Database

    MyThread connectThread = new MyThread(review_id, shop_id, texts, stars, review_date, review_time, sentimentValue, handler, useFirestore);
    connectThread.start();

    // For now, let's just show a toast message
    String message = "Submitted: Rating=" + rating + ", Review=" + texts;
    //Toast.makeText(this, message, Toast.LENGTH_SHORT).show();
    Intent intent1 = new Intent(ShopDisplay.this, ShopDisplay.class);
    intent1.putExtra("shopID", shop_id);
    startActivity(intent1);
}
```

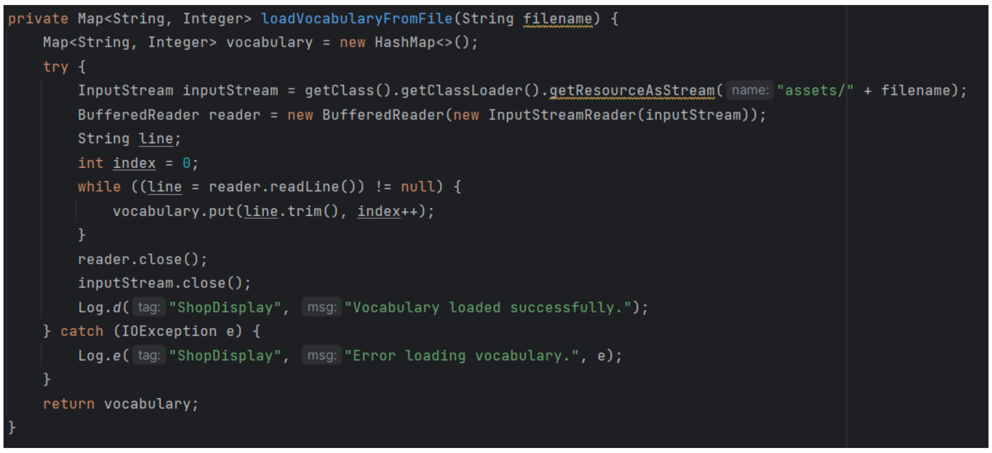
Figure 5-2-6-2-3-9 Code of Shop Display Activity _ Review and Rating Submit Function

According to Figure 5-2-6-2-3-10, this code loads the resources and the model in Android Studio, and calling the sentiment analysis function by passing the two data input which are rating and text.



```java
try {
    MappedByteBuffer tfliteModel = FileUtil.loadMappedFile(this, "rs4u_model_Q_V7.tflite");
    Map<String, Integer> vocab = loadVocabularyFromFile("vocab.txt");
    tflite = new Interpreter(tfliteModel);
    Log.i("SentimentValue", "The rating is (0,1,2): " + ratings);
    Log.i("SentimentValue", "The review or comment is : " + texts);
    SentimentAnalysis sentimentAnalysis = new SentimentAnalysis(tflite, vocab);
    if(ratings == 0 && texts != null){
        sentimentValue = sentimentAnalysis.analyzeSentiment(ratings, texts);
    }else{
        Log.i("SentimentValue", "The review or rating, one of them is empty: ");

    }
    //sentimentValue = sentimentAnalysis.analyzeSentiment(ratings, texts);
    Log.i("SentimentValue", "The result is : " + sentimentValue);
} catch (IOException e) {
    Log.e("tfliteException", "Error: couldn't load tflite model.", e);
}
```

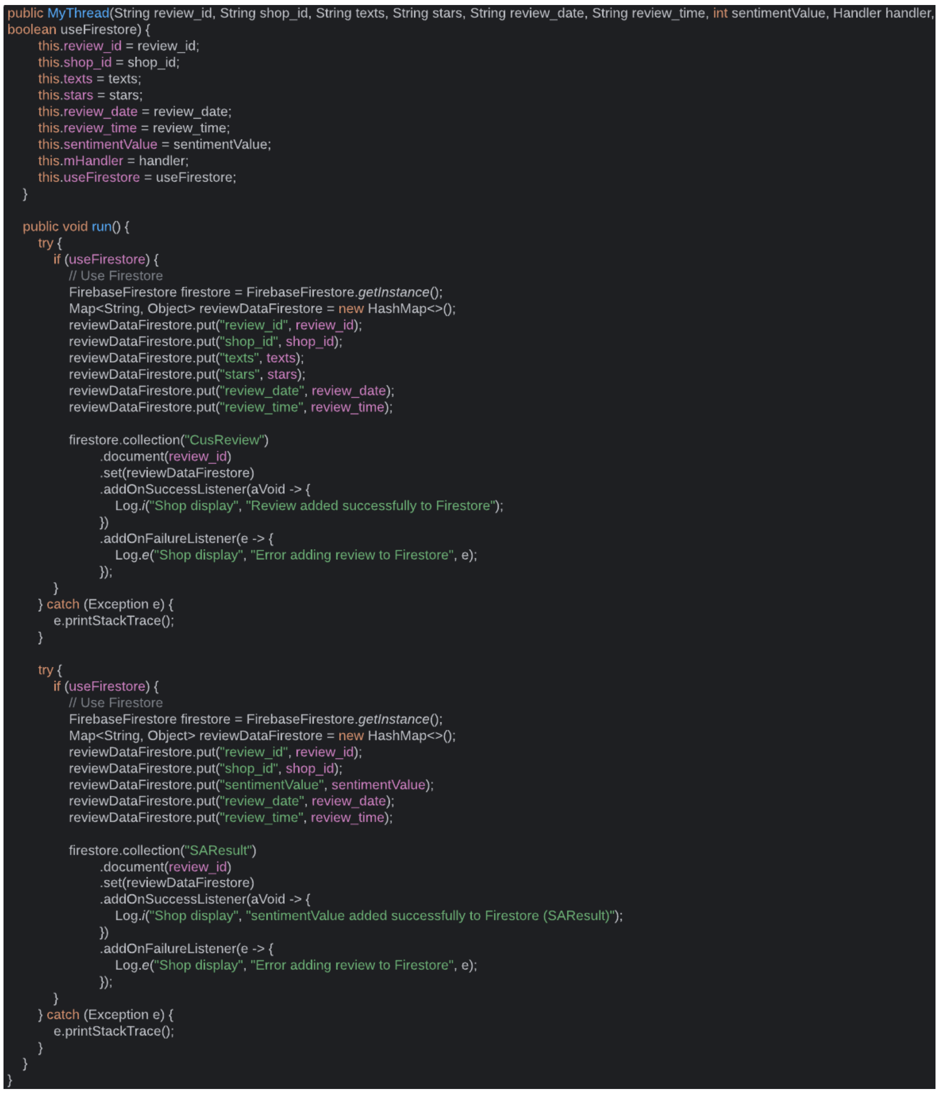
Figure 5-2-6-2-3-10 Code of Shop Display Activity _ Code of Performing Sentiment Analysis



```java
private Map<String, Integer> loadVocabularyFromFile(String filename) {
    Map<String, Integer> vocabulary = new HashMap<>();
    try {
        InputStream inputStream = getClass().getClassLoader().getResourceAsStream( name: "assets/" + filename);
        BufferedReader reader = new BufferedReader(new InputStreamReader(inputStream));
        String line;
        int index = 0;
        while ((line = reader.readLine()) != null) {
            vocabulary.put(line.trim(), index++);
        }
        reader.close();
        inputStream.close();
        Log.d( tag: "ShopDisplay",  msg: "Vocabulary loaded successfully.");
    } catch (IOException e) {
        Log.e( tag: "ShopDisplay",  msg: "Error loading vocabulary.", e);
    }
    return vocabulary;
}
```

Figure 5-2-6-2-3-11 Code of Shop Display Activity _ Model Loading Function

In this activity, the submit function will implement the sentiment analysis and save the result using the thread class. The sentiment analysis process is using the method from a class created out from the activity which will further discuss in next chapter. In this thread class, it save the data to the customer review table and sentiment analysis result table in the Firebase Firestore (Figure 5-2-6-2-3-12).

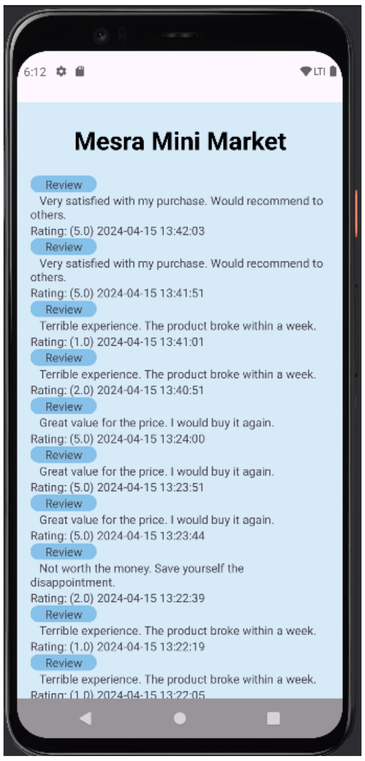


```java
public MyThread(String review_id, String shop_id, String texts, String stars, String review_date, String review_time, int sentimentValue, Handler handler,
boolean useFirestore) {
        this.review_id = review_id;
        this.shop_id = shop_id;
        this.texts = texts;
        this.stars = stars;
        this.review_date = review_date;
        this.review_time = review_time;
        this.sentimentValue = sentimentValue;
        this.mHandler = handler;
        this.useFirestore = useFirestore;
    }

    public void run() {
        try {
            if (useFirestore) {
                // Use Firestore
                FirebaseFirestore firestore = FirebaseFirestore.getInstance();
                Map<String, Object> reviewDataFirestore = new HashMap<>();
                reviewDataFirestore.put("review_id", review_id);
                reviewDataFirestore.put("shop_id", shop_id);
                reviewDataFirestore.put("texts", texts);
                reviewDataFirestore.put("stars", stars);
                reviewDataFirestore.put("review_date", review_date);
                reviewDataFirestore.put("review_time", review_time);

                firestore.collection("CusReview")
                        .document(review_id)
                        .set(reviewDataFirestore)
                        .addOnSuccessListener(aVoid -> {
                            Log.i("Shop display", "Review added successfully to Firestore");
                        })
                        .addOnFailureListener(e -> {
                            Log.e("Shop display", "Error adding review to Firestore", e);
                        });
            }
        } catch (Exception e) {
            e.printStackTrace();
        }

        try {
            if (useFirestore) {
                // Use Firestore
                FirebaseFirestore firestore = FirebaseFirestore.getInstance();
                Map<String, Object> reviewDataFirestore = new HashMap<>();
                reviewDataFirestore.put("review_id", review_id);
                reviewDataFirestore.put("shop_id", shop_id);
                reviewDataFirestore.put("sentimentValue", sentimentValue);
                reviewDataFirestore.put("review_date", review_date);
                reviewDataFirestore.put("review_time", review_time);

                firestore.collection("SAResult")
                        .document(review_id)
                        .set(reviewDataFirestore)
                        .addOnSuccessListener(aVoid -> {
                            Log.i("Shop display", "sentimentValue added successfully to Firestore (SAResult)");
                        })
                        .addOnFailureListener(e -> {
                            Log.e("Shop display", "Error adding review to Firestore", e);
                        });
            }
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

Figure 5-2-6-2-3-12 Code of Shop Display Activity _ Thread Class

5.2.6.2.4 Shop Review Display

In this activity page, it will display the review with the order based on the date and time. Figure 5-2-6-2-4-1 shows the interface of the page developed. Which have some difference with the draft design of interface (Figure 4-3-4). To display the reviews of a shop, there are four method or function is created. First, default function named "onCreate", second function for retrieving data, third function for displaying the shop's review it the shop have any review, last function to display the shop that do not have any review or rating.



Mesra Mini Market

Review
Very satisfied with my purchase. Would recommend to others.
Rating: (5.0) 2024-04-15 13:42:03
Review
Very satisfied with my purchase. Would recommend to others.
Rating: (5.0) 2024-04-15 13:41:51
Review
Terrible experience. The product broke within a week.
Rating: (1.0) 2024-04-15 13:41:01
Review
Terrible experience. The product broke within a week.
Rating: (2.0) 2024-04-15 13:40:51
Review
Great value for the price. I would buy it again.
Rating: (5.0) 2024-04-15 13:24:00
Review
Great value for the price. I would buy it again.
Rating: (5.0) 2024-04-15 13:23:51
Review
Great value for the price. I would buy it again.
Rating: (5.0) 2024-04-15 13:23:44
Review
Not worth the money. Save yourself the disappointment.
Rating: (2.0) 2024-04-15 13:22:39
Review
Terrible experience. The product broke within a week.
Rating: (1.0) 2024-04-15 13:22:19
Review
Terrible experience. The product broke within a week.
Rating: (1.0) 2024-04-15 13:22:05

Figure 5-2-6-2-4-1 Application Interface: Shop Review Display Activity

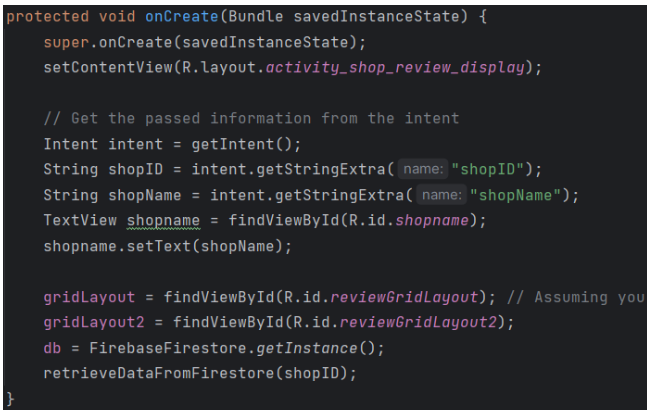
```
<Text View> Shop Name </Text View>

<Scroll View>
    <Linear Layout>
        <Grid Layout> Shops with review and rating data </Grid Layout>
        <Grid Layout> Shops without review and rating data </Grid Layout>
    </Linear Layout>
</Scroll View>
```

Figure 5-2-6-2-4-2 Simplified Version of Interface (XML) of Shop Review Display Activity

Variable Name	**Variable Data Type**	**Purpose**
db	FirebaseFirestore	To access the Firebase Firestore and retrieve data saved.
girdLayout	Grid Layout	To display the shop's review in separate layout.
girdLayout2	Gird Layout	
rowCount	Integer	To numbering and calculate number of grid layout per review and rating
rowCount2	Integer	

Table 5-2-6-2-4-1 Shop Review Display Code: Variables Creation

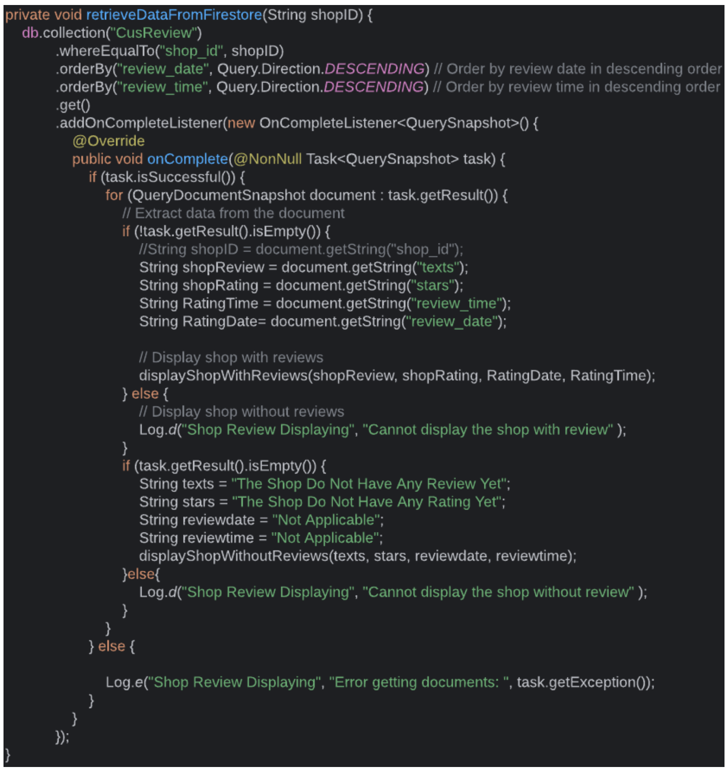
Figure 5-2-6-2-4-3 show the code of the default method. In this method, it mainly retrieves the data that passed from previous activity, link the element created in XML with the variable created in the activity, initialized the Firebase, and called the retrieving data function.



```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_shop_review_display);

    // Get the passed information from the intent
    Intent intent = getIntent();
    String shopID = intent.getStringExtra( name: "shopID");
    String shopName = intent.getStringExtra( name: "shopName");
    TextView shopname = findViewById(R.id.shopname);
    shopname.setText(shopName);

    gridLayout = findViewById(R.id.reviewGridLayout); // Assuming you
    gridLayout2 = findViewById(R.id.reviewGridLayout2);
    db = FirebaseFirestore.getInstance();
    retrieveDataFromFirestore(shopID);
}
```

Figure 5-2-6-2-4-3 Code of Shop Review Display Activity _ onCreate Method

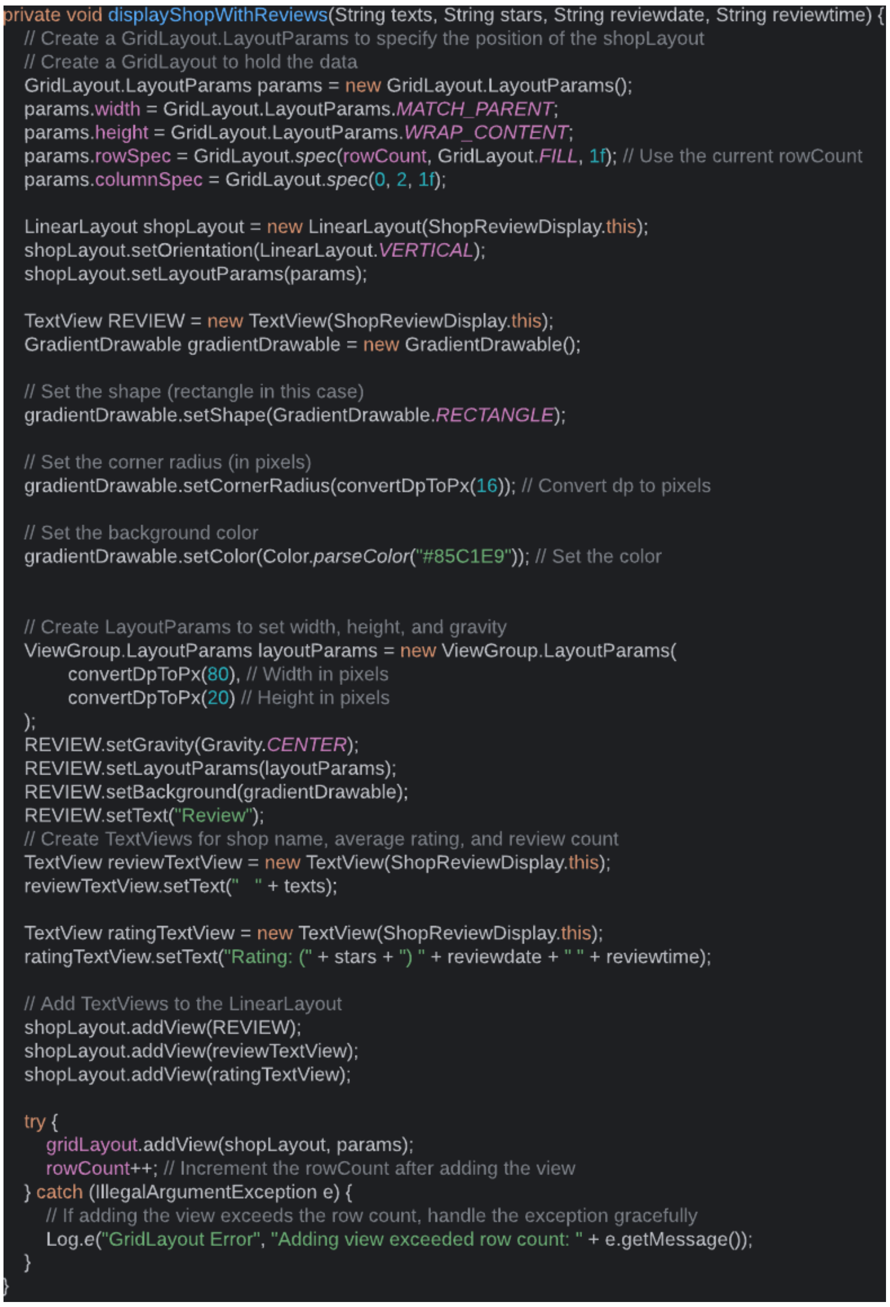
Figure 5-2-6-2-4-4 show the date retrieving function from the customer review table in Firebase Firestore. The reviews displayed have a requirement where it display according to the date and time. To achieve that, “orderBy” is used. Similar to previous activity, to retrieve the data, it use the “get” function, and to set the value to the text view use the “set” function.



```
private void retrieveDataFromFirestore(String shopID) {
    db.collection("CusReview")
            .whereEqualTo("shop_id", shopID)
            .orderBy("review_date", Query.Direction.DESCENDING) // Order by review date in descending order
            .orderBy("review_time", Query.Direction.DESCENDING) // Order by review time in descending order
            .get()
            .addOnCompleteListener(new OnCompleteListener<QuerySnapshot>() {
                @Override
                public void onComplete(@NonNull Task<QuerySnapshot> task) {
                    if (task.isSuccessful()) {
                        for (QueryDocumentSnapshot document : task.getResult()) {
                            // Extract data from the document
                            if (!task.getResult().isEmpty()) {
                                //String shopID = document.getString("shop_id");
                                String shopReview = document.getString("texts");
                                String shopRating = document.getString("stars");
                                String RatingTime = document.getString("review_time");
                                String RatingDate= document.getString("review_date");

                                // Display shop with reviews
                                displayShopWithReviews(shopReview, shopRating, RatingDate, RatingTime);
                            } else {
                                // Display shop without reviews
                                Log.d("Shop Review Displaying", "Cannot display the shop with review" );
                            }
                            if (task.getResult().isEmpty()) {
                                String texts = "The Shop Do Not Have Any Review Yet";
                                String stars = "The Shop Do Not Have Any Rating Yet";
                                String reviewdate = "Not Applicable";
                                String reviewtime = "Not Applicable";
                                displayShopWithoutReviews(texts, stars, reviewdate, reviewtime);
                            }else{
                                Log.d("Shop Review Displaying", "Cannot display the shop without review" );
                            }
                        }
                    } else {

                        Log.e("Shop Review Displaying", "Error getting documents: ", task.getException());
                    }
                }
            });
}
```

