MAKER SPACE STORAGE MANAGEMENT

BY

TAN WAN JING

A REPORT

SUBMITTED TO

Universiti Tunku Abdul Rahman

in partial fulfillment of the requirements

for the degree of

BACHELOR OF COMPUTER SCIENCE (HONOURS)

Faculty of Information and Communication Technology

(Kampar Campus)

JAN 2022

UNIVERSITI TUNKU ABDUL RAHMAN

REPORT STATUS DECLARATION FORM

Title: Development of Maker Space Storage Management System

Academic Session: Jan 2022

I TAN WAN JING

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,

[illegible]	[illegible]
(Author's signature)	(Supervisor's signature)

Address:
94, Lorong 17/SS1,
Bandar Tasek Mutiara,
14120 Simpang Ampat, Penang

Ts Dr Ooi Boon Yaik
Supervisor's name

Date: 21/04/2022

Date: 21/4/2022

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 COMMUNICATION TECHNOLOGY

UNIVERSITY TUNKU ABDUL RAHMAN

Date: 12/04/2022

SUBMISSION OF FINAL YEAR PROJECT

It is hereby certified that Tan Wan Jing (ID No: 18ACB01284) has completed this final year project entitled "Maker Space Storage Management" under the supervision of Ts Dr Ooi Boon Yaik from the Department of Computer Science, 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,

[illegible]

Tan Wan Jing

DECLARATION OF ORIGINALITY

I declare that this report entitled "**MAKER SPACE STORAGE MANAGEMENT**" 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	:	[illegible]
Name	:	Tan Wan Jing
Date	:	21/04/2022

ACKNOWLEDGEMENTS

I would want to express my heartfelt gratitude to my supervisor, Ts Dr Ooi Boon Yaik, for providing me with this wonderful opportunity to work on a Makerspace Storage Management project. It is the first step in my profession as an Android mobile application developer. I do also like to express my gratitude for his patience and unwavering support. Finally, I want to express my gratitude to my parents and family for their unwavering love, support, and encouragement throughout the process.

ABSTRACT

In this project, StoreHut, a Maker Space Storage Management mobile application for Android is being developed to solve the existence problems. Currently, makers are facing the problem of storage in managing their tools, equipment, and stock. Fixed location storage can ensure the neatness of the object and help maker to easily find the items. However, it is time-consuming, lacking of flexibility, and not space-efficient. Moreover, the implementation of chaotic storage system is not a new technique in storage management especially in the warehousing process due to its time-saving and space-efficient characteristics. However, in existing approaches, various resources such as RFID tags, RFID scanners, complex management system are needed to track and manage the objects in chaotic storage system. Therefore, a mobile application integrated with QR Code generator and scanner is being developed to produce a portable, simple, affordable and straightforward chaotic storage management system for solving the problems in current system. Moreover, an e-commerce system is built in this application which are specially created for the maker to buy, sell or rent the equipment and ingredients. The development of this project will have impacts and contribution to the population of makerspace and promotes the usage of chaotic storage system in makerspaces as it could save more time, flexible and space-efficient.

TABLE OF CONTENT

TITLE PAGE ii

REPORT STATUS DECLARATION FORM iii

FYP THESIS SUBMISSION FORM iv

DECLARATION OF ORIGINALITY iv

ACKNOWLEDGEMENTS v

ABSTRACT vi

LIST OF FIGURES x

LIST OF TABLE xiii

LIST OF ABBREVIATIONS xiv

CHAPTER 1 INTRODUCTION 1

1.1 Background information 1

1.2 Problem Statement and Motivation 1

1.3 Project Objective 3

1.4 Project Scope 3

1.4.1 Location Registration using QR Code Generator 4

1.4.2 QR Code Scanner 4

1.4.3 Listing of Objects and Storage box 4

1.4.4 E-commerce platform 4

1.5 Impact, Significance and Contribution 5

CHAPTER 2 LITERATURE REVIEW 6

2.1 Literature Review 6

2.1.1 Warehouse management model using FEFO, 5s, and chaotic storage to improve product loading times in small- and medium-sized non-metallic mining companies 6