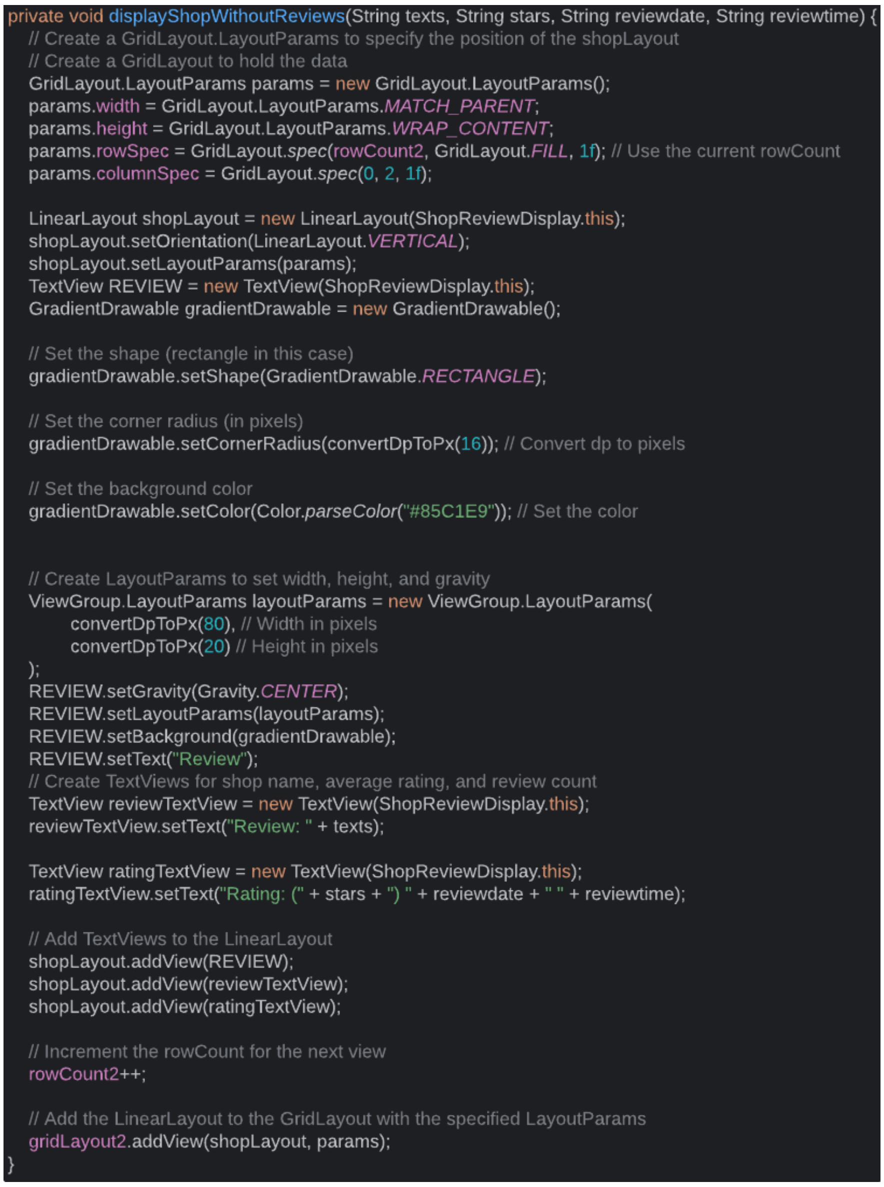
Figure 5-2-6-2-4-4 Code of Shop Review Display Activity _ Retrieving Data Method

Figure 5-2-6-2-4-5 and Figure 5-2-6-2-4-6 show the ways on displaying the reviews. The logic applied is similar with the explanation in Chapter 5-2-6-2-1 of the Figure 5-2-6-2-1-18 to Figure 5-2-6-2-1-20. The function show in Figure 5-2-6-1-21 used in the activity as well.



```java
private void displayShopWithReviews(String texts, String stars, String reviewdate, String reviewtime) {
    // Create a GridLayout.LayoutParams to specify the position of the shopLayout
    // Create a GridLayout to hold the data
    GridLayout.LayoutParams params = new GridLayout.LayoutParams();
    params.width = GridLayout.LayoutParams.MATCH_PARENT;
    params.height = GridLayout.LayoutParams.WRAP_CONTENT;
    params.rowSpec = GridLayout.spec(rowCount, GridLayout.FILL, 1f); // Use the current rowCount
    params.columnSpec = GridLayout.spec(0, 2, 1f);

    LinearLayout shopLayout = new LinearLayout(ShopReviewDisplay.this);
    shopLayout.setOrientation(LinearLayout.VERTICAL);
    shopLayout.setLayoutParams(params);

    TextView REVIEW = new TextView(ShopReviewDisplay.this);
    GradientDrawable gradientDrawable = new GradientDrawable();

    // Set the shape (rectangle in this case)
    gradientDrawable.setShape(GradientDrawable.RECTANGLE);

    // Set the corner radius (in pixels)
    gradientDrawable.setCornerRadius(convertDpToPx(16)); // Convert dp to pixels

    // Set the background color
    gradientDrawable.setColor(Color.parseColor("#85C1E9")); // Set the color


    // Create LayoutParams to set width, height, and gravity
    ViewGroup.LayoutParams layoutParams = new ViewGroup.LayoutParams(
            convertDpToPx(80), // Width in pixels
            convertDpToPx(20) // Height in pixels
    );
    REVIEW.setGravity(Gravity.CENTER);
    REVIEW.setLayoutParams(layoutParams);
    REVIEW.setBackground(gradientDrawable);
    REVIEW.setText("Review");
    // Create TextViews for shop name, average rating, and review count
    TextView reviewTextView = new TextView(ShopReviewDisplay.this);
    reviewTextView.setText("   " + texts);

    TextView ratingTextView = new TextView(ShopReviewDisplay.this);
    ratingTextView.setText("Rating: (" + stars + ") " + reviewdate + " " + reviewtime);

    // Add TextViews to the LinearLayout
    shopLayout.addView(REVIEW);
    shopLayout.addView(reviewTextView);
    shopLayout.addView(ratingTextView);

    try {
        gridLayout.addView(shopLayout, params);
        rowCount++; // Increment the rowCount after adding the view
    } catch (IllegalArgumentException e) {
        // If adding the view exceeds the row count, handle the exception gracefully
        Log.e("GridLayout Error", "Adding view exceeded row count: " + e.getMessage());
    }
}
```

Figure 5-2-6-2-4-5 Code of Shop Review Display Activity _ Display Shop Review Method One

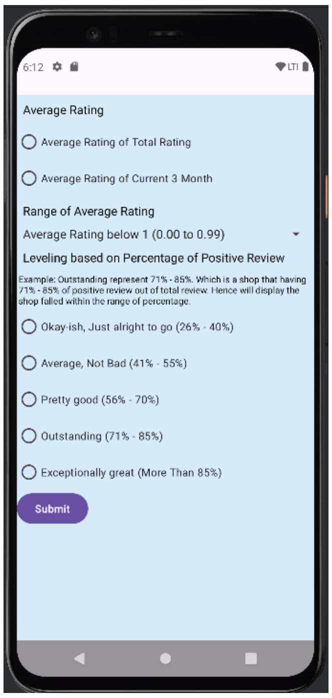


```
private void displayShopWithoutReviews(String texts, String stars, String reviewdate, String reviewtime) {
    // Create a GridLayout.LayoutParams to specify the position of the shopLayout
    // Create a GridLayout to hold the data
    GridLayout.LayoutParams params = new GridLayout.LayoutParams();
    params.width = GridLayout.LayoutParams.MATCH_PARENT;
    params.height = GridLayout.LayoutParams.WRAP_CONTENT;
    params.rowSpec = GridLayout.spec(rowCount2, GridLayout.FILL, 1f); // Use the current rowCount
    params.columnSpec = GridLayout.spec(0, 2, 1f);

    LinearLayout shopLayout = new LinearLayout(ShopReviewDisplay.this);
    shopLayout.setOrientation(LinearLayout.VERTICAL);
    shopLayout.setLayoutParams(params);
    TextView REVIEW = new TextView(ShopReviewDisplay.this);
    GradientDrawable gradientDrawable = new GradientDrawable();

    // Set the shape (rectangle in this case)
    gradientDrawable.setShape(GradientDrawable.RECTANGLE);

    // Set the corner radius (in pixels)
    gradientDrawable.setCornerRadius(convertDpToPx(16)); // Convert dp to pixels

    // Set the background color
    gradientDrawable.setColor(Color.parseColor("#85C1E9")); // Set the color


    // Create LayoutParams to set width, height, and gravity
    ViewGroup.LayoutParams layoutParams = new ViewGroup.LayoutParams(
            convertDpToPx(80), // Width in pixels
            convertDpToPx(20) // Height in pixels
    );
    REVIEW.setGravity(Gravity.CENTER);
    REVIEW.setLayoutParams(layoutParams);
    REVIEW.setBackground(gradientDrawable);
    REVIEW.setText("Review");
    // Create TextViews for shop name, average rating, and review count
    TextView reviewTextView = new TextView(ShopReviewDisplay.this);
    reviewTextView.setText("Review: " + texts);

    TextView ratingTextView = new TextView(ShopReviewDisplay.this);
    ratingTextView.setText("Rating: (" + stars + ") " + reviewdate + " " + reviewtime);

    // Add TextViews to the LinearLayout
    shopLayout.addView(REVIEW);
    shopLayout.addView(reviewTextView);
    shopLayout.addView(ratingTextView);

    // Increment the rowCount for the next view
    rowCount2++;

    // Add the LinearLayout to the GridLayout with the specified LayoutParams
    gridLayout2.addView(shopLayout, params);
}
```

Figure 5-2-6-2-4-6 Code of Shop Review Display Activity _ Display Shop Review Method Two

5.2.6.2.5 Filter

The filter page provided mainly two criteria of filtering or ranking to perform recommendation. However, it came with difference comparing to the initial implementation planning stated in Chapter 4 (Figure 4-3-5). Firstly, it will utilize the average rating. The application user can choose either to calculate the average rating from all rating, or only current 3 months of the rating. After choosing it, the user can choose the range of rating they would like to get recommended shops. For example, rating between 2 to 2.99. Next, the second criteria is based on the sentiment analysis result. According to the objective two of the project. It is expected to utilize the result of sentiment analysis by categorise the shop. For example, according to the percentage of shop getting positive review, the percentage is categorized into 5 different key words or phrase. For example, percentage more than or equal to 86%.


Average Rating

◯ Average Rating of Total Rating

◯ Average Rating of Current 3 Month

Range of Average Rating

Average Rating below 1 (0.00 to 0.99)

Leveling based on Percentage of Positive Review

Example: Outstanding represent 71% - 85%. Which is a shop that having 71% - 85% of positive review out of total review. Hence will display the shop falled within the range of percentage.

◯ Okay-ish, Just alright to go (26% - 40%)

◯ Average, Not Bad (41% - 55%)

◯ Pretty good (56% - 70%)

◯ Outstanding (71% - 85%)

◯ Exceptionally great (More Than 85%)

Figure 5-2-6-2-5-1 Application Interface: Filter Activity

```
<Linear Layout>
     <Text View> Average Rating </Text View>
     <Radio Group>
          <Radio Button> Average Rating of Total Rating </Radio Button>
          <Radio Button> Average Rating of Current 3 Month </Radio Button>
     </Radio Group>
     <Text View> Range of Average Rating</Text View>
     <Spinner> Display Drop Down List of 5 option </Spinner>
     <Text View> Leveling based on Percentage of Positive Review </Text View>
     <Text View> Description Sentences </Text View>
     <Radio Group>
          <Radio Button> Okay-ish, Just alright to go (26% - 40%) </Radio Button>
          <Radio Button> Average, Not Bad (41% - 55%) </Radio Button>
          <Radio Button> Pretty good (56% - 70%) </Radio Button>
          <Radio Button> Outstanding (71% - 85%) </Radio Button>
          <Radio Button> Exceptionally great (More Than 85%) </Radio Button>
     </Radio Group>
     <Button> Submit </Button>
</Linear Layout>
```

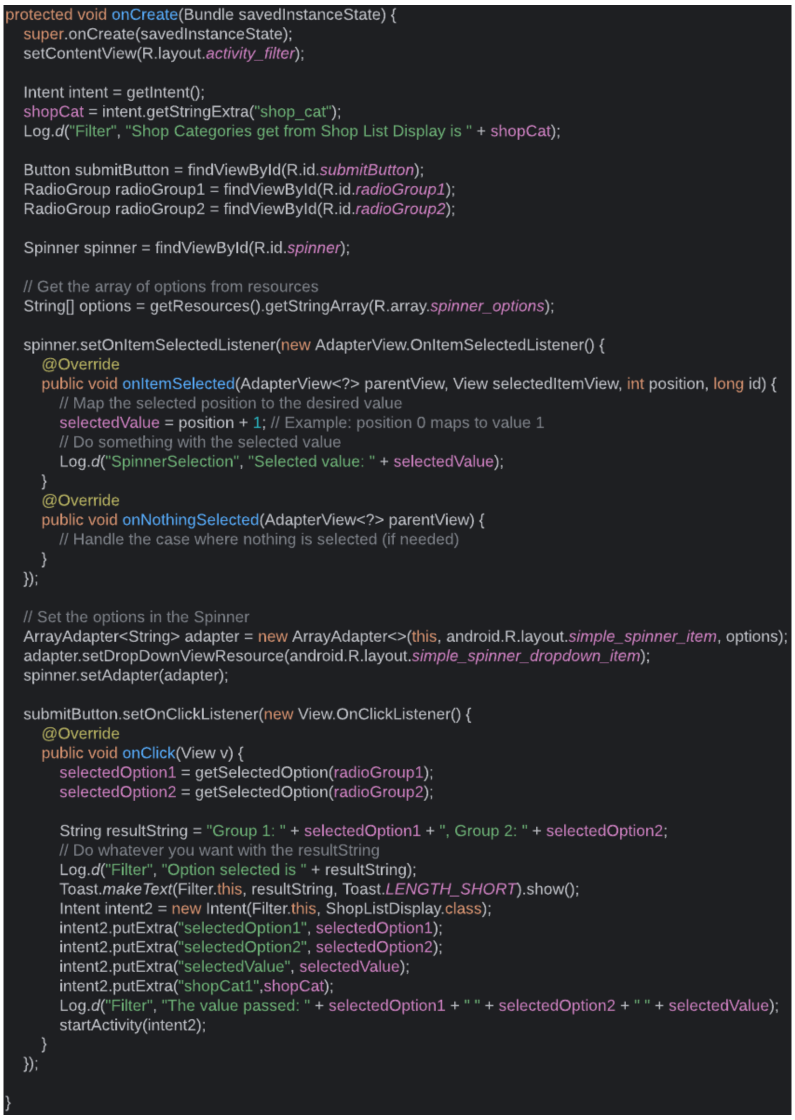
Figure 5-2-6-2-5-2 Simplified Version of Interface (XML) of Filter Activity

In this activity, there is mainly two functions. Default Function and numbering function to track the option chosen. There are some variables created that present in Table 5-3-6-2-5-1.

Variable Name	**Variable Data Type**	**Purpose**
selectedValue	Integer	To score the choice that user choose from two radio button group
selectedOption1	Integer	
selectedUOption2	Integer	
shopCat	String	To save the category of shop.

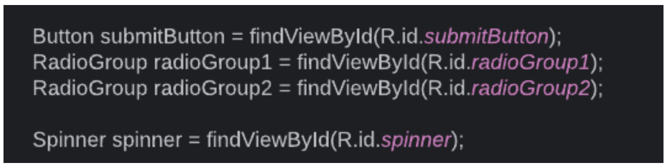
Table 5-2-6-2-5-1 Filter Code: Variables Creation

Figure 5-2-6-2-5-3 show the complete code of on create method. In this section, there will be a further explanation for Figure 5-2-6-2-5-3 that split into Figure 5-2-6-2-5-4 to Figure 5-2-6-2-5-7. Firstly, Figure 5-2-6-2-5-4 show the code of linking the elements created in XML with the variable create in activity. Next, there are two option of radio button in radio group one. A drop-down list is created using spinner which provide five option that stated in Figure 5-2-6-2-5-6. And the options will be set into an array created in the activity, and a selected listener is set for tracking the option chosen and store into the variable named “selectedValue”. And set the options to the layout created in XML. Lastly, a submit listener is created to track the option chosen by using the method shown in Figure 5-2-6-2-5-8. At last, it will pass the data to the next activity which is Shop List Display Activity to perform recommendation by filtering and ranking.



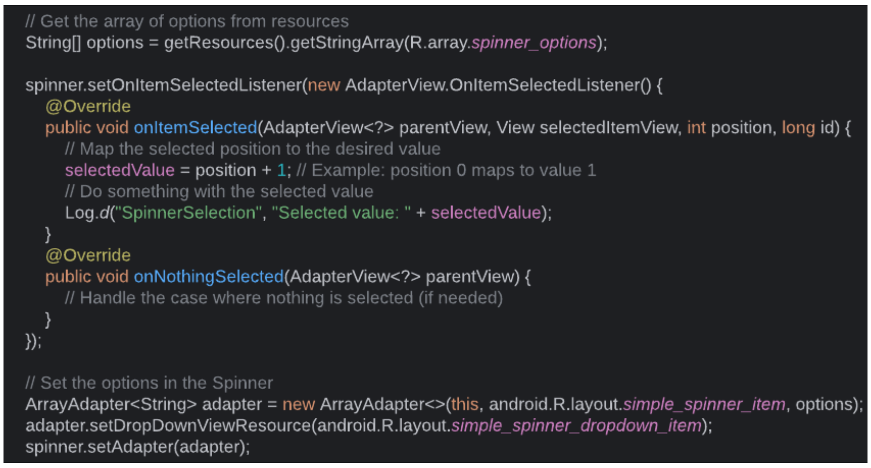
```
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_filter);

    Intent intent = getIntent();
    shopCat = intent.getStringExtra("shop_cat");
    Log.d("Filter", "Shop Categories get from Shop List Display is " + shopCat);

    Button submitButton = findViewById(R.id.submitButton);
    RadioGroup radioGroup1 = findViewById(R.id.radioGroup1);
    RadioGroup radioGroup2 = findViewById(R.id.radioGroup2);

    Spinner spinner = findViewById(R.id.spinner);

    // Get the array of options from resources
    String[] options = getResources().getStringArray(R.array.spinner_options);

    spinner.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
        @Override
        public void onItemSelected(AdapterView<?> parentView, View selectedItemView, int position, long id) {
            // Map the selected position to the desired value
            selectedValue = position + 1; // Example: position 0 maps to value 1
            // Do something with the selected value
            Log.d("SpinnerSelection", "Selected value: " + selectedValue);
        }
        @Override
        public void onNothingSelected(AdapterView<?> parentView) {
            // Handle the case where nothing is selected (if needed)
        }
    });

    // Set the options in the Spinner
    ArrayAdapter<String> adapter = new ArrayAdapter<>(this, android.R.layout.simple_spinner_item, options);
    adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
    spinner.setAdapter(adapter);

    submitButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            selectedOption1 = getSelectedOption(radioGroup1);
            selectedOption2 = getSelectedOption(radioGroup2);

            String resultString = "Group 1: " + selectedOption1 + ", Group 2: " + selectedOption2;
            // Do whatever you want with the resultString
            Log.d("Filter", "Option selected is " + resultString);
            Toast.makeText(Filter.this, resultString, Toast.LENGTH_SHORT).show();
            Intent intent2 = new Intent(Filter.this, ShopListDisplay.class);
            intent2.putExtra("selectedOption1", selectedOption1);
            intent2.putExtra("selectedOption2", selectedOption2);
            intent2.putExtra("selectedValue", selectedValue);
            intent2.putExtra("shopCat1",shopCat);
            Log.d("Filter", "The value passed: " + selectedOption1 + " " + selectedOption2 + " " + selectedValue);
            startActivity(intent2);
        }
    });

}
```

Figure 5-2-6-2-5-3 Code of Filter Activity _ onCreate Method



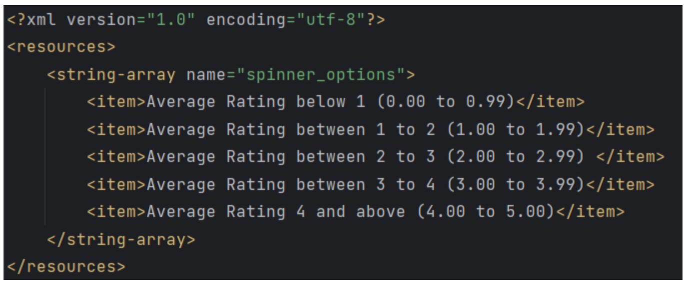
```
Button submitButton = findViewById(R.id.submitButton);
RadioGroup radioGroup1 = findViewById(R.id.radioGroup1);
RadioGroup radioGroup2 = findViewById(R.id.radioGroup2);

Spinner spinner = findViewById(R.id.spinner);
```

Figure 5-2-6-2-5-4 Code of Filter Activity _ onCreate Method Part 1



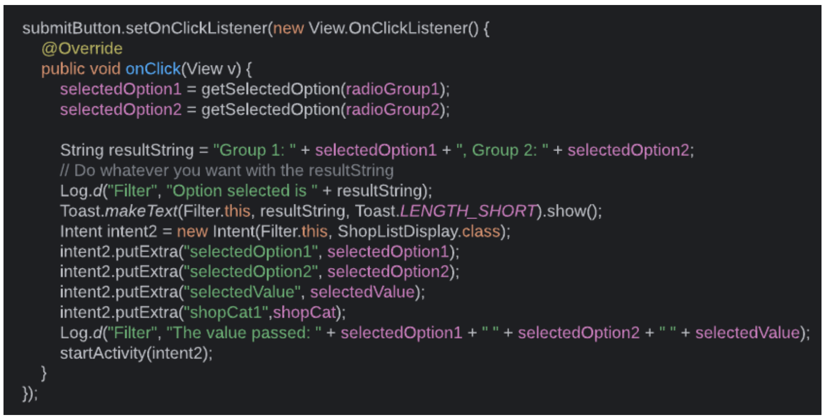
```
// Get the array of options from resources
String[] options = getResources().getStringArray(R.array.spinner_options);

spinner.setOnItemSelectedListener(new AdapterView.OnItemSelectedListener() {
    @Override
    public void onItemSelected(AdapterView<?> parentView, View selectedItemView, int position, long id) {
        // Map the selected position to the desired value
        selectedValue = position + 1; // Example: position 0 maps to value 1
        // Do something with the selected value
        Log.d("SpinnerSelection", "Selected value: " + selectedValue);
    }
    @Override
    public void onNothingSelected(AdapterView<?> parentView) {
        // Handle the case where nothing is selected (if needed)
    }
});

// Set the options in the Spinner
ArrayAdapter<String> adapter = new ArrayAdapter<>(this, android.R.layout.simple_spinner_item, options);
adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
spinner.setAdapter(adapter);
```

Figure 5-2-6-2-5-5 Code of Filter Activity _ onCreate Method Part 2



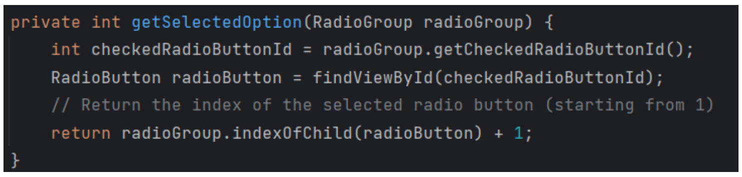
```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string-array name="spinner_options">
        <item>Average Rating below 1 (0.00 to 0.99)</item>
        <item>Average Rating between 1 to 2 (1.00 to 1.99)</item>
        <item>Average Rating between 2 to 3 (2.00 to 2.99) </item>
        <item>Average Rating between 3 to 4 (3.00 to 3.99)</item>
        <item>Average Rating 4 and above (4.00 to 5.00)</item>
    </string-array>
</resources>
```

Figure 5-2-6-2-5-6 Code of Filter Activity _ Drop Down List Option



```
submitButton.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        selectedOption1 = getSelectedOption(radioGroup1);
        selectedOption2 = getSelectedOption(radioGroup2);

        String resultString = "Group 1: " + selectedOption1 + ", Group 2: " + selectedOption2;
        // Do whatever you want with the resultString
        Log.d("Filter", "Option selected is " + resultString);
        Toast.makeText(Filter.this, resultString, Toast.LENGTH_SHORT).show();
        Intent intent2 = new Intent(Filter.this, ShopListDisplay.class);
        intent2.putExtra("selectedOption1", selectedOption1);
        intent2.putExtra("selectedOption2", selectedOption2);
        intent2.putExtra("selectedValue", selectedValue);
        intent2.putExtra("shopCat1",shopCat);
        Log.d("Filter", "The value passed: " + selectedOption1 + " " + selectedOption2 + " " + selectedValue);
        startActivity(intent2);
    }
});
```

Figure 5-2-6-2-5-7 Code of Filter Activity _ onCreate Method Part 3



```
private int getSelectedOption(RadioGroup radioGroup) {
    int checkedRadioButtonId = radioGroup.getCheckedRadioButtonId();
    RadioButton radioButton = findViewById(checkedRadioButtonId);
    // Return the index of the selected radio button (starting from 1)
    return radioGroup.indexOfChild(radioButton) + 1;
}
```

Figure 5-2-6-2-5-8 Code of Filter Activity _ Option Tracking Method

At last, the filter feature provides a fundamental option for the user to choose. Table 5-2-6-2-5-2. Which mainly based on average rating and percentage of a shop getting positive review and rating.

Options	**Descriptions**
Average rating with all data	The average rating calculates using all data.
Average rating with three-month data	The average rating calculates using the data within three months.
5 Range of average rating	1st = Average rating between 0 – 0.99 2nd= Average rating between 1 – 1.99 3rd= Average rating between 2 – 2.99 4th= Average rating between 3 – 3.99 5th= Average rating between 4 – 5
5 Level of percentage for a shop getting positive review and rating among all review and rating.	1st = Okay-ish, Just alright to go; between 26% – 40% 2nd= Average, Not Bad; between 41% - 55% 3rd= Pretty good; between56% - 70% 4th= Outstanding; between 71% - 85% 5th= Exceptionally great; between 86% - 100%

Table 5-2-6-2-5-2 Fundamental Options for Filtering and Ranking

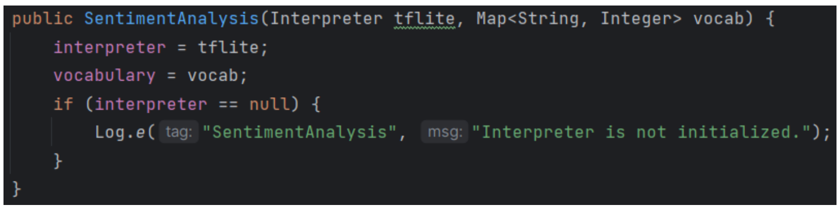
5.2.6.2.6 Sentiment Analysis Class

According to the implementation planning in Chapter 4, it is expected to direct used the model and libraries imported to perform sentiment analysis. However, during the development phases, due to the Android Studio version restriction, it is not able to import the main and key library which is BERT and its tokenizer. Hence, it causes the model usage in the application different in terms of data preprocessing. Therefore, the preprocessing needed to perform in manually by downloading the BERT tokenizer and perform tokenization manually in the class. And, this class create for implementation of sentiment analysis. By calling the function in the class from other activity, the sentiment analysis can be performed. In this project, the sentiment analysis process begins when user submit their review and rating. Which simultaneously saving the data into Firebase Firestore. In the Shop Display Activity, the model and its resources are being load which explain in Chapter 5.2.6.2.3. In Table 5-2-6-2-6-1, it shows three variables created.

Variables	**Data Types**
MAX_SEQ_LENGTH	Integer
vocabulary	Map<String, Integer>
interpreter	Interpreter
inputIds	Array List
attentionMask	Array List

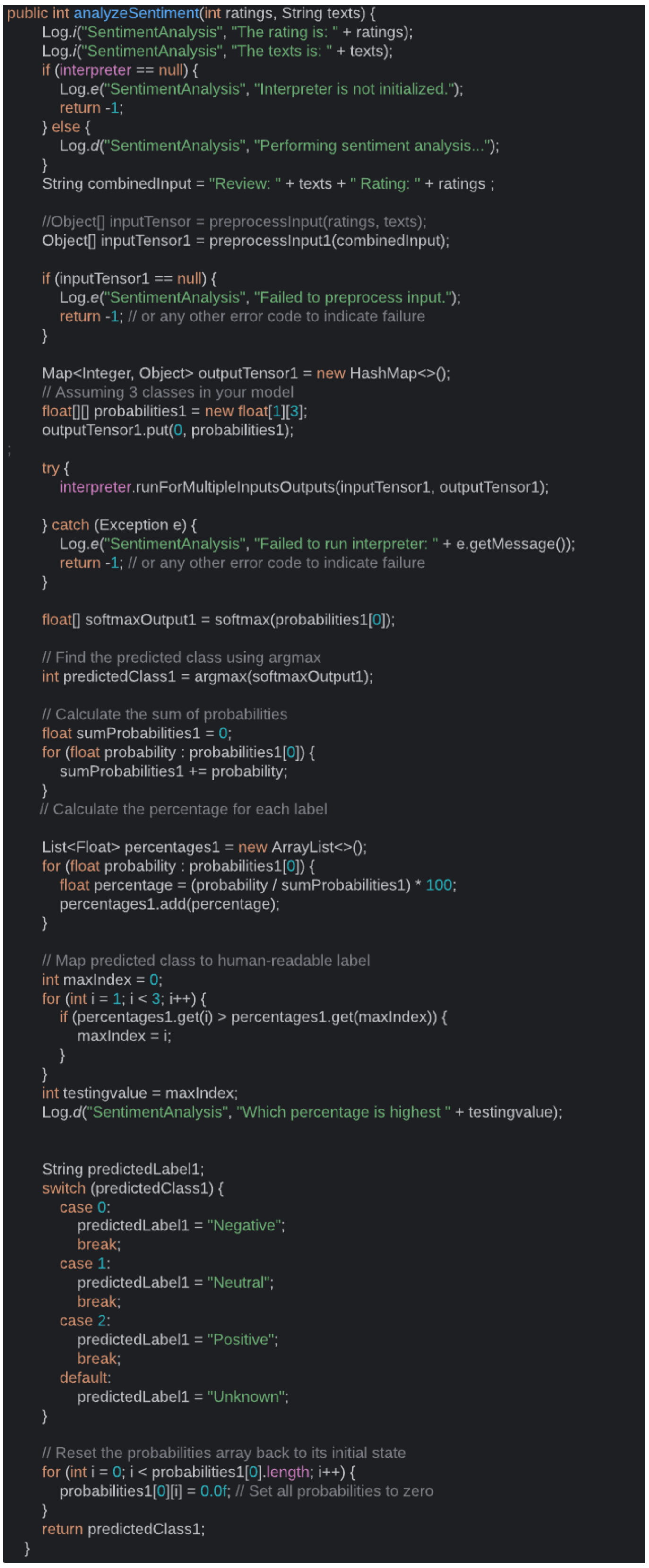
Table 5-2-6-2-6-1 Sentiment Analysis Class Code: Variables Creation

In sentiment analysis class, there are mainly four functions supporting each other for sentiment analysis. Figure 5-2-6-2-6-1 show the classes object which having two input such as interpreter and map. Figure 5-2-6-2-6-2 show the complete code of main function that analyses the sentiment of customer review and rating. The code in Figure 5-2-6-2-6-2 can be separated to five parts from Figure5-2-6-2-6-3 to Figure 5-2-6-2-6-7. Explanation and discussion are provided by figure below.



```
public SentimentAnalysis(Interpreter tflite, Map<String, Integer> vocab) {
    interpreter = tflite;
    vocabulary = vocab;
    if (interpreter == null) {
        Log.e( tag: "SentimentAnalysis",  msg: "Interpreter is not initialized.");
    }
}
```

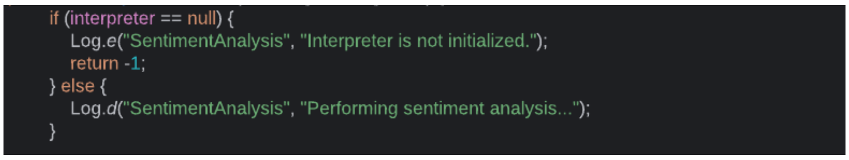
Figure 5-2-6-2-6-1 Code of Sentiment Analysis Class _ Sentiment Analysis



```java
public int analyzeSentiment(int ratings, String texts) {
    Log.i("SentimentAnalysis", "The rating is: " + ratings);
    Log.i("SentimentAnalysis", "The texts is: " + texts);
    if (interpreter == null) {
        Log.e("SentimentAnalysis", "Interpreter is not initialized.");
        return -1;
    } else {
        Log.d("SentimentAnalysis", "Performing sentiment analysis...");
    }
    String combinedInput = "Review: " + texts + " Rating: " + ratings ;

    //Object[] inputTensor = preprocessInput(ratings, texts);
    Object[] inputTensor1 = preprocessInput1(combinedInput);

    if (inputTensor1 == null) {
        Log.e("SentimentAnalysis", "Failed to preprocess input.");
        return -1; // or any other error code to indicate failure
    }

    Map<Integer, Object> outputTensor1 = new HashMap<>();
    // Assuming 3 classes in your model
    float[][] probabilities1 = new float[1][3];
    outputTensor1.put(0, probabilities1);
;
    try {
        interpreter.runForMultipleInputsOutputs(inputTensor1, outputTensor1);

    } catch (Exception e) {
        Log.e("SentimentAnalysis", "Failed to run interpreter: " + e.getMessage());
        return -1; // or any other error code to indicate failure
    }

    float[] softmaxOutput1 = softmax(probabilities1[0]);

    // Find the predicted class using argmax
    int predictedClass1 = argmax(softmaxOutput1);

    // Calculate the sum of probabilities
    float sumProbabilities1 = 0;
    for (float probability : probabilities1[0]) {
        sumProbabilities1 += probability;
    }
    // Calculate the percentage for each label

    List<Float> percentages1 = new ArrayList<>();
    for (float probability : probabilities1[0]) {
        float percentage = (probability / sumProbabilities1) * 100;
        percentages1.add(percentage);
    }

    // Map predicted class to human-readable label
    int maxIndex = 0;
    for (int i = 1; i < 3; i++) {
        if (percentages1.get(i) > percentages1.get(maxIndex)) {
            maxIndex = i;
        }
    }
    int testingvalue = maxIndex;
    Log.d("SentimentAnalysis", "Which percentage is highest " + testingvalue);


    String predictedLabel1;
    switch (predictedClass1) {
        case 0:
            predictedLabel1 = "Negative";
            break;
        case 1:
            predictedLabel1 = "Neutral";
            break;
        case 2:
            predictedLabel1 = "Positive";
            break;
        default:
            predictedLabel1 = "Unknown";
    }

    // Reset the probabilities array back to its initial state
    for (int i = 0; i < probabilities1[0].length; i++) {
        probabilities1[0][i] = 0.0f; // Set all probabilities to zero
    }
    return predictedClass1;
  }
```

Figure 5-2-6-2-6-2 Code of Sentiment Analysis Class _ Analyze Sentiment Function Complete Version

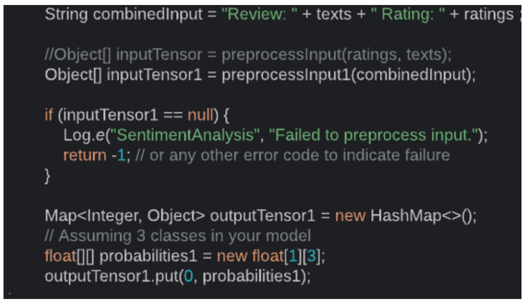
In this function, Figure 5-2-6-2-6-3 show the first element, which is to ensure the interpreter is not null. It is important as after loading the model saved, it is assigned to interpreter and used the model by using interpreter. If the model is not loaded successfully, it will return a value of negative one for the result of sentiment analysis.



```
if (interpreter == null) {
    Log.e("SentimentAnalysis", "Interpreter is not initialized.");
    return -1;
} else {
    Log.d("SentimentAnalysis", "Performing sentiment analysis...");
}
```

Figure 5-2-6-2-6-3 Code of Sentiment Analysis Class _ Analyze Sentiment Function Part 1

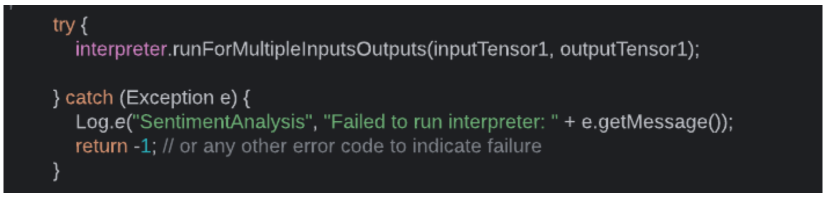
Next, to perform sentiment analysis, two data input must be prepared. Which are input which represent the review and rating by combining together, and output represent the percentage of the three label which are negative, neutral, and positive. To prepare the input tensor, the data inputs from user must be pre-processed by a function created show in Figure 5-2-6-2-6-8. To pass these two data input to the model used, the input must not be null. Therefore, if it is null, it will return negative one as the sentiment analysis result. One of the reasons can cause getting value of negative one is the user input is having a size of over the maximum size set during model development. It is also due to the reason of the input is exceeding the max length set. Hence, it not able to perform sentiment analysis (Figure 5-2-6-2-6-4).



```
String combinedInput = "Review: " + texts + " Rating: " + ratings

//Object[] inputTensor = preprocessInput(ratings, texts);
Object[] inputTensor1 = preprocessInput1(combinedInput);

if (inputTensor1 == null) {
    Log.e("SentimentAnalysis", "Failed to preprocess input.");
    return -1; // or any other error code to indicate failure
}

Map<Integer, Object> outputTensor1 = new HashMap<>();
// Assuming 3 classes in your model
float[][] probabilities1 = new float[1][3];
outputTensor1.put(0, probabilities1);
```

Figure 5-2-6-2-6-4 Code of Sentiment Analysis Class _ Analyze Sentiment Function Part 2

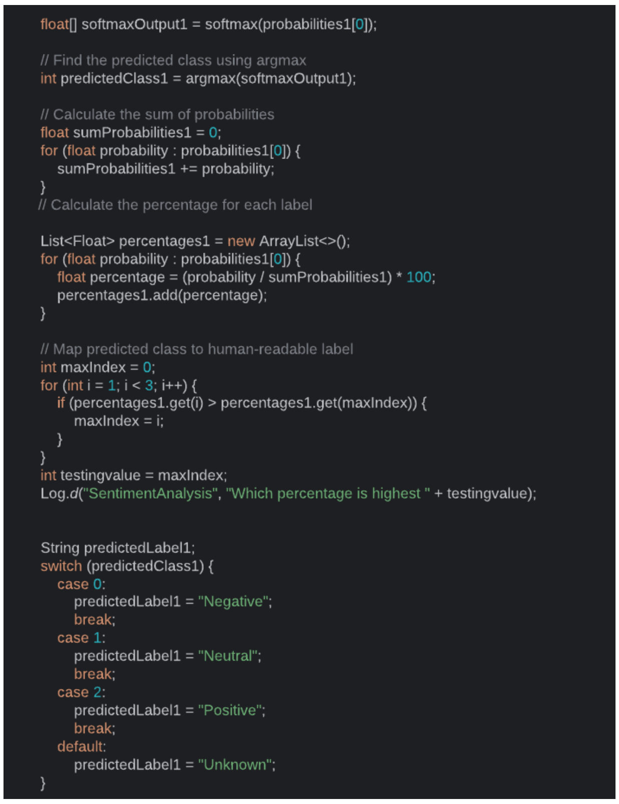
Figure 5-2-6-2-6-5 show the code of using a function named "runForMultipleInputsOutpus" which import from Interpreter Library to use the model integrated into Android Studio. By using try and catch method, the system will test run and determined whether the sentiment analysis is performing or not. If the sentiment analysis is not performing in correct way such as reason of exceed input size passed. It will return negative one as the sentiment analysis result.



```
try {
    interpreter.runForMultipleInputsOutputs(inputTensor1, outputTensor1);

} catch (Exception e) {
    Log.e("SentimentAnalysis", "Failed to run interpreter: " + e.getMessage());
    return -1; // or any other error code to indicate failure
}
```