2.2.2 Integrated production planning and warehouse storage assignment problem: An IoT assisted case 7

2.2.3 Design of a reference architecture for developing smart warehouses in industry 4.0 8

2.2.4 The Construction of a Consumer To Consumer E-commerce System Based on Japanese Elderly 9

2.2.5 Innovation Management And Automated Accounting In The Chaotic Storage Logistics 10
2.2 Review and Comparison of Previous Work 11
2.2.1 NFC Part Management System 11
2.2.2 Sortly 12
2.2.3 Cashier Live 13
2.2.4 Carousell 14
2.2.5 Shopee 15
2.3 Critical Remarks of Previous Works 16
CHAPTER 3 SYSTEM METHODOLOGY AND SYSTEM DESIGN 17
3.1 System methodology 17
3.2 System Design Diagram 18
3.2.1 System Architecture Diagram 18
3.2.2 Use Case Diagram 20
3.1.3 Use Case Description 21
3.2.4 Sequence Diagram 28
3.3 System Design 37
3.3.1 System Flow Chart 37
3.3.2 ERD Diagram 39
3.3.3 Data Dictionary 39
CHAPTER 4 SYSTEM IMPLEMENTATION 45
4.1 Project Timeline 45
4.2 Technologies and tools involved 45
4.2.1 Hardware requirement 45
4.2.2 Software requirement 45
4.2.3 Connectivity Requirement 46
4.3 Code Snippet 47
4.3.1 Code snippet of the procedure on uploading data and image to Firebase Realtime Database and Firebase Storage respectively. 47
4.3.2 Code snippet of the procedure on generating QR Code 49
4.3.3 Code snippet of the procedure on creating QR Code Scanner 50
4.3.5 Code snippet of producing scrollable grid view. 52
4.4 User Interface 55

4.4.1 Dashboard 55
4.4.2 QR Code Generator 56
4.4.3 QR Code Scanner 56
4.4.4 Item and Storage Location List 57
4.4.5 StoreHut Mart Interface 57
4.4.6 MyStore Interface 58
CHAPTER 5 SYSTEM EVALUATION AND DISCUSSION 59
5.1 Software Testing 59
5.2 User Feedback Analysis 59
CHAPTER 6 CONCLUSION 65
6.1 Review of Project 65
6.2 Novelties and Contributions 65
6.3 Limitation 66
6.4 Future Work 66
REFERENCES 657
Appendix A : User Feedback Form Template A-1
Appendix B : Final Year Project Weekly Report B-1
Appendix C : Poster 75
Appendix D: Plagiarism Check Result D-1

LIST OF FIGURES

Figure 2.2.2.1 The detail of Pseudocode for location assignment.	8
Figure 2.2.3.1 Feature Diagram of Scanner	9
Figure 2.2.3.2: Feature Diagram of Warehouse Management System	10
Figure 2.2.4.1 Business Process Online	11
Figure 2.2.1.1 Image of NFC Part Management System	12
Figure 2.2.2.1 User Interface of Sortly application	13
Figure 2.2.3.1 User interface of Cashier Live application	14
Figure 2.2.4.1 User interface of Carousell	15
Figure 2.2.5.1 User interface of Shopee	16
Figure 3.1.1 SDLC – Model of Agile development	18
Figure 3.2.1.1 Architecture Diagram	19
Figure 3.2.2.1 Use Case Diagram	21
Figure 3.2.4.1 Generate QR Code Sequence Diagram	29
Figure 3.2.4.2 Scan QR Code Sequence Diagram	30
Figure 3.2.4.3 Keep Item Sequence Diagram	31
Figure 3.2.4.4 Withdraw Item Sequence Diagram	32
Figure 3.2.4.5 View Item List Sequence Diagram	33
Figure 3.2.4.6 View Storage Location List Sequence Diagram	34
Figure 3.2.4.7 Add Product Into Cart Sequence Diagram	35
Figure 3.2.4.8 Sequence Diagram of View On-Shelf Product	36
Figure 3.2.4.9 Sequence Diagram of Add New Product to Shelf	37
Figure 3.3.1.1 System Flow Chart	38
Figure 3.3.2.1 Entity Relationship Diagram	40
Figure 3.3.3.1 Data dictionary of Users table	41
Figure 3.3.3.1 Data dictionary of Item table	42