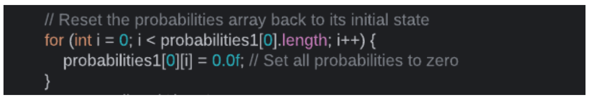
Figure 5-3-6-2-6-5 Code of Sentiment Analysis Class _ Analyze Sentiment Function Part 3

After the sentiment analysis performed, an function created that shown in Figure 5-2-6-2-6-14 and Figure 5-2-6-2-6-15 which named “softmax”, and argmax” will be performed to convert the output to human understandable value. Next, percentage calculation will be performed to determine the predicted class of the customer review and rating fall in either negative, neutral or positive. And based on the class predicted, it will return the value accordingly. Lastly, return the result and assign the result to the variable created in submit function in Shop Display Activity.



```
float[] softmaxOutput1 = softmax(probabilities1[0]);

// Find the predicted class using argmax
int predictedClass1 = argmax(softmaxOutput1);

// Calculate the sum of probabilities
float sumProbabilities1 = 0;
for (float probability : probabilities1[0]) {
    sumProbabilities1 += probability;
}
// Calculate the percentage for each label

List<Float> percentages1 = new ArrayList<>();
for (float probability : probabilities1[0]) {
    float percentage = (probability / sumProbabilities1) * 100;
    percentages1.add(percentage);
}

// Map predicted class to human-readable label
int maxIndex = 0;
for (int i = 1; i < 3; i++) {
    if (percentages1.get(i) > percentages1.get(maxIndex)) {
        maxIndex = i;
    }
}
int testingvalue = maxIndex;
Log.d("SentimentAnalysis", "Which percentage is highest " + testingvalue);


String predictedLabel1;
switch (predictedClass1) {
    case 0:
        predictedLabel1 = "Negative";
        break;
    case 1:
        predictedLabel1 = "Neutral";
        break;
    case 2:
        predictedLabel1 = "Positive";
        break;
    default:
        predictedLabel1 = "Unknown";
}
```

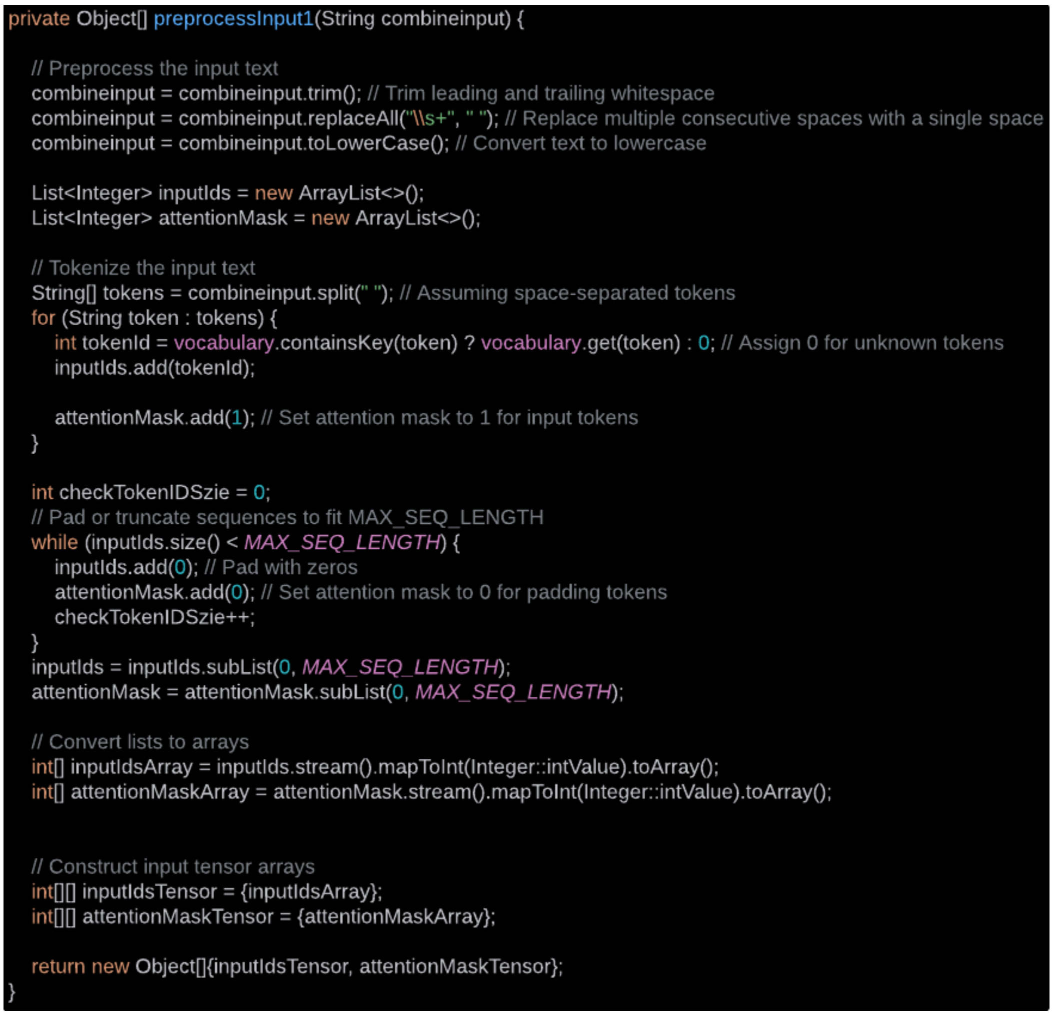
Figure 5-2-6-2-6-6 Code of Sentiment Analysis Class _ Analyze Sentiment Function Part 4



```
// Reset the probabilities array back to its initial state
for (int i = 0; i < probabilities1[0].length; i++) {
    probabilities1[0][i] = 0.0f; // Set all probabilities to zero
}
```

Figure 5-2-6-2-6-7 Code of Sentiment Analysis Class _ Analyze Sentiment Function Part 5

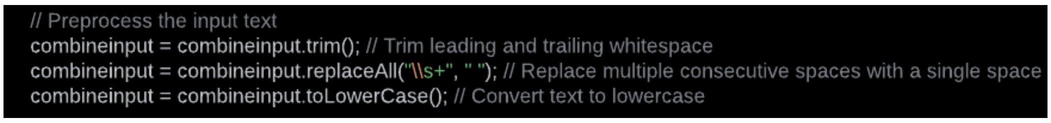
Due to the unexpected issues of unable to import BERT library and use its resources. Hence, to perform sentiment analysis in Android Studio, the pre-processing process of data input must have same logic with the model development. To ensure it is same logic, the vocabulary of the tokenizer is downloaded and saved in the resources file of android studio. Figure 5-2-6-2-6-8 show the complete code on pre-processing steps. The report will provide further information in Figure 5-2-6-2-6-9 to Figure 5-2-6-2-6-12.



```java
private Object[] preprocessInput1(String combineinput) {

    // Preprocess the input text
    combineinput = combineinput.trim(); // Trim leading and trailing whitespace
    combineinput = combineinput.replaceAll("\\s+", " "); // Replace multiple consecutive spaces with a single space
    combineinput = combineinput.toLowerCase(); // Convert text to lowercase

    List<Integer> inputIds = new ArrayList<>();
    List<Integer> attentionMask = new ArrayList<>();

    // Tokenize the input text
    String[] tokens = combineinput.split(" "); // Assuming space-separated tokens
    for (String token : tokens) {
        int tokenId = vocabulary.containsKey(token) ? vocabulary.get(token) : 0; // Assign 0 for unknown tokens
        inputIds.add(tokenId);

        attentionMask.add(1); // Set attention mask to 1 for input tokens
    }

    int checkTokenIDSzie = 0;
    // Pad or truncate sequences to fit MAX_SEQ_LENGTH
    while (inputIds.size() < MAX_SEQ_LENGTH) {
        inputIds.add(0); // Pad with zeros
        attentionMask.add(0); // Set attention mask to 0 for padding tokens
        checkTokenIDSzie++;
    }
    inputIds = inputIds.subList(0, MAX_SEQ_LENGTH);
    attentionMask = attentionMask.subList(0, MAX_SEQ_LENGTH);

    // Convert lists to arrays
    int[] inputIdsArray = inputIds.stream().mapToInt(Integer::intValue).toArray();
    int[] attentionMaskArray = attentionMask.stream().mapToInt(Integer::intValue).toArray();


    // Construct input tensor arrays
    int[][] inputIdsTensor = {inputIdsArray};
    int[][] attentionMaskTensor = {attentionMaskArray};

    return new Object[]{inputIdsTensor, attentionMaskTensor};
}
```

Figure 5-2-6-2-6-8 Code of Sentiment Analysis Class _ Preprocess Input

First steps in pre-processing phases it to removing the whitespace at beginning, middle and end of the sentences. Next, the word will be converted to lower case as it is performed in the model development phases. Figure 5-2-6-2-6-9 show how the text is performing first preprocessing steps.



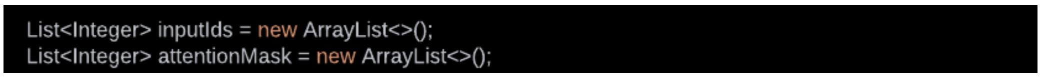
```
// Preprocess the input text
combineinput = combineinput.trim(); // Trim leading and trailing whitespace
combineinput = combineinput.replaceAll("\\s+", " "); // Replace multiple consecutive spaces with a single space
combineinput = combineinput.toLowerCase(); // Convert text to lowercase
```

Figure 5-2-6-2-6-9 Code of Sentiment Analysis Class _ Preprocess Input Part 1

Next, after the extra whitespace of the text is removed. An array list is created for preparing to store the tokens after the text is tokenized (Table 5-2-6-2-6-1). Figure 5-2-6-2-6-10 show the code of creating array for input ID and attention mask. Table 5-2-6-2-6-2 show the description of the variable created for pre-processing inputs.

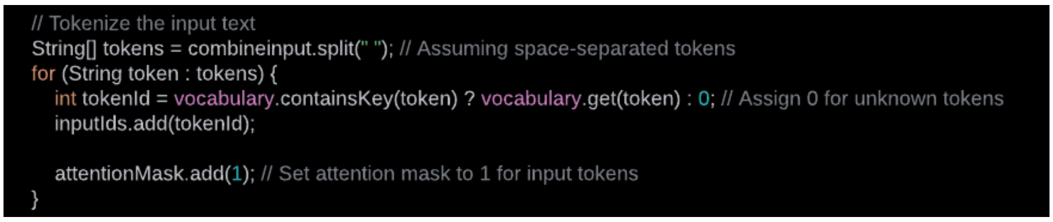
Variables	**Description**
inputIds	According to the vocabulary, tokenization is to find and match the vocabulary of the sentences of the review and retrieve its ID of each word from the BERT tokenization vocabulary. In this project, the input ID is set to have a maximum 128 ID.
attentionMask	Attention Mask is used as reference, with having maximum 128 ID. In the array list, if the value in array is filled with token ID in input ID, therefore in attention mask, it will store a value of 1, else value of 0 will store for empty space.

Table 5-2-6-2-6-2 Description of Input ID and Attention Mask



```
List<Integer> inputIds = new ArrayList<>();
List<Integer> attentionMask = new ArrayList<>();
```

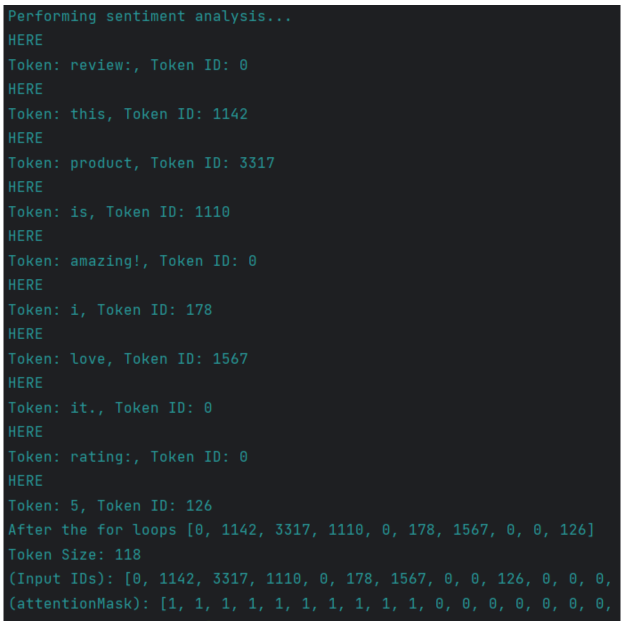
Figure 5-2-6-2-6-10 Code of Sentiment Analysis Class _ Preprocess Input Part 2



```
// Tokenize the input text
String[] tokens = combineinput.split(" "); // Assuming space-separated tokens
for (String token : tokens) {
    int tokenId = vocabulary.containsKey(token) ? vocabulary.get(token) : 0; // Assign 0 for unknown tokens
    inputIds.add(tokenId);

    attentionMask.add(1); // Set attention mask to 1 for input tokens
}
```

Figure 5-2-6-2-6-11 Code of Sentiment Analysis Class _ Preprocess Input Part 3

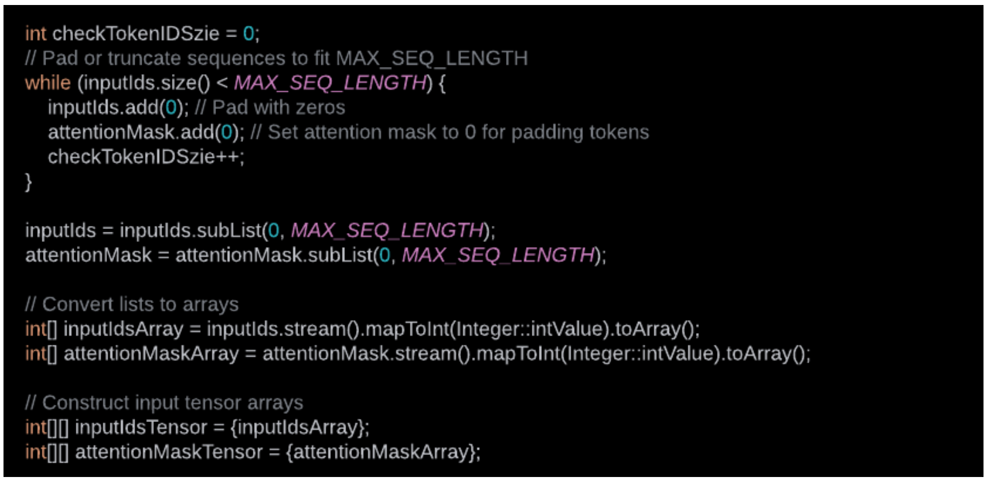
The text tokenization process will start by splitting the text based on space. And based on the vocabulary of BERT tokenizer to find and match the word ID. Figure 5-2-6-2-6-12 show the example of the text is tokenized. The example input is " Review: This product is amazing! I love it. Rating: 5".



```
Performing sentiment analysis...
HERE
Token: review:, Token ID: 0
HERE
Token: this, Token ID: 1142
HERE
Token: product, Token ID: 3317
HERE
Token: is, Token ID: 1110
HERE
Token: amazing!, Token ID: 0
HERE
Token: i, Token ID: 178
HERE
Token: love, Token ID: 1567
HERE
Token: it., Token ID: 0
HERE
Token: rating:, Token ID: 0
HERE
Token: 5, Token ID: 126
After the for loops [0, 1142, 3317, 1110, 0, 178, 1567, 0, 0, 126]
Token Size: 118
(Input IDs): [0, 1142, 3317, 1110, 0, 178, 1567, 0, 0, 126, 0, 0, 0,
(attentionMask): [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0,
```

Figure 5-2-6-2-6-12 Code of Sentiment Analysis Class _ Example Text Tokenization

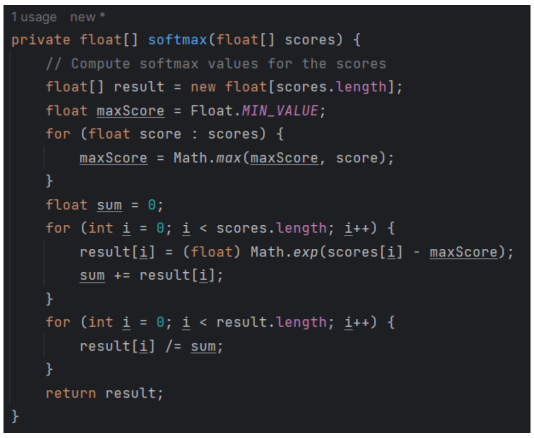
Last, according to the size of array. If there is empty value after text tokenization. This empty value will replace with a value of 0. After all steps is completed, convert the variable with array list data type into array data type. Lastly, creating an object with two data of input ID and attention mask for sentiment analysis.



```java
int checkTokenIDSzie = 0;
// Pad or truncate sequences to fit MAX_SEQ_LENGTH
while (inputIds.size() < MAX_SEQ_LENGTH) {
    inputIds.add(0); // Pad with zeros
    attentionMask.add(0); // Set attention mask to 0 for padding tokens
    checkTokenIDSzie++;
}

inputIds = inputIds.subList(0, MAX_SEQ_LENGTH);
attentionMask = attentionMask.subList(0, MAX_SEQ_LENGTH);

// Convert lists to arrays
int[] inputIdsArray = inputIds.stream().mapToInt(Integer::intValue).toArray();
int[] attentionMaskArray = attentionMask.stream().mapToInt(Integer::intValue).toArray();

// Construct input tensor arrays
int[][] inputIdsTensor = {inputIdsArray};
int[][] attentionMaskTensor = {attentionMaskArray};
```

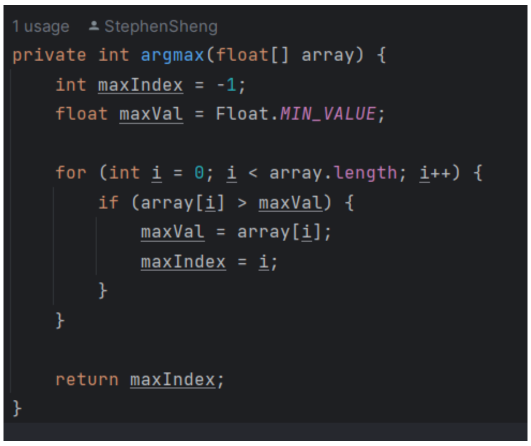
Figure 5-2-6-2-6-13 Code of Sentiment Analysis Class _ Preprocess Input Part 4

After the sentiment analysis is completed. To ensure the result is human understandable. The code show in Figure 5-2-6-2-6-14 and Figure 5-2-6-2-6-15 is used. As the result may have many decimal points, hence these two codes will help to convert it to readable format for human.



```
1 usage  new *
private float[] softmax(float[] scores) {
    // Compute softmax values for the scores
    float[] result = new float[scores.length];
    float maxScore = Float.MIN_VALUE;
    for (float score : scores) {
        maxScore = Math.max(maxScore, score);
    }
    float sum = 0;
    for (int i = 0; i < scores.length; i++) {
        result[i] = (float) Math.exp(scores[i] - maxScore);
        sum += result[i];
    }
    for (int i = 0; i < result.length; i++) {
        result[i] /= sum;
    }
    return result;
}
```

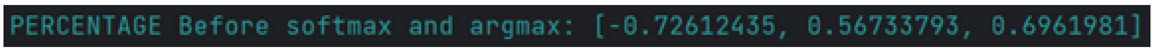
Figure 5-2-6-2-6-14 Code of Sentiment Analysis Class _ Softmax



```
1 usage  StephenSheng
private int argmax(float[] array) {
    int maxIndex = -1;
    float maxVal = Float.MIN_VALUE;

    for (int i = 0; i < array.length; i++) {
        if (array[i] > maxVal) {
            maxVal = array[i];
            maxIndex = i;
        }
    }

    return maxIndex;
}
```

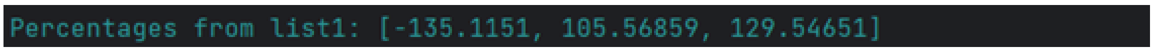
Figure 5-2-6-2-6-15 Code of Sentiment Analysis Class _ Argmax

Figure 5-2-6-2-6-16 show the result before performing softmax and avgmax function to convert the result into value that human understandable format. And Figure 5-2-6-2-6-17 show the result after applying the softmax and avgmax function.



```
PERCENTAGE Before softmax and argmax: [-0.72612435, 0.56733793, 0.6961981]
```

Figure 5-2-6-2-6-16 Result Before Applying Softmax and Avgmax Function



```
Percentages from list1: [-135.1151, 105.56859, 129.54651]
```

Figure 5-2-6-2-6-17 Result After Applying Softmax and Avgmax Function

5.3 Implementation Issues and Challenges

In this project, there are mainly three issues faced. Which are insufficient data used for modelling, model overfitting, performance of model in application. **Insufficient data** for modelling is a common issue faced in machine learning. It can lead to a result of low accuracy and affect significantly to the model performance. Moving on, it led to a failure when the mode dealing with the real customer's review and rating. It will lead to the model memorizing instead of learning. Eventually, when the model used in an application or software. It will fail to perform with complicated customer's review and rating. Hence, to ensure the model performance, batch data processing is implemented. According to Figure 5-3-3-9, the batch size is set to a value 8. Which allow the model to train by using the data batch by batch. Next, it helps to utilize the hardware memory and computational power more efficiently.

Model overfitting is another common issue happened in machine learning. According to Figure 5-2-4-4 to Figure 5-2-4-8, the data of training and validating loss show that the model is having model overfitting issues. As the training loss is decreasing and validating loss is increasing. This issue can cause by many factors. Including insufficient data, noisy data, in representative data, lack of regularization and so on. This issue happens when a model is performing unexpected outstanding on the training dataset, however having low ability and lead to low performance on classification on the testing dataset which in unseen data. This is because the model did not perform learning the pattern from the dataset provided, but only memorizing the training dataset. Hence, with the size of the dataset used increasing, the better the model performance with providing better accuracy.

Lastly, the **performance of model integration in application** is unexpectedly low accuracy which having mismatch with the performance of the model trained. During implementation phases, there are some issues and challenges faced and having suspicion to the issue may be the factor affect the performance of the model in application. Which is BERT library not able to import to Android Studio. Hence the application has to perform its own preprocessing in the system with same logic of the model developed. For that, the vocabulary resources file of BERT tokenizer is imported to perform the data preprocessing. Figure 5-2-6-2-3-10 and Figure 5-2-6-2-3-11 show the code of vocabulary importing and Figure 5-2-6-2-6-11 showing the code of data preprocessing. There are some hypotheses on the factors causing this issue which are the preprocessing logic mismatch with the modelling, output interpretation mismatch with modelling, and reliability and effectiveness of model developed affected by model overfitting. The main suspicion factors that affect the performance of model in the application is the problem of reliability and effectiveness of model developed affected by model overfitting and insufficient dataset. Table 5-3-1 show a list of potential solutions that could improve the performance. Including, increasing the size of dataset, hyperparameter tuning, could model development, and data augmentation. Increasing the dataset for model development will help to increase the performance of the model by avoiding model overfitting and other issues. It helps the model avoid memorizing the pattern, and start learning it better. Next, hyperparameter tuning help me select the better parameters for the model development including the batch size, token size and so on. Which help on the efficiency of the process. Moving on, developed a model in could such as using Google Cloud AI Platform. It provide more flexibility for the process such as using cloud infrastructure enable the process be more easy, efficient, flexible, and effective. Lastly, by performing data augmentation. It help to generate similar data by using original data to increase the diversity of the dataset for model training, validating, and testing. Which able to help on resolving issue of model overfitting.

No	Potential Solutions	Descriptions
1	Ideal Size of Dataset	By increasing the amount of data for model development will help to solve the model overfitting issues to avoid the model memorizing the patterns.
2	Hyperparameter Tuning	It is to select the best parameters for the model development. In terms of, batch size, token size, learning rate, and so on.
3	Cloud Model Development	Able to utilize the cloud infrastructure such as virtual environment. For example, Google Collaborative. With cloud model development, it can access different dataset to ensure the diversity and larger of dataset. It enables the integration process more easy, efficient, and effective. By using tools such as Google Cloud AI Platform.
4	Data Augmentation with Generative Models	It is to create a data sample that is similar with original data but having variations. It can help to increase the diversity of the dataset, improvement of robustness for the model, as well as resolve the model overfitting issue.

Table 5-3-1 Potential Solutions for Performance Improvement

5.4 Concluding Remark

The sentiment analysis model prioritizing on using review and rating as data input to perform classification to the three categories which are negative, neutral, and positive. This model is for data in language English. And further perform classification by using the sentiment analysis result and display in the application. This application provided a total four categories of shop for user to choose for recommendation. Which is healthcare, technology, mini market, and stationary. There are 10 shops for each category from the shop located in Kampar, Perak, Malaysia. The data of shops, and data collected from user, and sentiment analysis result is saved into Firebase Firestore.

For the features and functionalities of the application, it will recommend shops by displaying the shops of each category in main page that with fulfilling the requirement set which are having average rating of 3 to 5, and percentage from 86% to 100% of a shop getting positive review. Next, the user able to choose the category of shop pervaded for displaying the list of shop by a default displaying which is based on ordering of average rating. Moving on, a filter function is provided for user to perform filtering and ranking in two main criteria. Which are average rating and sentiment analysis result. According to the requirement and criteria that user can set, there are total 5 cases which elaborate in Table 5-2-6-2-2-4 and Figure 5-2-6-2-2-7 to Figure 5-2-6-2-2-7. The user able to click the shop for viewing more information such as location, contact method, and others. And user able to view the past review and provide review and rating.

Throughout the project development and implementation process, there are issues and challenges faced. Which including three main issues faced in the system implementation which are insufficient data, model overfitting, and performance of model integration in applications. There are come ineffective solution used, and inefficient solutions used. At last, to using the maximum data size of the hardware computational resources supported and perform batch processing for model training. Eventually, the accuracy of the model towards unseen data is 85.64%.

Chapter 6
System Evaluation and Discussion

This chapter will be discussing the system evaluation and discussion by system testing. Which according to the evaluation metric such as accuracy, f1-score, precision, recall, and confusion matrix. To further evaluate the model performance, there are 20 reviews and ratings are used for model testing by loading the model in Python IDLE, and applications. Hence, it enable to study the performance between then in terms of the final result and percentages of negative, neutral, and positive of review and rating.

6.1 System Performance Metrics

In this project, one of its objectives is to achieve at least an 80% average of accuracy of the sentiment analysis on training dataset, validation dataset, and testing dataset. However, in this project evaluation, it will include also other performance metrics that listed in Chapter 2.3 of the Table 2-3-1, such as recall, precision, and others. Which this evaluation will be performed using testing dataset. Moving on, the sentiment analysis model able to provide the percentages of either a review is negative, neutral, and positive. Table 6-1-1 show an example. System testing divided into three testing and evaluation direction. Firstly, testing and evaluating in model development process. Especially monitoring the accuracy calculated during training, validating, and testing. Second, after model is downloaded, by providing 20 reviews with rating to monitor the result given. Lastly, after model integrated with the application. Perform testing and monitoring the results. Further elaboration given if the performance is normal, outstanding, or bad. Giving hypothesis or estimation about the issues affecting the model performances.

Review / Comment	This is a good shop, it provides promotion and offer for student. Product price is low with good quality.		
Rating / Stars	5		
Percentages			**Result**
Negative (0)	**Neutral (1)**	**Positive (2)**	2
0.03%	0.51%	99.46%	

Table 6-1-1 Example of Sentiment Analysis Result

6.2 System Testing Setup and Result

6.2.1 Testing and Evaluation during Model Development

Figure 5-2-4-4 to Figure 5-2-4-8 show the result of the accuracy and loss of training and validating. And summarized them into Table 6-2-1-1 and Table 6-2-1-2. According to the first table, the average accuracy of the four iterations of training and validating is 90.27% and 85.70%. Based on the Table 6-2-1-2, it shows the data of training losses and validation losses. This result of losses show that the model is having issues of overfitting. This is because, a model that perform well is expecting to have training loss decreasing and validation loss decreasing. However, in this model developed, it is having decreasing of training loss and increasing in validation loss. The main reason of model overfitting happening is due to the size of the dataset for training is insufficient.

Phases	**Accuracy**				**Average Accuracy**
	1st Iteration	**2nd Iteration**	**3rd Iteration**	**4th Iteration**	
Training	84.54%	89.07%	92.62%	94.83%	90.27%
Validating	86.37%	86.06%	85.43%	84.92%	85.70%
Testing	85.64%				85.64%

Table 6-2-1-1 Evaluation of Model Development (Accuracy)

Phases	**Losses**				**Average Losses**
	1st Iteration	**2nd Iteration**	**3rd Iteration**	**4th Iteration**	
Training	-1951.5916	-1380.5557	-960.3949	-685.2775	-1224.4549
Validating	-221.1854	-249.1560	-265.9890	-309.9155	-261.5615

Table 6-2-1-2 Evaluation of Model Development (Losses)

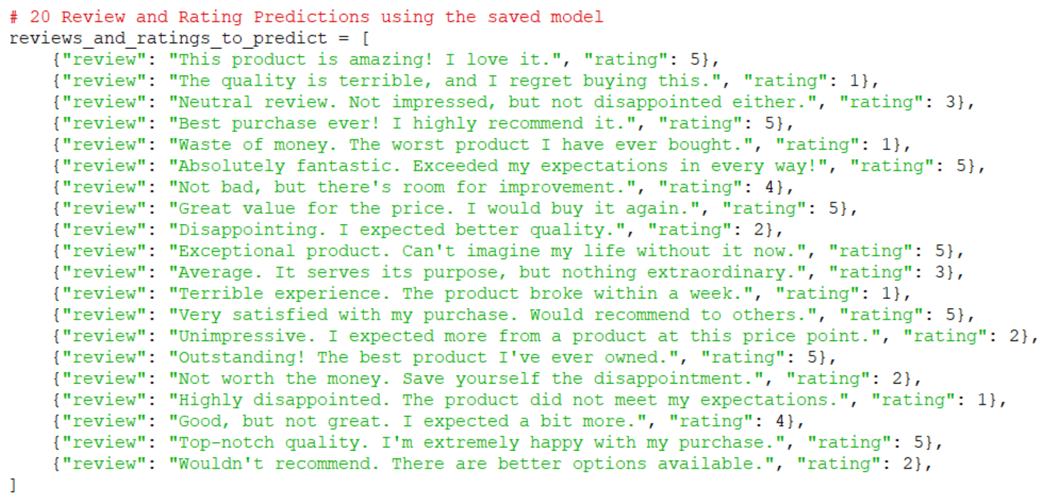
Figure 5-2-5-3 show the result of evaluation of the model towards unseen data and summarize into Table 6-2-1-3. The performance of the model towards the unseen data, which is testing data, the accuracy shows 85.64%. Next, precision indicate the proportion of true positive prediction among all positive prediction. The result show that the model has 86.05% can avoid false positive prediction. 85.64% of recall show that the model's ability on performing true positive prediction among all actual positive cases in the unseen data. Moving on, F1 score provide a value of mean among precision and recall which is 85.83%. Lastly, confusion matrix shows the detail data on actual positive and negative cases and predicted positive and negative cases. There are 1838 cases predicted to be positive and the actual is positive, 200 cases predicted to be positive however the actual is negative. Next, there are 249 cases predicted negative but actual is positive, and 638 cases predicted as negative and actual is negative too. In conclusion, the model performance is having 85.64% accuracy towards the unseen data.

Accuracy	0.8564	
Precision	0.8605	
Recall	0.8564	
F1 Score	0.8583	
Confusion Matrix		
	Predicted Positive	**Predicted Negative**
Actual Positive	1838 (TP)	249 (FN)
Actual Negative	200 (FP)	638 (TN)

Table 6-2-1-2 Losses of Modelling

6.2.2 Testing and Evaluation with Python IDLE

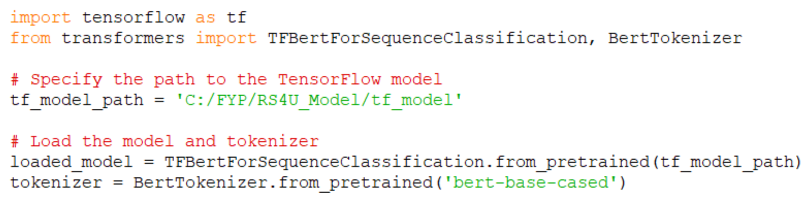
In this sub chapter, there will be a 20 reviews and ratings to test and evaluate the performance of the model saved shown in Figure 6-2-2-1.



```
# 20 Review and Rating Predictions using the saved model
reviews_and_ratings_to_predict = [
    {"review": "This product is amazing! I love it.", "rating": 5},
    {"review": "The quality is terrible, and I regret buying this.", "rating": 1},
    {"review": "Neutral review. Not impressed, but not disappointed either.", "rating": 3},
    {"review": "Best purchase ever! I highly recommend it.", "rating": 5},
    {"review": "Waste of money. The worst product I have ever bought.", "rating": 1},
    {"review": "Absolutely fantastic. Exceeded my expectations in every way!", "rating": 5},
    {"review": "Not bad, but there's room for improvement.", "rating": 4},
    {"review": "Great value for the price. I would buy it again.", "rating": 5},
    {"review": "Disappointing. I expected better quality.", "rating": 2},
    {"review": "Exceptional product. Can't imagine my life without it now.", "rating": 5},
    {"review": "Average. It serves its purpose, but nothing extraordinary.", "rating": 3},
    {"review": "Terrible experience. The product broke within a week.", "rating": 1},
    {"review": "Very satisfied with my purchase. Would recommend to others.", "rating": 5},
    {"review": "Unimpressive. I expected more from a product at this price point.", "rating": 2},
    {"review": "Outstanding! The best product I've ever owned.", "rating": 5},
    {"review": "Not worth the money. Save yourself the disappointment.", "rating": 2},
    {"review": "Highly disappointed. The product did not meet my expectations.", "rating": 1},
    {"review": "Good, but not great. I expected a bit more.", "rating": 4},
    {"review": "Top-notch quality. I'm extremely happy with my purchase.", "rating": 5},
    {"review": "Wouldn't recommend. There are better options available.", "rating": 2},
]
```

Figure 6-2-2-1 20 Reviews and Ratings Prepared for Testing and Evaluation

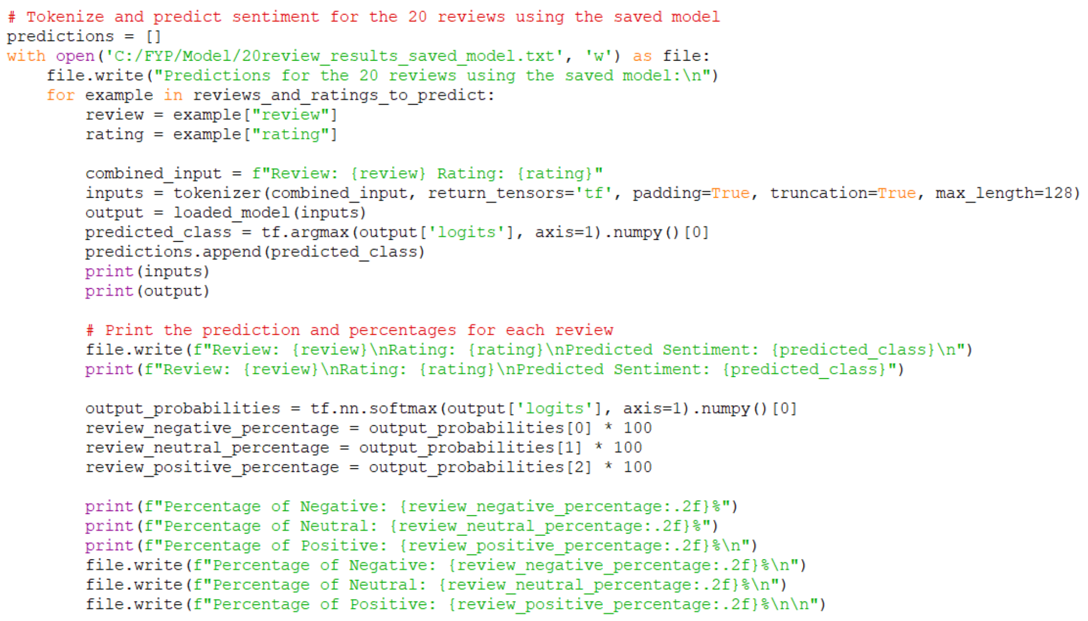
Before the model start to perform, the code will load the model saved and the BERT tokenizer.



```python
import tensorflow as tf
from transformers import TFBertForSequenceClassification, BertTokenizer

# Specify the path to the TensorFlow model
tf_model_path = 'C:/FYP/RS4U_Model/tf_model'

# Load the model and tokenizer
loaded_model = TFBertForSequenceClassification.from_pretrained(tf_model_path)
tokenizer = BertTokenizer.from_pretrained('bert-base-cased')
```

Figure 6-2-2-2 20 Model and Resources Loading

Figure 6-2-2-3 show the code of model performing. Firstly, creating an array for storing the prediction for each review and ratings. Next, extract the review and rating from the list, and perform preprocessing lastly used the model saved to perform sentiment analysis. After output is retrieve, it will be calculated by using arguments of the maxima function and SoftMax function to get the probabilities for percentage calculation. Lastly, saving the result into a text file. Full result presented in appendix A1 from page A-1 and restructure into Table 6-2-2-1. In this table, it provided the review and rating with its percentage of each label which are negative, neutral, and positive. According to the result, it can be determined that the model trained is having well performance.



```python
# Tokenize and predict sentiment for the 20 reviews using the saved model
predictions = []
with open('C:/FYP/Model/20review_results_saved_model.txt', 'w') as file:
    file.write("Predictions for the 20 reviews using the saved model:\n")
    for example in reviews_and_ratings_to_predict:
        review = example["review"]
        rating = example["rating"]

        combined_input = f"Review: {review} Rating: {rating}"
        inputs = tokenizer(combined_input, return_tensors='tf', padding=True, truncation=True, max_length=128)
        output = loaded_model(inputs)
        predicted_class = tf.argmax(output['logits'], axis=1).numpy()[0]
        predictions.append(predicted_class)
        print(inputs)
        print(output)

        # Print the prediction and percentages for each review
        file.write(f"Review: {review}\nRating: {rating}\nPredicted Sentiment: {predicted_class}\n")
        print(f"Review: {review}\nRating: {rating}\nPredicted Sentiment: {predicted_class}")

        output_probabilities = tf.nn.softmax(output['logits'], axis=1).numpy()[0]
        review_negative_percentage = output_probabilities[0] * 100
        review_neutral_percentage = output_probabilities[1] * 100
        review_positive_percentage = output_probabilities[2] * 100

        print(f"Percentage of Negative: {review_negative_percentage:.2f}%")
        print(f"Percentage of Neutral: {review_neutral_percentage:.2f}%")
        print(f"Percentage of Positive: {review_positive_percentage:.2f}%\n")
        file.write(f"Percentage of Negative: {review_negative_percentage:.2f}%\n")
        file.write(f"Percentage of Neutral: {review_neutral_percentage:.2f}%\n")
        file.write(f"Percentage of Positive: {review_positive_percentage:.2f}%\n\n")
```

Figure 6-2-2-3 20 Model Performing towards 20 Reviews and Ratings

No	**Review**	**Rating**	**Percentages**		
			Negative (0)	**Neutral (1)**	**Positive (2)**
1	This product is amazing! I love it.	5	0.13%	1.64%	**98.24%**
2	The quality is terrible, and I regret buying this.	1	**98.85%**	0.61%	0.53%

3	Neutral review. Not impressed, but not disappointed either.	3	6.18%	**93.40%**	0.42%
4	Best purchase ever! I highly recommend it.	5	0.04%	0.02%	**99.94%**
5	Waste of money. The worst product I have ever bought.	1	**99.31%**	0.46%	0.23%
6	Absolutely fantastic. Exceeded my expectations in every way!	5	0.03%	0.02%	**99.95%**
7	Not bad, but there's room for improvement.	4	5.58%	**93.41%**	1.01%
8	Great value for the price. I would buy it again.	5	0.36%	30.49%	**69.15%**
9	Disappointing. I expected better quality.	2	**97.51%**	2.28%	0.20%
10	Exceptional product. Can't imagine my life without it now.	5	0.65%	35.61%	**63.73%**
11	Average. It serves its purpose, but nothing extraordinary.	3	0.34%	**99.43%**	0.23%
12	Terrible experience. The product broke within a week.	1	**99.47%**	0.34%	0.19%
13	Very satisfied with my purchase. Would recommend to others.	5	0.08%	0.43%	**99.49%**
14	Unimpressive. I expected more from a product at this price point.	2	**98.66%**	1.16%	0.18%
15	Outstanding! The best product I've ever owned.	5	0.05%	0.02%	**99.93%**
16	Not worth the money. Save yourself the disappointment.	2	**99.09%**	0.72%	0.19%
17	Highly disappointed. The product did not meet my expectations.	1	**99.18%**	0.60%	0.22%
18	Good, but not great. I expected a bit more.	4	0.38%	**97.24%**	2.38%
19	Top-notch quality. I'm extremely happy with my purchase.	5	0.03%	0.04%	**99.93%**

20	Wouldn't recommend. There are better options available.	2	**97.87%**	1.92%	0.21%

Table 6-2-2-1 Result of Model Testing on 20 Reviews and Ratings in Python IDLE

6.2.3 Testing and Evaluation after Integration with Application

From Figure 6-2-3-1 to Figure 6-2-3-20, these figures represent the results of the sentiment analysis for the reviews and ratings stated in Table 6-2-3-1. The result shows the model integrated in application is having issues on memorizing the pattern instead of learning. There are two specials cases that the percentage calculated, and the result given is different from each other. Which are review and rating number 15 and 17.