Figure 3.3.3.1 Data dictionary of Storage table 43
Figure 3.3.3.1 Data dictionary of MyMartProduct table 44
Figure 3.3.3.1 Data dictionary of MyCart table 45
Figure 4.3.1.1 Code snippet of uploading data to Firebase Realtime Database 48
Figure 4.3.1.2 Code snippet of uploading Image to Firebase Storage 49
Figure 4.3.1.3 Code snippet of getting image Uri 49
Figure 4.3.2.1 Code snippet of generating QR Code with a given key of storage location 50
Figure 4.3.3.1 Code snippet of creating QR Scanner 51
Figure 4.3.4.1 Code snippet of handling situation where no image attached 52
Figure 4.3.4.2 Code snippet of handling situation where original image is changed to another one. 52
Figure 4.3.4.2 Code snippet of handling situation where original image is changed to another one. 53
Figure 4.3.5.1 Components of RecyclerView 53
Figure 4.3.5.2 Code snippet of getting the available product from Realtime Database and add into the array list for further usage. 54
Figure 4.3.5.3 Code snippet of designing the UI of Grid Layout which is used to show the products in StoreHut Mart 54
Figure 4.3.5.4 Implementation of GridLayoutManager and RecyclerAdapter into recycler view 55
Figure 4.3.5.5 Code snippet of ViewHolder of adapter class. 55
Figure 4.3.5.6 Code snippet of onClick() listener implemented on each component of recycler view. 56
Figure 4.4.1.1 UI of Dashboard 56
Figure 4.4.2.1 UI of QR Code Generator 57
Figure 4.4.3.1 UI of QR Code Scanner 57
Figure 4.4.4.1 UI of Item and Storage Location List 58
Figure 4.4.5.1 UI of StoreHut Mart Interface 58

Figure 4.4.6.1 UI of MyStore Interface 59
Figure 5.2.1 Respondent’s Occupation 61
Figure 5.2.2 The feedback given on how does StoreHut Management System help the respondents 62
Figure 5.2.3 The result of rating the system UI design. 62
Figure 5.2.4 The result of rating the system intuitiveness. 62
Figure 5.2.5 The result of rating the usefulness of QR Code implementation in the system 63
Figure 5.2.6 The result of rating the effectiveness of the system on tracking the equipment or items. 63
Figure 5.2.7 The result of rating the usefulness of renting and selling service provided. 64
Figure 5.2.8 The list of suggestion provided by each of the respondents. 64
Figure A-1 User feedback form A-1

LIST OF TABLE

Table 2.3.1 Summaries of Application Feature of Storage Management System 17

Table 2.3.2 Summaries of Application Feature of Local E-Commmerce Application 17

Table 3.2.3.1 Use Case Description of QR Code 22

Table 3.2.3.2 Use Case Description of Add new item 23

Table 3.2.3.3 Use Case Description of Keep item 23

Table 3.2.3.4 Use Case Description of Withdraw item 24

Table 3.2.3.5 Use Case Description of View item list 25

Table 3.2.3.6 Use Case Description of View storage location list 26

Table 3.2.3.7 Use Case Description of Add product into cart 26

Table 3.2.3.8 Use Case Description of Contact seller of product 26

Table 3.2.3.9 Use Case Description of View on-shelf products 27

Table 3.2.3.10 Use Case Description of Add new product to shelf 28

Table 4.1.1 Project Timeline 46

Table 4.2.1.1 Hardware requirement 46

LIST OF ABBREVIATIONS

RFID	Radio-Frequency Identification
QR Code	Quick Response Code
UI	User Interface
App	Application

CHAPTER 1 INTRODUCTION

1.1 Background information

A maker space would have a variety of tools, equipment and ingredients prepared and utilized by the maker. Hence, a good storage of those objects could save time and space while still improving the convenience of tasks and jobs. In our research, we are focusing on the study of chaotic storage system by understanding its benefits and challenges, as well as developing a storage application which implements chaotic storage approach.

A storage approaches that place the objects randomly and disorderedly in a storage location is known as chaotic storage. It is the opposite of fixed location storage where location for each objects are predefined and the allocation of object must strictly follow the predetermined location. Although the chaotic storage required less storage space and time-saving, it definitely need a management system to help identify, track and manage the objects within a storage centre [9].

Amazon, the world's largest online retailer, is known for its lightning-fast inventory system. The Amazon warehouse's inventory system is a chaotic storage system, in which things are virtually and disorderly kept in the warehouse. While owning the world's largest storage space, Amazon had tackled the logistical challenge by deploying automated inventory and warehouse management systems that use barcodes and scanners to create a system known as chaotic storage. The incoming products are placed in the available shelves at random, with no regard for their location. Simultaneously, a systematic and comprehensive inventory management system (IMS) is employed to control each product's storage location. The warehouse efficiency and inventory process have both improved as a result of this strategy.