No	**Review**	**Rating**	**Percentages**			**Result**
			Negative (0); Neutral (1); Positive (2)			
1	This product is amazing! I love it.	5	-582.12665%	-208.61723,	**890.7439%**	2
2	The quality is terrible, and I regret buying this.	1	-408.9186%	**1232.3624%**	-723.44385%	1
3	Neutral review. Not impressed, but not disappointed either.	3	-759.662%	352.31033%	**507.35168%**	2
4	Best purchase ever! I highly recommend it.	5	-135.1151%	105.56859%	**129.54651%**	2
5	Waste of money. The worst product I have ever bought.	1	**720.55457%**	-510.4788%	-110.07576%	0
6	Absolutely fantastic. Exceeded my expectations in every way!	5	-4547.8174%	**2374.632%**	2273.185%	1
7	Not bad, but there's room for improvement.	4	-1037.7306%	**920.74414%**	216.98647%	1
8	Great value for the price. I would buy it again.	5	-408.9186%	**1232.3624%**	-723.44385%	1

9	Disappointing. I expected better quality.	2	-4547.8174%	**2374.632%**	2273.185**%**	1
10	Exceptional product. Can't imagine my life without it now.	5	**720.55457%**	-510.4788%	-110.07576%	0
11	Average. It serves its purpose, but nothing extraordinary.	3	-4547.8174%	**2374.632%**	2273.185**%**	1
12	Terrible experience. The product broke within a week.	1	**720.55457%**	-510.4788%	-110.07576%	0
13	Very satisfied with my purchase. Would recommend to others.	5	-759.662%	352.31033%	**507.35168%**	2
14	Unimpressive. I expected more from a product at this price point.	2	-453.23633%	-84.66494%	**637.90125%**	2
15	Outstanding! The best product I've ever owned.	5	**23365.885%**	-4731.924%	-18533.96**%**	2
16	Not worth the money. Save yourself the disappointment.	2	-582.12665%	-208.61723,	**890.7439%**	2
17	Highly disappointed. The product did not meet my expectations.	1	**23365.885%**	-4731.924%	-18533.96**%**	2
18	Good, but not great. I expected a bit more.	4	-135.1151%	105.56859%	**129.54651%**	2
19	Top-notch quality. I'm extremely happy with my purchase.	5	-837.718%	364.73328%	**572.98474%**	2
20	Wouldn't recommend. There are better options available.	2	-135.1151%	105.56859%	**129.54651%**	2

Table 6-2-3-1 Result of Model Testing on 20 Reviews and Ratings in Android Studio

6.2.4 Comparison of Testing and Evaluation in Python IDLE and Application

According to Table 6-2-4-1, it shows the comparison of the result between the testing by using Python IDLE and application in Android Studio from Table 6-2-2-1 and Table 6-2-3-1. There is total 20 results, 8 result is same, and 12 result is different from each other's. Total of 7 results are having totally different result in terms of negative to positive and positive to negative considering huge difference. The factors cause these situations happening could be the insufficient data, model overfitting, library usage. For library usage, due to the Android Studio version restriction, it is not able to directly import BERT Tokenizer as the code in Figure 6-2-2-2. Hence, the preprocessing of the data input is having slightly different, but the logic apply is same. The method of tokenization's. The other factor such as the model file type using. In Chapter 6.2.2, the model used is in file type of H5, and the model format using in application is file type of TFLITE. This is due to the model file size is huge and to ensure the model can perform more efficiently in Android Studio, it is suggested that file type of TFLITE is better than H5. When converting the model from H5 file type to TFLITE file type, it is expected that there will be slightly affecting the accuracy of the model performance.

No	**Review**	**Rating**	**Percentages**		**Result Matching** (Match) (Not Match)
			Negative (0); Neutral (1); Positive (2)		
			Python	**Android Studio**	
1	This product is amazing! I love it.	5	2	2	Match
2	The quality is terrible, and I regret buying this.	1	0	1	Not Match
3	Neutral review. Not impressed, but not disappointed either.	3	1	2	Not Match
4	Best purchase ever! I highly recommend it.	5	2	2	Match
5	Waste of money. The worst product I have ever bought.	1	0	0	Match
6	Absolutely fantastic. Exceeded my expectations in every way!	5	2	1	Not Match

7	Not bad, but there's room for improvement.	4	1	1	Match
8	Great value for the price. I would buy it again.	5	2	1	Not Match
9	Disappointing. I expected better quality.	2	0	1	Not Match
10	Exceptional product. Can't imagine my life without it now.	5	2	0	Not Match
11	Average. It serves its purpose, but nothing extraordinary.	3	1	1	Match
12	Terrible experience. The product broke within a week.	1	0	0	Match
13	Very satisfied with my purchase. Would recommend to others.	5	2	2	Match
14	Unimpressive. I expected more from a product at this price point.	2	0	2	Not Match
15	Outstanding! The best product I've ever owned.	5	2	0	Not Match
16	Not worth the money. Save yourself the disappointment.	2	0	2	Not Match
17	Highly disappointed. The product did not meet my expectations.	1	0	0	Match
18	Good, but not great. I expected a bit more.	4	1	2	Not Match
19	Top-notch quality. I'm extremely happy with my purchase.	5	2	0	Not Match
20	Wouldn't recommend. There are better options available.	2	0	2	Not Match

Table 6-2-4-1 Result of Model Testing Comparison Between Table 6-2-2-1 and Table 6-2-3-1

6.3 Project Challenges

In this project, there are few challenges faced including hardware capabilities, and software effectiveness. The hardware capabilities play a vital role in the project especially for model development. During system implementation, the abilities of Central Processing Unit (CPU) and the memory of the hardware will affect the progress and result of the system implementation. The modelling phases require a huge computational resource such as memory allocation and time consuming. Which causing the model development is not able to perform with the ideal dataset size such as in millions. Due to the limited capability of the hardware, the model development must reduce the time consume and avoid unexpected problem occur to the hardware. The system must reduce the dataset size and setting the dataset into batch to ensure the model can be developed within short period of time.

Second challenges faced was the software selection and usage. Android Studio is chosen for application development. The latest version is not supporting every library from third party. The model trained using BERT with its resources such as tokenizer. However, in Android Studio, BERT library is not allowed to import to the dependencies of the application. Hence, some process of the application system has to perform itself. For example, tokenization process. It is expected to use the pre-created function from BERT library in Android Studio. Unfortunately, part of the process of sentiment analysis have to perform manually such as text tokenization by downloading the vocabulary resources of the BERT. The second software is the Python IDLE. Due to the hardware capabilities restriction, during the modelling phase, the model was decided to change to another platform to develop. Hence, Google Collaborative was selected. However, due to the free version, the capabilities are also not able to support the model development with ideal dataset size. Eventually, the model is developed by continue using Python IDLE. To perform improvement, there are suggestions provided in Chapter 7 recommendations.

6.4 Objectives Evaluation

Objectives 1: To develop a system that can use the result of sentiment analysis of customers' reviews and comments to classify the shops into different groups.

The primary goal of this project is to build a sentiment analysis system that able to classify the customer's review and rating into different groups which are negative, neutral, and positive. In this project, the sentiment analysis model successfully developed with an accuracy of 85.64% to the unseen data.

Objectives 2: To develop a system that utilize the result of sentiment analysis to improve customer purchasing experience and satisfaction.

This objective is achieved through a further classification on the filtering function for user to retrieve a better recommendation result from the application. Which the sentiment analysis model developed provide a result of percentage for negative, neutral, and positive of a customer's review and rating. In this project, the result is utilized by providing 5 different level of choice according to the range percentage of a shop getting positive review. Hence, the application user can choose the specific range of percentage to get recommendation.

Objectives 3: To develop a system that achieve at least a minimum 80% average of accuracy to the unseen data which is testing dataset.

Third goal it to access the efficiency and reliability of the model developed on performing sentiment analysis on the customer's review and rating. In addition, the average accuracy for the model to the training dataset is 90.27%, and to the validating dataset is 85.70%. And 85.64% accuracy of the model performance on unseen data known as testing data. Table 6-2-1-1 show the accuracy of each iteration of the training and validating phases, and all the accuracy of them is having more than 80%. Hence, it can be determined that, the model is performing well based on the goal set.

6.5 Concluding Remark

In conclusion, system evaluation used accuracy metrics as main evaluation method. The accuracy of the model perform in unseen data is having more than 80% accuracy. And the accuracy of each iteration of training and validating is having more than 80% accuracy as well. Other than that, other evaluation methods are included as well. Including, recall, precision, F1 score, and confusion metrics in Table 6-2-1-2 with explanation in Chapter 6.2.1.

The project facing two main challenges which is the hardware capabilities, and software usage and selection. Where hardware capabilities play a vital role on model development. Which affect the efficiency and effectiveness of model development process. Next, software can affect the process of integration of the model trained with the application. The project faced the library that used in model development is not able to import to Android Studio. Which affect the performance of the model in the application. As the preprocessing process has to perform manually by using same resources file of the model development used. With facing insufficient dataset for model development, it affects the performance of the model in application significantly. Causing the performance of the model is having mismatch accuracy and expectation.

The fulfilment of the objective shows the success of a sentiment analysis model being developed. Objective one aim to develop a system that can use the result of sentiment analysis of customers' reviews and comments to classify the shops into different groups. The groups such as negative, neutral, and positive. By providing a review and rating to the model. It can analyse and provide the percentage of particular review and rating fall into which groups. As the example stated in Table 6-1-1. Second objective aim to utilize the sentiment analysis result for further classification to perform recommendation. In this project, there is 5 level of range for the percentage of a shop getting positive review. These five levels represent by an adjective in word or phrase (Table 5-3-6-2-5-2). Lastly, the third objective is to aim to have an accuracy of more than 80% on the unseen data. Table 6-2-1-1 show the average of accuracy to the model performing on training and validating dataset, and testing dataset with more than an average accuracy of 80%.

Chapter 7
Conclusion and Recommendations

This chapter will divide into two parts: conclusion and recommendation. Which conclude the entire project process, progress, and performance. This chapter provided suggestions and recommendations for future improvement and enhancement as well.

7.1 Conclusion

In conclusion, the development of the recommender system by using user's review and rating shows an importantcy and improvement in the recommendation area. The project overall achieve its main and primary objective including develop a system that can use the result of sentiment analysis of customers' reviews and comments to classify the shops into different groups, to develop a system that utilize the result of sentiment analysis to improve customer purchasing experience and satisfaction, and to develop a system that achieve an acuuracy of 85.64% where the goal is to get at least a minimum 80% average of accuracy to the unseen data which is testing dataset to ensure its reliability and effectiveness.

This project development process faced lots of challenges, which included data insufficient, model overfitting, unmatch in performance or the model, hardware restrictions and limitation, and software version restrictions. However, the issues are solved by changing the parameters of the modelling to maximize the dataset size. The setting of parameters changed including the batch size, token size, and number of iterations. Next, due to retriction of the software of application development, the preprocessing steps is conducted manually in the activity code by downloading the same resources file form model development. To improve the performance in the application, it is related to the model development process. Hence, in the model development process, it maximized the dataset size for trying to reduce the model memorizing and ensuring the model learning is effective.

The project covers the basic functions that planned and aimed to developed. However, there are more improvement and suggestions for the model and application development no matter to resolve current issues or improve and enhance current features. Which will further discussed in next sub chapter.

7.2 Recommendation

This project proposes a recommender system and its implementation are not cover and include many features and performing outstanding compare with the existing applications that is mature and well developed. Hence there are some recommendations for this project for further development and perfection. Firstly, the **model development** in this project mainly used BERT techniques. Hence, to achieve further, it is suggested to have different techniques used together for comparison and usage in the application. Next, the sentiment analysis model is suggested to support for multi-language. The main language of sentiment analysis is English in the project. To develop a model that is outstanding, the data sources is suggested to be more diverse. As there are difference even same language is used by user. For example, Malaysia, Singapore, and other country. The pattern on usage of the English is different with United Kingdom, United State, and others. Second example, Mandarin in China, Malaysia, Singapore, and other country are having a certain difference. Hence there is a need to have domain specific preprocessing to handle the special vocabulary such as slang or jargon. Moving on, the model is suggested build in cloud to ensure the model can be developed in real time by using Google Could and AI, Azure Platform, and others. Lastly, this project focuses on recommending store and shop. Hence, it is suggested that there can be sentiment analysis model based on item and product.

There are not many **features and functionalities** being developed for the application. Hence it is suggested that the application shall have features and functionalities such as user account creation, shop owner account creation, shopping system, payment system, search function, location service such as integrate with Google Map or Waze, and others features and functionalities. Instead of application development, it is suggested to change to website application developments which the platform can be access in both ways. To development an outstanding recommendation system application, the application can support recommendation based on categories of item. Next, the user interface can be more attractive and interactive. For example, a small game provided to user to earn coin for promotion or offer.

Lastly, the **area of recommendation** is suggested to be more diverse. For example, perform recommendations based on locations, user profile and background, and other areas. The recommendation system that based on keywords, location, and so on, it is suggested to develop in real time basis, hence the model can update itself in real time to perform better.

REFERENCES

[1] S. Dan, R. Jean-David and S. Badrul, "Large-scale Item Categorization for e-Commerce," in *Proceedings of the 21st ACM international conference on information and knowledge management*, 2012.

[2] J. Z. Zoo, "Analysis of e-Commerce Characteristics Based on Edge Algorithm," Wireless Communications and Mobile Computing, 2021.

[3] G. Shahriari-Mehr, D. R. Mahmoud, C. Christophe, A. N. Babak and D. Mohammad-Reza, "A store location-based recommender system using user's position and web searches," Journal of Location Based Services, 2021.

[4] S. Rithika, R. R. Gundla, R. Varun, K. Gv Santosh and P. Chandra Sekhar, "The Role of E-commerce on Customer Engagement in 2021," International Journal of Scientific Development and Research, 2021.

[5] X. Wang and S. Dong, "Users' Sentiment Analysis of Shopping Websites Based on Online Reviews," Applied Mathematics and Nonlinear Sciences, 2020.

[6] B. J. Jiang and T. X. Zou, "Consumer Search and Filtering on Online Retail Platforms," Journal of Marketing Research, 2020.

[7] R. Baptiste, "Towards Data Science," Towards Data Science, 3 Jun 2019. [Online]. Available: https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada. [Accessed 20 November 2022].

[8] H. S. Chen, J. S. Zhao and D. W. Yin, "Fine-Grained Product Categorization in E-commerce," E-commerce & Production, 2019.

[9] M. Nitin, S. Chaturvedi, K. Prashanthi and N. K. Chandrakar, "Review of Various Recommendation Techniques," in International Journal of Engineering Research & Technology, 2015.

[10] N. P. Akash, H. Manoj, A. S. Suhas and L. Abhay, "Online Product Reviews and Their Impact on Third Party Sellers Using Natural Language Processing," in International Journal of Business Intelligence Research, 2021.

[11] S. K. Pradeep, P. K. D. Pijush, D. K. Avick and C. Pramanik, "Recommender systems: an overview, research trends, and future directions," in National Institute of Technology, 2021.

[12] Y. Li, Y. Li, J. Wang and S. Sherratt, "Sentiment Analysis for E-Commerce Product Reviews in Chinese Based on Sentiment Lexicon and Deep Learning," in Digital Object Identifier, 2019.

[13] S. A. Dhruv, S. and A. , "Feature based Sentiment Analysis for Product," in International Journal of Engineering Research & Technology, 2022.

[14] Z. Drus and H. Khalid, "Sentiment Analysis in Social Media and Its Application: Systematic Literature Review," *Procedia Computer Science*, 2019.

[15] C. JingFeng, W. ZhaoXia, H. SendBeng and C. Erik , "Survey on sentiment analysis: evolution of research methods," *Artifcial Intelligence Review,* 2023.

[16] T. Hamed and M. Mitra, "Artificial Intelligence and Sentiment Analysis: A Review in Competitive Research," *MDPI Computer,* 2023.

[17] F. Xing and Z. Justin, "Sentiment analysis using product review data," *Journal of Big Data,* 2015.

[18] M. H. Ataie, "Basic Implementation of sentiment analysis using BERT," Department of Computer Engineering University of Tehran, 2022.

[19] B. Himanshu, P. Narinder Sigh, S. Snajay Kumar and A. Sonali, "BERT-Based Sentiment Analysis: A Software," *Computer Vision and Pattern Recognition*, 2021.

[20] Z. Zhou Gui, "Research on Sentiment Analysis Model of Short Text Based on Deep Learning," *Artificial Intelligence for Evaluation Decision-making in Modern Product Design,* 2022.

[21] W. YiLi, G. JiaXuan, Y. ChengSheng and L. BaoZhu, "Sentiment Analysis of Twitter Data," *Applied Sciences,* 2022.

[22] S. Dorababu, K. Siva and R. Ljagajeevan, "Sentiment Analysis for Social Networks Using MachineLearning Techniques," *International Journal of Engineering and Technology,* 2018.

[23] M. Samah, "Social Media Analysis of User's Responses to Terrorism Using Sentiment Analysis and Text Mining," *Procedia Computer Science*, 2019.

[24] A. A. Qaid Aqlan, M. Bairam and L. Naik, "A Study of Sentiment Analysis: Concepts, Techniques, and Challenges," 2019.

[25] S. Shayaa, P. Seuk Wai, Y. Wai Chung, S. Ainin, N. Ismawati Jaafar and S. Bahri Zakaria, "Social Media Sentiment Analysis on Employment in Malaysia," in *8th Global Business and Finance Research Conference*, Taipei, 2017.

[26] S. Schröer, F. Kruse and G. C. M. Jorge, "A Systematic Literature Review on Applying CRISP-DM Process Model," Procedia Computer Science, 2021.

[27] V. Singh, "https://medium.com/codingurukul/introduction-to-firebase-f9f6ccc8a785," Medium, 18 December 2018. [Online]. Available: https://medium.com/codingurukul/introduction-to-firebase-f9f6ccc8a785. [Accessed 15 January 2024].

[28] S. Malgaonkar, S. Sumeet and Y. Radia, "A Review and Basic Guidelines on Developing Android Applications," International Journal of Computer Applications, 2015.

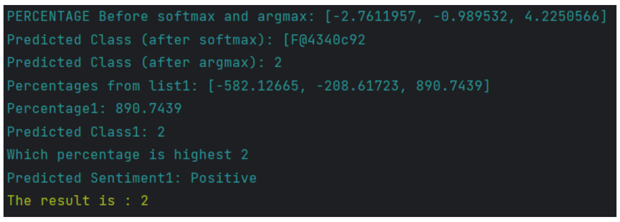
[29] M. Neary , "Getting Started With Python IDLE," Real Python, [Online]. Available: https://realpython.com/python-idle/. [Accessed 14 January 2024].

APPENDIX

A.1 Result of Testing and Evaluation with Python IDLE

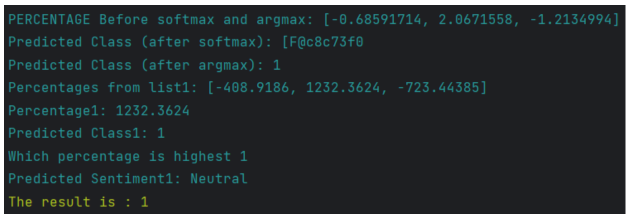
```
Predictions for the 20 reviews using the saved model:
Review: This product is amazing! I love it.
Rating: 5
Predicted Sentiment: 2
Percentage of Negative: 0.13%
Percentage of Neutral: 1.64%
Percentage of Positive: 98.24%

Review: The quality is terrible, and I regret buying this.
Rating: 1
Predicted Sentiment: 0
Percentage of Negative: 98.85%
Percentage of Neutral: 0.61%
Percentage of Positive: 0.53%

Review: Neutral review. Not impressed, but not disappointed either.
Rating: 3
Predicted Sentiment: 1
Percentage of Negative: 6.18%
Percentage of Neutral: 93.40%
Percentage of Positive: 0.42%

Review: Best purchase ever! I highly recommend it.
Rating: 5
Predicted Sentiment: 2
Percentage of Negative: 0.04%
Percentage of Neutral: 0.02%
Percentage of Positive: 99.94%

Review: Waste of money. The worst product I have ever bought.
Rating: 1
Predicted Sentiment: 0
Percentage of Negative: 99.31%
Percentage of Neutral: 0.46%
Percentage of Positive: 0.23%

Review: Absolutely fantastic. Exceeded my expectations in every way!
Rating: 5
Predicted Sentiment: 2
Percentage of Negative: 0.03%
Percentage of Neutral: 0.02%
Percentage of Positive: 99.95%

Review: Not bad, but there's room for improvement.
Rating: 4
Predicted Sentiment: 1
Percentage of Negative: 5.58%
Percentage of Neutral: 93.41%
Percentage of Positive: 1.01%

Review: Great value for the price. I would buy it again.
Rating: 5
Predicted Sentiment: 2
Percentage of Negative: 0.36%
Percentage of Neutral: 30.49%
Percentage of Positive: 69.15%

Review: Disappointing. I expected better quality.
Rating: 2
Predicted Sentiment: 0
Percentage of Negative: 97.51%
Percentage of Neutral: 2.28%
Percentage of Positive: 0.20%

Review: Exceptional product. Can't imagine my life without it now.
Rating: 5
Predicted Sentiment: 2
Percentage of Negative: 0.65%
Percentage of Neutral: 35.61%
Percentage of Positive: 63.73%

Review: Average. It serves its purpose, but nothing extraordinary.
Rating: 3
Predicted Sentiment: 1
Percentage of Negative: 0.34%
Percentage of Neutral: 99.43%
Percentage of Positive: 0.23%

Review: Terrible experience. The product broke within a week.
Rating: 1
Predicted Sentiment: 0
Percentage of Negative: 99.47%
Percentage of Neutral: 0.34%
Percentage of Positive: 0.19%

Review: Very satisfied with my purchase. Would recommend to others.
Rating: 5
Predicted Sentiment: 2
Percentage of Negative: 0.08%
Percentage of Neutral: 0.43%
Percentage of Positive: 99.49%

Review: Unimpressive. I expected more from a product at this price point.
Rating: 2
Predicted Sentiment: 0
Percentage of Negative: 98.66%
Percentage of Neutral: 1.16%
Percentage of Positive: 0.18%

Review: Outstanding! The best product I've ever owned.
Rating: 5
Predicted Sentiment: 2
Percentage of Negative: 0.05%
Percentage of Neutral: 0.02%
Percentage of Positive: 99.93%

Review: Not worth the money. Save yourself the disappointment.
Rating: 2
Predicted Sentiment: 0
Percentage of Negative: 99.09%
Percentage of Neutral: 0.72%
Percentage of Positive: 0.19%

Review: Highly disappointed. The product did not meet my expectations.
Rating: 1
Predicted Sentiment: 0
Percentage of Negative: 99.18%
Percentage of Neutral: 0.60%
Percentage of Positive: 0.22%

Review: Good, but not great. I expected a bit more.
Rating: 4
Predicted Sentiment: 1
Percentage of Negative: 0.38%
Percentage of Neutral: 97.24%
Percentage of Positive: 2.38%

Review: Top-notch quality. I'm extremely happy with my purchase.
Rating: 5
Predicted Sentiment: 2
Percentage of Negative: 0.03%
Percentage of Neutral: 0.04%
Percentage of Positive: 99.93%

Review: Wouldn't recommend. There are better options available.
Rating: 2
Predicted Sentiment: 0
Percentage of Negative: 97.87%
Percentage of Neutral: 1.92%
Percentage of Positive: 0.21%

Overall Review
Percentage of Negative: 40.00%
Percentage of Neutral: 20.00%
Percentage of Positive: 40.00%
```

A.2 Result of Testing and Evaluation with Application



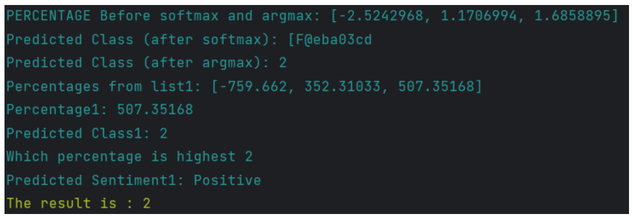
```
PERCENTAGE Before softmax and argmax: [-2.7611957, -0.989532, 4.2250566]
Predicted Class (after softmax): [F@4340c92
Predicted Class (after argmax): 2
Percentages from list1: [-582.12665, -208.61723, 890.7439]
Percentage1: 890.7439
Predicted Class1: 2
Which percentage is highest 2
Predicted Sentiment1: Positive
The result is : 2
```

A.2-1 Percentages of 1st Review and Rating in Table 6-2-3-1



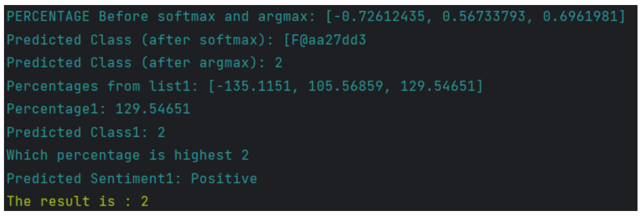
```
PERCENTAGE Before softmax and argmax: [-0.68591714, 2.0671558, -1.2134994]
Predicted Class (after softmax): [F@c8c73f0
Predicted Class (after argmax): 1
Percentages from list1: [-408.9186, 1232.3624, -723.44385]
Percentage1: 1232.3624
Predicted Class1: 1
Which percentage is highest 1
Predicted Sentiment1: Neutral
The result is : 1
```

A.2-2 Percentages of 2nd Review and Rating in Table 6-2-3-1



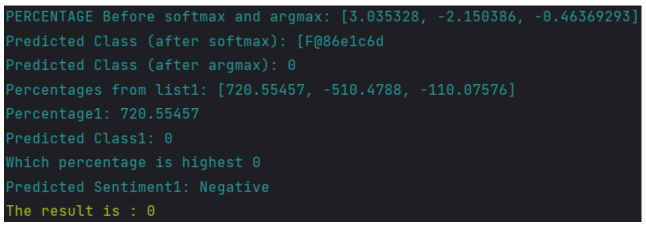
```
PERCENTAGE Before softmax and argmax: [-2.5242968, 1.1706994, 1.6858895]
Predicted Class (after softmax): [F@eba03cd
Predicted Class (after argmax): 2
Percentages from list1: [-759.662, 352.31033, 507.35168]
Percentage1: 507.35168
Predicted Class1: 2
Which percentage is highest 2
Predicted Sentiment1: Positive
The result is : 2
```

A.2-3 Percentages of 3rd Review and Rating in Table 6-2-3-1



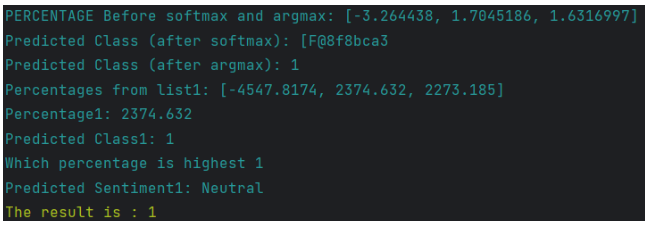
```
PERCENTAGE Before softmax and argmax: [-0.72612435, 0.56733793, 0.6961981]
Predicted Class (after softmax): [F@aa27dd3
Predicted Class (after argmax): 2
Percentages from list1: [-135.1151, 105.56859, 129.54651]
Percentage1: 129.54651
Predicted Class1: 2
Which percentage is highest 2
Predicted Sentiment1: Positive
The result is : 2
```

A.2-4 Percentages of 4th Review and Rating in Table 6-2-3-1



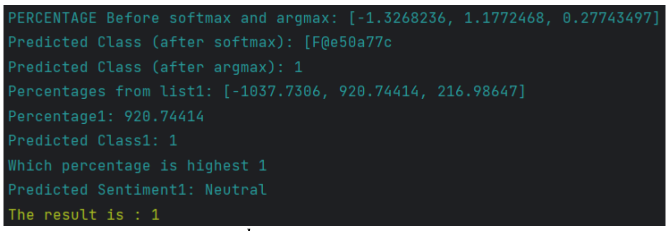
```
PERCENTAGE Before softmax and argmax: [3.035328, -2.150386, -0.46369293]
Predicted Class (after softmax): [F@86e1c6d
Predicted Class (after argmax): 0
Percentages from list1: [720.55457, -510.4788, -110.07576]
Percentage1: 720.55457
Predicted Class1: 0
Which percentage is highest 0
Predicted Sentiment1: Negative
The result is : 0
```

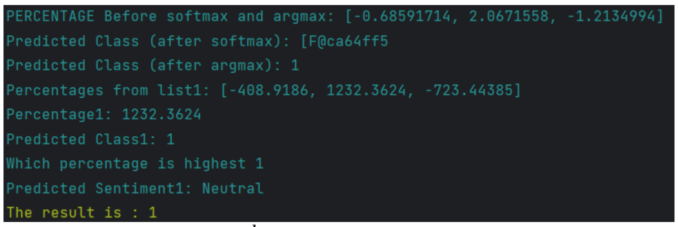
A.2-5 Percentages of 5th Review and Rating in Table 6-2-3-1

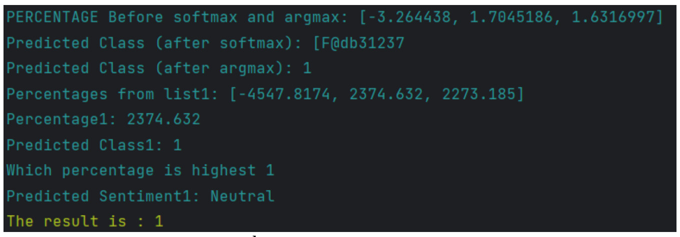


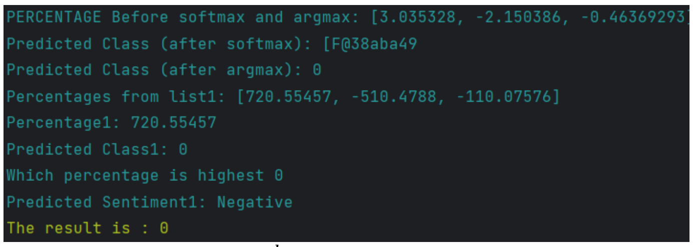
```
PERCENTAGE Before softmax and argmax: [-3.264438, 1.7045186, 1.6316997]
Predicted Class (after softmax): [F@8f8bca3
Predicted Class (after argmax): 1
Percentages from list1: [-4547.8174, 2374.632, 2273.185]
Percentage1: 2374.632
Predicted Class1: 1
Which percentage is highest 1
Predicted Sentiment1: Neutral
The result is : 1
```

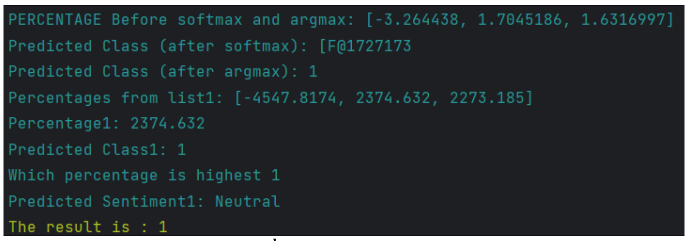
A.2-6 Percentages of 6th Review and Rating in Table 6-2-3-1

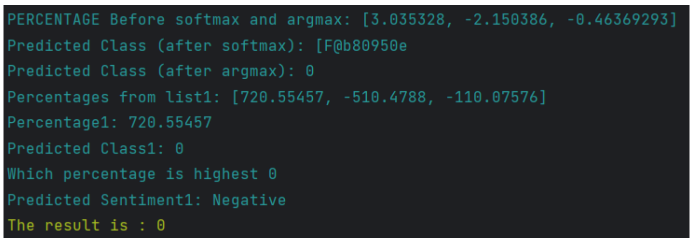

```
PERCENTAGE Before softmax and argmax: [-1.3268236, 1.1772468, 0.27743497]
Predicted Class (after softmax): [F@e50a77c
Predicted Class (after argmax): 1
Percentages from list1: [-1037.7306, 920.74414, 216.98647]
Percentage1: 920.74414
Predicted Class1: 1
Which percentage is highest 1
Predicted Sentiment1: Neutral
The result is : 1
```
A.2-7 Percentages of 7th Review and Rating in Table 6-2-3-1


```
PERCENTAGE Before softmax and argmax: [-0.68591714, 2.0671558, -1.2134994]
Predicted Class (after softmax): [F@ca64ff5
Predicted Class (after argmax): 1
Percentages from list1: [-408.9186, 1232.3624, -723.44385]
Percentage1: 1232.3624
Predicted Class1: 1
Which percentage is highest 1
Predicted Sentiment1: Neutral
The result is : 1
```
A.2-8 Percentages of 8th Review and Rating in Table 6-2-3-1

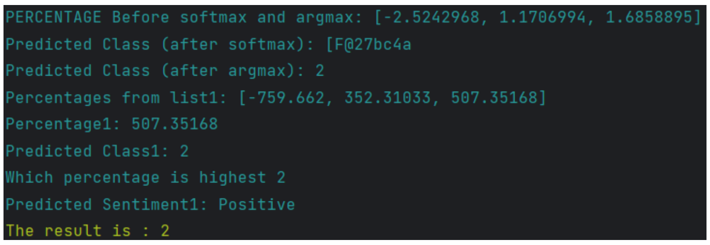

```
PERCENTAGE Before softmax and argmax: [-3.264438, 1.7045186, 1.6316997]
Predicted Class (after softmax): [F@db31237
Predicted Class (after argmax): 1
Percentages from list1: [-4547.8174, 2374.632, 2273.185]
Percentage1: 2374.632
Predicted Class1: 1
Which percentage is highest 1
Predicted Sentiment1: Neutral
The result is : 1
```
A.2-9 Percentages of 9th Review and Rating in Table 6-2-3-1


```
PERCENTAGE Before softmax and argmax: [3.035328, -2.150386, -0.46369293
Predicted Class (after softmax): [F@38aba49
Predicted Class (after argmax): 0
Percentages from list1: [720.55457, -510.4788, -110.07576]
Percentage1: 720.55457
Predicted Class1: 0
Which percentage is highest 0
Predicted Sentiment1: Negative
The result is : 0
```
A.2-10 Percentages of 10th Review and Rating in Table 6-2-3-1


```
PERCENTAGE Before softmax and argmax: [-3.264438, 1.7045186, 1.6316997]
Predicted Class (after softmax): [F@1727173
Predicted Class (after argmax): 1
Percentages from list1: [-4547.8174, 2374.632, 2273.185]
Percentage1: 2374.632
Predicted Class1: 1
Which percentage is highest 1
Predicted Sentiment1: Neutral
The result is : 1
```
A.2-11 Percentages of 11th Review and Rating in Table 6-2-3-1

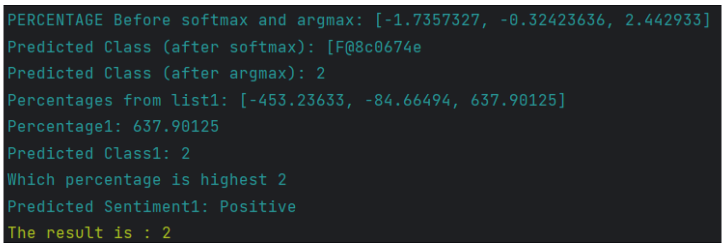

```
PERCENTAGE Before softmax and argmax: [3.035328, -2.150386, -0.46369293]
Predicted Class (after softmax): [F@b80950e
Predicted Class (after argmax): 0
Percentages from list1: [720.55457, -510.4788, -110.07576]
Percentage1: 720.55457
Predicted Class1: 0
Which percentage is highest 0
Predicted Sentiment1: Negative
The result is : 0
```

A.2-12 Percentages of 12th Review and Rating in Table 6-2-3-1



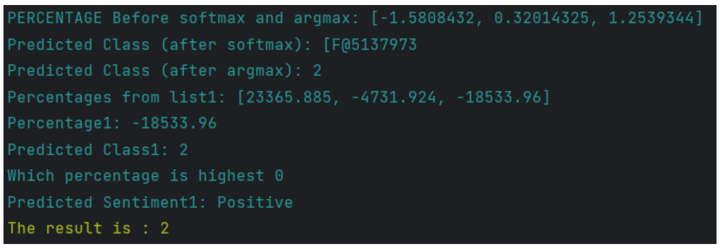
```
PERCENTAGE Before softmax and argmax: [-2.5242968, 1.1706994, 1.6858895]
Predicted Class (after softmax): [F@27bc4a
Predicted Class (after argmax): 2
Percentages from list1: [-759.662, 352.31033, 507.35168]
Percentage1: 507.35168
Predicted Class1: 2
Which percentage is highest 2
Predicted Sentiment1: Positive
The result is : 2
```

A.2-13 Percentages of 13th Review and Rating in Table 6-2-3-1



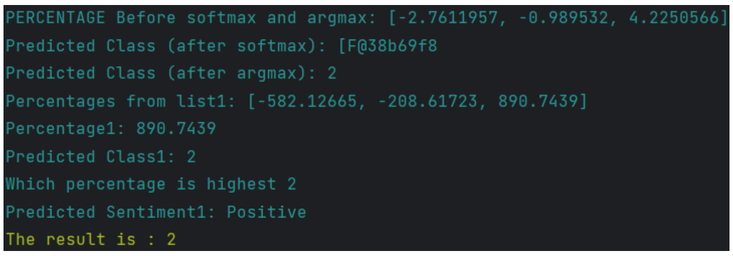
```
PERCENTAGE Before softmax and argmax: [-1.7357327, -0.32423636, 2.442933]
Predicted Class (after softmax): [F@8c0674e
Predicted Class (after argmax): 2
Percentages from list1: [-453.23633, -84.66494, 637.90125]
Percentage1: 637.90125
Predicted Class1: 2
Which percentage is highest 2
Predicted Sentiment1: Positive
The result is : 2
```

A.2-14 Percentages of 14th Review and Rating in Table 6-2-3-1



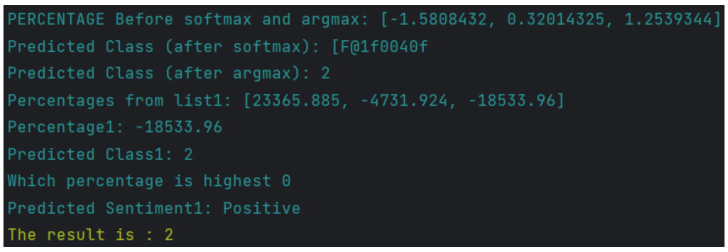
```
PERCENTAGE Before softmax and argmax: [-1.5808432, 0.32014325, 1.2539344]
Predicted Class (after softmax): [F@5137973
Predicted Class (after argmax): 2
Percentages from list1: [23365.885, -4731.924, -18533.96]
Percentage1: -18533.96
Predicted Class1: 2
Which percentage is highest 0
Predicted Sentiment1: Positive
The result is : 2
```

A.2-15 Percentages of 15th Review and Rating in Table 6-2-3-1



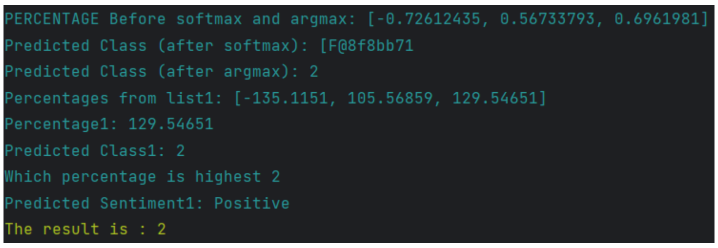
```
PERCENTAGE Before softmax and argmax: [-2.7611957, -0.989532, 4.2250566]
Predicted Class (after softmax): [F@38b69f8
Predicted Class (after argmax): 2
Percentages from list1: [-582.12665, -208.61723, 890.7439]
Percentage1: 890.7439
Predicted Class1: 2
Which percentage is highest 2
Predicted Sentiment1: Positive
The result is : 2
```