1.2 Problem Statement and Motivation

In makerspace, fixed location storage strategy are lacking of flexibility, time-consuming and not space-efficient, while the existing approaches to manage chaotic storage system is complex and need more resources. First and foremost, the fixed location storage strategy required the maker to place the object in a fixed location that are specific and pre-defined [1]. Hence, maker is not allowed to place the objects anywhere, else it will end up with unorganized condition again. When the size of the makerspace is larger and the amount of objects are increasing, the maker might need more time to retrieve and allocate every single objects to the predetermined location without messing it up [1]. Besides, objects are not allowed to be placed in the location other than the predefined one even there is space available.

Secondly, the location of the objects is hardly tracked when they are placed randomly and disorderly in chaotic storage approach. Although the existing warehouse management system (WMS) can systematically track and manage the objects in chaotic storage system, it is not suitable to be implemented in a makerspace due to the complexity of implementation and more resources need to be prepared such as RFID tags, RFID scanners, complex management system and so on.

Thirdly, some of the equipment could cost hundreds or thousands and they are mostly reusable and valuable. However, it might be a form of resources wastage if it is seldom used and gradually lost its value.

By developing a simple and straightforward mobile-based makerspace storage management application, it can increase the flexibility of storing the tools and equipment because it allows random placing of objects into any empty space. Hence, less space is wasted as well as utilizing the space better. Not only the space is being saved, the time to organize, keep and retrieve the objects is reduced too. With the implementation of chaotic storage, the makers do not need to spend much time in organizing the locations to keep their tools and equipment, for example placing the items accordingly and remember the exactly place where the items are located. Other than that, the application can tell the user the exact location of the object accurately with just a few clicks. Furthermore, the user can rent or sell the resources they had such as the equipment, tools, ingredients and so on. It is also promoting the value of reuse and reduce since equipment and tools can be shared among the makers without purchasing a brand new item. Lastly, the storage management system is easily implemented.

1.3 Project Objective

1. To analyse the existing approaches used in tracking object location in random chaotic storage system at makerspace.
   The functionalities, characteristics and efficacy of existing chaotic storage management system will be evaluated in order to better understand how to design and develop a better storage management application in terms of value, usability and accessibility.
2. To design an application in mobile device that helps tracking object location in random chaotic storage system at makerspace via simple and straightforward way.
   We aimed to design and develop a mobile application that can track the location of objects when they are placed randomly and disorderedly in a storage location. We would like to produce the system that could be easily implemented and understood by the user while effectively managing the random and chaotic storage of objects.
3. To design a mobile application that helps user to purchase or rent equipment and ingredients.
   We want to develop a feature that allow users to easily sell, rent and buy equipment and ingredients.
4. To evaluate the usability of the proposed mobile application in tracking object location in random chaotic storage system at makerspace.
   In order to develop a mobile application that could achieve higher customer satisfaction, the user requirements will be reviewed, and the application will be assessed via validation and verification throughout the software testing process.

1.4 Project Scope

We are going to develop a maker space storage management mobile application, named StoreHut. Instead of predefining the location of certain object, the proposed application allowed user to place the object anywhere, as long as the location is registered in the application. The steps of location registration will be further discussed in the following session. The key features of the mobile application are QR Code generator, QR Code scanner, listing of objects and storage location, tracking of object's location, and e-commerce platform for equipment renting and selling purpose.

1.4.1 Location Registration using QR Code Generator

A QR Code generator is embedded in the application to ease the user for customizing his own QR code. It is a process of location registration in which the certain information of the storage container or location is prompted and embedded in the QR Code. After generating the QR Code, it can be downloaded into the local storage of smart phone. Other than that, the generated QR Code could be printed out and attached to the storage box, shelf or other storage location.

1.4.2 QR Code Scanner

QR code scanner is used to retrieve the storage location key embedded in the QR Code attached to the storage box. After scanning the QR Code, user can select the action he would like to perform towards the respective storage box, either adding new objects into it, taking out objects from it or placing object back into it.