A.2-16 Percentages of 16th Review and Rating in Table 6-2-3-1



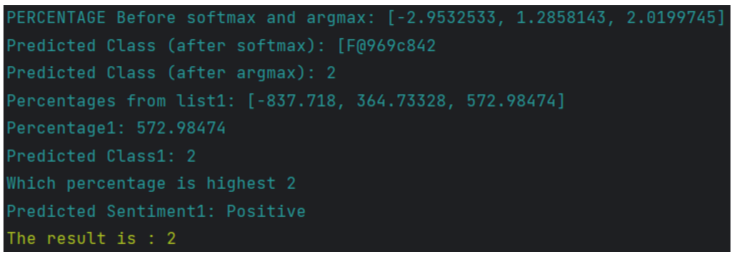
```
PERCENTAGE Before softmax and argmax: [-1.5808432, 0.32014325, 1.2539344]
Predicted Class (after softmax): [F@1f0040f
Predicted Class (after argmax): 2
Percentages from list1: [23365.885, -4731.924, -18533.96]
Percentage1: -18533.96
Predicted Class1: 2
Which percentage is highest 0
Predicted Sentiment1: Positive
The result is : 2
```

A.2-17 Percentages of 17th Review and Rating in Table 6-2-3-1



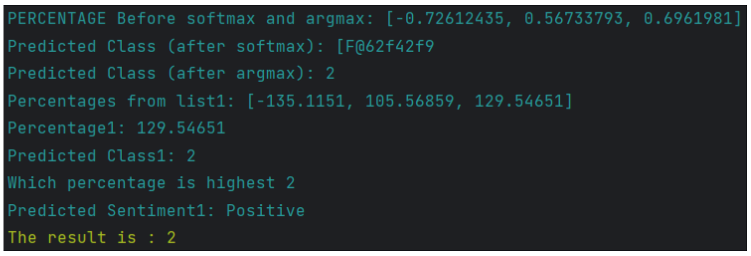
```
PERCENTAGE Before softmax and argmax: [-0.72612435, 0.56733793, 0.6961981]
Predicted Class (after softmax): [F@8f8bb71
Predicted Class (after argmax): 2
Percentages from list1: [-135.1151, 105.56859, 129.54651]
Percentage1: 129.54651
Predicted Class1: 2
Which percentage is highest 2
Predicted Sentiment1: Positive
The result is : 2
```

A.2-18 Percentages of 18th Review and Rating in Table 6-2-3-1



```
PERCENTAGE Before softmax and argmax: [-2.9532533, 1.2858143, 2.0199745]
Predicted Class (after softmax): [F@969c842
Predicted Class (after argmax): 2
Percentages from list1: [-837.718, 364.73328, 572.98474]
Percentage1: 572.98474
Predicted Class1: 2
Which percentage is highest 2
Predicted Sentiment1: Positive
The result is : 2
```

A.2-19 Percentages of 19th Review and Rating in Table 6-2-3-1



```
PERCENTAGE Before softmax and argmax: [-0.72612435, 0.56733793, 0.6961981]
Predicted Class (after softmax): [F@62f42f9
Predicted Class (after argmax): 2
Percentages from list1: [-135.1151, 105.56859, 129.54651]
Percentage1: 129.54651
Predicted Class1: 2
Which percentage is highest 2
Predicted Sentiment1: Positive
The result is : 2
```

A.2-20 Percentages of 20th Review and Rating in Table 6-2-3-1

A.3 Poster

SHOPPING RECOMMENDER SYSTEM FOR UNIVERSITY STUDENTS

INTRODUCTION

This project focus on building a recommender system by utilizing the result from sentiment analysis. BERT is a model that are used to perform sentiment analysis or natural language processing in machine learning. The algorithm will mainly use two data: Review/Comment and Rating/Start. Which can give a high accurate result.

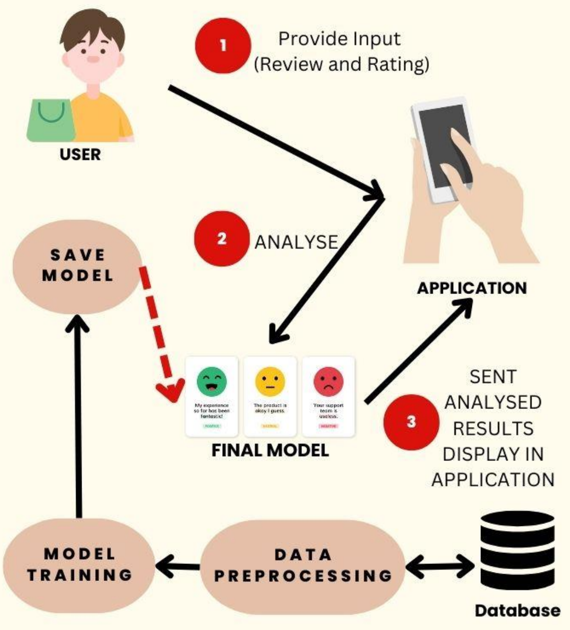
OBJECTIVES

1. 1. To develop a system that can use the result of sentiment analysis of customers' reviews and comments to classify the shops into different groups.
2. 2. To develop a system that utilize the result of sentiment analysis to improve customer purchasing experience and satisfaction.
3. 3. To develop a system that achieve at least a minimum 80% average of accuracy to the unseen data which is testing dataset.

METHODS

Rapid Application Development: Throw-away Prototyping and Agile Development Methodology, both provide a cycles and flexibility to developer to develop the system including: Planning, Analysis, Design, Implementation, System. Both methodology complement to each other to provide a greater benefits to developer.

OVERVIEW



CONCLUSION

The recommender system will build by using BERT model to identify the negativity, positivity, and neutral of the customer review and rating. Which is a Machine Learning method.

SUPERVISOR

DR ABDULKARIM KANAAN JEBNA

BY LIM WOON SHENG (20ACB03109)

A.4 Weekly Report

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 2**
Student Name & ID: Lim Woon Sheng 20ACB03109	
Supervisor: Dr Abdulkarim Kanaan Jebna	
Project Title: Shopping Recommender System for University Students	

1. WORK DONE
[Please write the details of the work done in the last fortnight.]

Code of the model preprocess and training is completed.
Basic activity of the application is completed.

2. WORK TO BE DONE

Model Training and Testing Evaluation
Model Saving
Apply model saved in android studio.
Write code of function of using the model
Function of Shop information displaying in application

3. PROBLEMS ENCOUNTERED

Process of model training taking many times due to the amount of data.

4. SELF EVALUATION OF THE PROGRESS

Progressing but the main part of the project which is the model is having issues in order to have perfection. Hence, there is only 10% progress.

[illegible] Sheng

Supervisor's signature Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 4**
Student Name & ID: Lim Woon Sheng 20ACB03109	
Supervisor: Dr Abdulkarim Kanaan Jebna	
Project Title: Shopping Recommender System for University Students	

1. WORK DONE
[Please write the details of the work done in the last fortnight.]

Model saving is completed.

2. WORK TO BE DONE

Apply model saved in android studio.
Write code of function of using the model
Function of Shop information displaying in application

3. PROBLEMS ENCOUNTERED

Model saved is in large size of file which not able to use in android studio

4. SELF EVALUATION OF THE PROGRESS

Main part of the project which is the model is completed, but the other part which is application is still progressing. Hence, there is only 30% progress.

[illegible] Sheng

Supervisor's signature　　　　Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

Trimester, Year: Y3S3	**Study week no.: 7**
Student Name & ID: Lim Woon Sheng 20ACB03109	
Supervisor: Dr Abdulkarim Kanaan Jebna	
Project Title: Shopping Recommender System for University Students	

1. WORK DONE
[Please write the details of the work done in the last fortnight.]

Function of Shop information displaying in application

2. WORK TO BE DONE

Model application in android studio.
Perfection on the application interface
Model training with a larger size of data.

3. PROBLEMS ENCOUNTERED

Model saved is in large size of file which not able to use in android studio.
The usage of model saved in application are having large inconsistency. Which not able to perform well in android studio.

4. SELF EVALUATION OF THE PROGRESS

Main part of the project which is the model is completed, and application is able to perform well in terms of showing information from Firebase into application. Hence, there is only 50% progress.

[illegible] Sheng

Supervisor's signature Student's signature

FINAL YEAR PROJECT WEEKLY REPORT

(Project II)

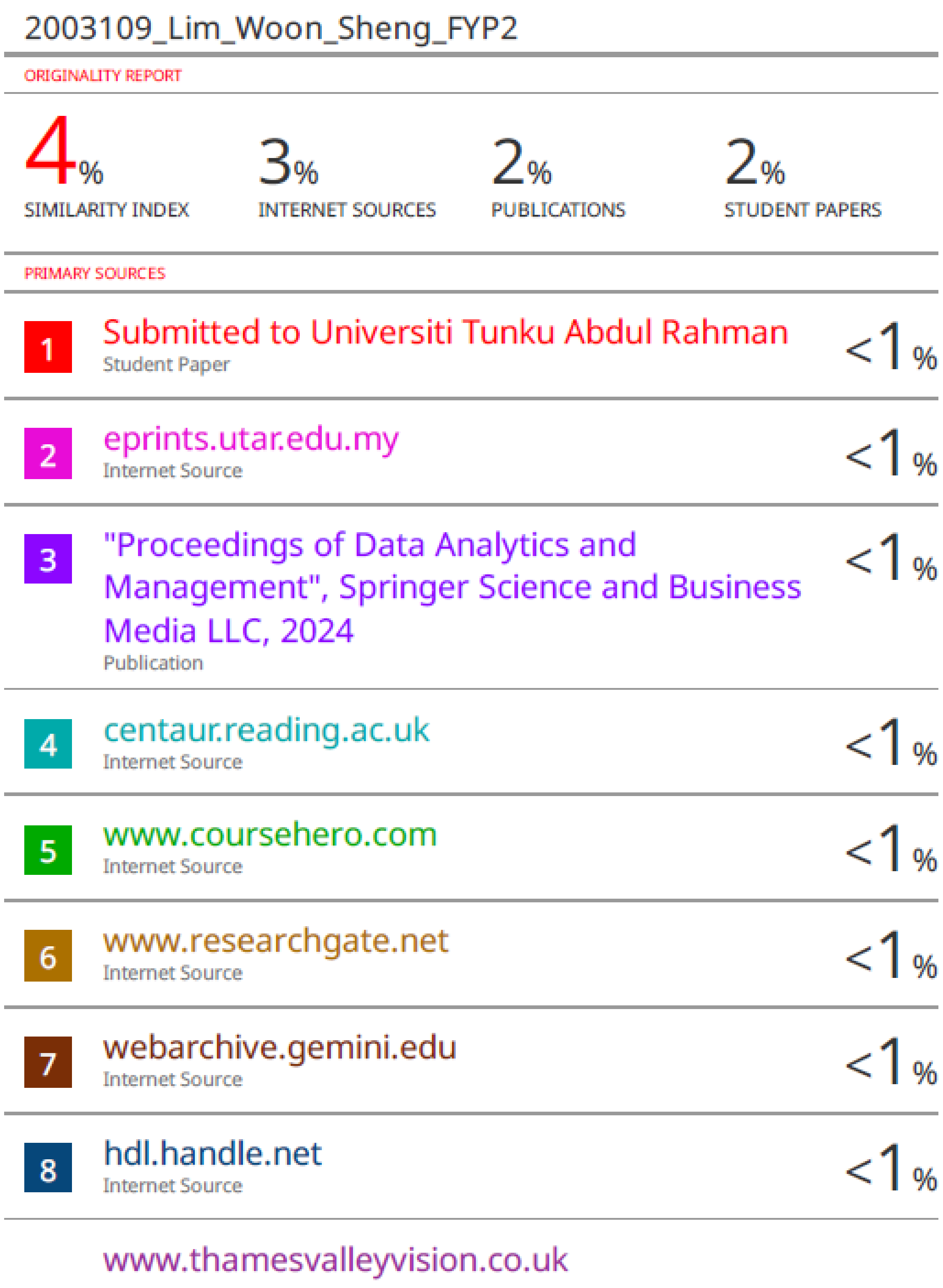
Trimester, Year: Y3S3	**Study week no.: 10**
Student Name & ID: Lim Woon Sheng 20ACB03109	
Supervisor: Dr Abdulkarim Kanaan Jebna	
Project Title: Shopping Recommender System for University Students	

1. WORK DONE [Please write the details of the work done in the last fortnight.] Perfection on the application interface
2. WORK TO BE DONE Report writing Model training with a larger size of data.
3. PROBLEMS ENCOUNTERED Model trained did not perform well due to the size of data used The usage of model saved in application are having large inconsistency. Which not able to perform well in android studio.
4. SELF EVALUATION OF THE PROGRESS Main part of the project which is the model is completed, and application is able to perform well in terms of showing information from Firebase into application. Hence, there is only 60% progress.

[illegible] Sheng

Supervisor's signature Student's signature

PLAGIARISM CHECK RESULT



2003109_Lim_Woon_Sheng_FYP2

ORIGINALITY REPORT

4%	3%	2%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	Submitted to Universiti Tunku Abdul Rahman Student Paper	<1%
2	eprints.utar.edu.my Internet Source	<1%
3	"Proceedings of Data Analytics and Management", Springer Science and Business Media LLC, 2024 Publication	<1%
4	centaur.reading.ac.uk Internet Source	<1%
5	www.coursehero.com Internet Source	<1%
6	www.researchgate.net Internet Source	<1%
7	webarchive.gemini.edu Internet Source	<1%
8	hdl.handle.net Internet Source	<1%
	www.thamesvalleyvision.co.uk	

9	Internet Source	<1%
10	Submitted to Sim University Student Paper	<1%
11	www.mdpi.com Internet Source	<1%
12	downloads.hindawi.com Internet Source	<1%
13	Ismael Ramos-Pérez, Álvar Arnaiz-González, Juan J. Rodríguez, César García-Osorio. "When is resampling beneficial for feature selection with imbalanced wide data?", Expert Systems with Applications, 2022 Publication	<1%
14	Li Yang, Ying Li, Jin Wang, R. Simon Sherratt. "Sentiment Analysis for E-Commerce Product Reviews in Chinese Based on Sentiment Lexicon and Deep Learning", IEEE Access, 2020 Publication	<1%
15	Muhammad Ismail, Shahzad Memon, Lachhman Das Dhomeja, Shahid Munir Shah, Dostdar Hussain, Sabit Rahim, Imran Ali. "Development of a regional voice dataset and speaker classification based on machine learning", Journal of Big Data, 2021 Publication	<1%

16	Submitted to University of Hertfordshire Student Paper	<1 %
17	androidstudio.googleblog.com Internet Source	<1 %
18	www.3gpp.org Internet Source	<1 %
19	www.sciendo.com Internet Source	<1 %
20	dokumen.pub Internet Source	<1 %
21	Submitted to University of Central England in Birmingham Student Paper	<1 %
22	Submitted to University of Warwick Student Paper	<1 %
23	vdoc.pub Internet Source	<1 %
24	www.igi-global.com Internet Source	<1 %
25	www.numdam.org Internet Source	<1 %
26	Submitted to Liverpool John Moores University Student Paper	<1 %

27	ejurnal.poliban.ac.id Internet Source	<1 %
28	www.ijisae.org Internet Source	<1 %
29	www.sweetstudy.com Internet Source	<1 %
30	Submitted to University of Wollongong Student Paper	<1 %
31	isset.shiksha.com Internet Source	<1 %
32	core.ac.uk Internet Source	<1 %
33	subscription.packtpub.com Internet Source	<1 %
34	www.baeldung.com Internet Source	<1 %
35	www.journal-isi.org Internet Source	<1 %
36	Submitted to KEDGE Business Schools Student Paper	<1 %
37	fdocuments.net Internet Source	<1 %
38	github.com Internet Source	<1 %

39	www.stet-review.org Internet Source	<1 %
40	Gotelli, Nicholas J.. "A Primer of Ecological Statistics", Oxford University Press Publication	<1 %
41	Liu, Y.. "Combining integrated sampling with SVM ensembles for learning from imbalanced datasets", Information Processing and Management, 201107 Publication	<1 %
42	Nyein Ei Ei Kyaw, Thinn Thinn Wai. "Inferring User Preferences Using Reviews for Rating Prediction", 2019 International Conference on Advanced Information Technologies (ICAIT), 2019 Publication	<1 %
43	Rhoda Viviane Achieng Ogutu, Richard M. Rimiru, Calvins Otieno. "Target Sentiment Analysis Ensemble for Product Review Classification", Journal of Information Technology Research, 2022 Publication	<1 %
44	digital.library.unt.edu Internet Source	<1 %
45	ejurnal.seminar-id.com Internet Source	<1 %

icons8.com

46	Internet Source	<1%
47	repository.smuc.edu.et Internet Source	<1%
48	search.library.wisc.edu Internet Source	<1%
49	technoaretepublication.org Internet Source	<1%
50	uir.unisa.ac.za Internet Source	<1%
51	www.taafoo.com Internet Source	<1%
52	Paolo Dell'Aversana. "Enhancing Deep Learning and Computer Image Analysis in Petrography through Artificial Self-Awareness Mechanisms", Minerals, 2024 Publication	<1%
53	Xiaohong Wang, Shuang Dong. "Users' Sentiment Analysis of Shopping Websites Based on Online Reviews", Applied Mathematics and Nonlinear Sciences, 2020 Publication	<1%
54	Zulfadzli Drus, Haliyana Khalid. "Sentiment Analysis in Social Media and Its Application: Systematic Literature Review", Procedia Computer Science, 2019	<1%

Publication

55	Kwabena Adu, Yongbin Yu, Jingye Cai, Victor Dela Tattrah, James Adu Ansere, Nyima Tashi. "S-CCCapsule: Pneumonia detection in chest X-ray images using skip-connected convolutions and capsule neural network", Journal of Intelligent & Fuzzy Systems, 2021 Publication	<1%

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		

Universiti Tunku Abdul Rahman			
Form Title : Supervisor's Comments on Originality Report Generated by Turnitin for Submission of Final Year Project Report (for Undergraduate Programmes)			
Form Number: FM-IAD-005	Rev No.: 0	Effective Date: 01/10/2013	Page No.: 1of 1

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY

Full Name(s) of Candidate(s)	Lim Woon Sheng
ID Number(s)	20ACB03109
Programme / Course	Business Information System (IB)
Title of Final Year Project	Shopping Recommender System for University Students

Similarity	**Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)**
Overall similarity index: 4 **%** **Similarity by source** Internet Sources: 3 % Publications: 2 % Student Papers: 2 %	
Number of individual sources listed of more than 3% similarity: 0	
Parameters of originality required and limits approved by UTAR are as Follows: **(i) Overall similarity index is 20% and below, and** **(ii) Matching of individual sources listed must be less than 3% each, and** **(iii) Matching texts in continuous block must not exceed 8 words** ***Note: Parameters (i) – (ii) shall exclude quotes, bibliography and text matches which are less than 8 words.***	

Note Supervisor/Candidate(s) is/are required to provide softcopy of full set of the originality report to Faculty/Institute

Based on the above results, I hereby declare that I am satisfied with the originality of the Final Year Project Report submitted by my student(s) as named above.

[illegible]

Signature of Supervisor	Signature of Co-Supervisor
Name: Dr Abdulkarim Kanaan Jebna	Name:
Date: 24 April 2024	Date:

UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY
(KAMPAR CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	20ACB03109
Student Name	Lim Woon Sheng
Supervisor Name	Dr Abdulkarim Kanaan Jebna

TICK (√)	**DOCUMENT ITEMS** Your report must include all the items below. Put a tick on the left column after you have checked your report with respect to the corresponding item.
√	Title Page
√	Signed Report Status Declaration Form
√	Signed FYP Thesis Submission Form
√	Signed form of the Declaration of Originality
√	Acknowledgement
√	Abstract
√	Table of Contents
√	List of Figures (if applicable)
√	List of Tables (if applicable)
√	List of Symbols (if applicable)
√	List of Abbreviations (if applicable)
√	Chapters / Content
√	Bibliography (or References)
√	All references in bibliography are cited in the thesis, especially in the chapter of literature review
√	Appendices (if applicable)
√	Weekly Log
√	Poster
√	Signed Turnitin Report (Plagiarism Check Result - Form Number: FM-IAD-005)
√	I agree 5 marks will be deducted due to incorrect format, declare wrongly the ticked of these items, and/or any dispute happening for these items in this report.

*Include this form (checklist) in the thesis (Bind together as the last page)

I, the author, have checked and confirmed all the items listed in the table are included in my report.

Sheng

____
(Signature of Student)
Date: 24 April 2024