1.4.3 Listing of Objects and Storage box

After adding or registering a new object, it will automatically be added into the list of objects. Besides, the list will be displayed on a screen neatly and informing the corresponding location of each objects. Not only that, the storage locations will be listed out in another screen while informing the user about the objects kept in the specific storage location. These features ease the user to have an overview of the relationship of each objects and the storage box. It can potentially reduce the time to search for the storage box of each objects.

1.4.4 E-commerce platform

An e-commerce platform will be developed and provide a suitable platform for makers to sell or rent their resources, such as equipment and ingredients among each other. As a seller, they will have a store management page which can add new products to shelf and view the current on-shelf products. Next, as a buyer, they can add the desired products into cart and make payment after that.

1.5 Impact, Significance and Contribution

The StoreHut maker space storage management mobile application could ease the tracking of object location in a maker space by a simple and effective way. It is reducing the workload of the makers in finding the objects and understanding the type and quantity of equipment. It also promotes the usage of chaotic storage system in makerspace as this storage method could save more time, flexible and space-efficient. Besides, it provides a platform for maker to share and sell their tools and equipment. Furthermore, the emergence of makerspaces, of which there are now many all over the world, has increased the popularity of maker culture. We had observed the population of hackerspace in hackerspaces.org, which serves as a collaboration and communication platform among hackerspace. At the time of this writing, there are 2419 hackerspaces listed in hackerspace.org while 925 of them are still active and 360 of them are marked as planned [6]. Those hackerspaces are registered in different location all over the world. Therefore, the development of this project will definitely benefit a great range of people.

CHAPTER 2 LITERATURE REVIEW

Chaotic storage management systems are not a new concept, particularly in the warehouse and manufacturing industries. However, in terms of the size of the area, resources available, and user requirements, the chaotic storage management system in the makerspace is not the same as what was done in the warehouse. Although they differ in certain ways, they share some of the same implementation concepts. As a result, the following studies provide methods for developing and improving the chaotic storage management system in warehouses and makerspaces. Furthermore, the functionality and effectiveness of a list of existing systems are being examined in order to gain a better understanding of previous work in this field.

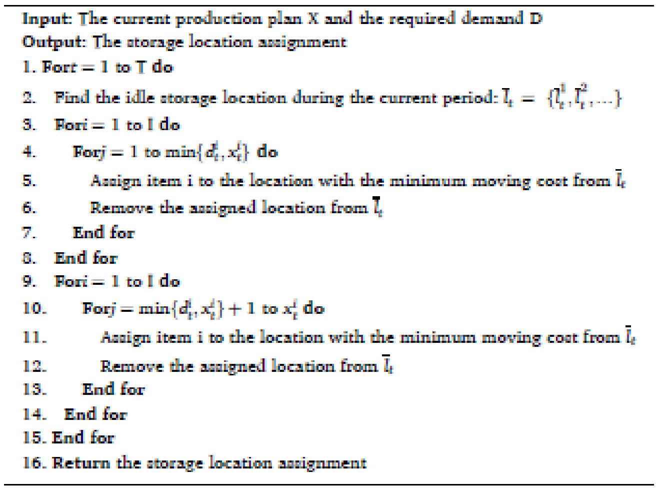
2.1 Literature Review

2.1.1 Warehouse management model using FEFO, 5s, and chaotic storage to improve product loading times in small- and medium-sized non-metallic mining companies

The aims of study for this paper is to propose an economically viable warehouse management system that helps the SME to overcome the problems of delayed shipping and to prevent products from being expired when they are sent. According to the study of [9], this chaotic storage management is integrated with the application of First Expired, First Out (FEFO) logic in handling the products with limited shelf life and manage to reduce expired products by 50%. If the product is non-perishable, First In, First Out (FIFO) logic can be implemented because it is more time-consuming. By implementing chaotic storage management, the stock quantity can be tracked at all time without moving the products. Besides, the stock map is used to record the warehouse movements so that the operators can manage the orders and evaluate the capacity of warehouse in real time basis. In this case, the products could be relocated daily based on the expiring period using the traffic light rules, illustrated in Figure 2.1.1.1 [9]. However, the research does not discuss the tools and method used to track the location of each product in the warehouse.

2.2.2 Integrated production planning and warehouse storage assignment problem: An IoT assisted case

The purpose of this study was to solve the problem of shortage of warehouse space for finished products that cause limit production in a food company. According to [5], a novel integrated strategy that combine production planning with a randomized storage assignment policy is produced. They had implemented IoT-enabled tracking systems, which might aid in improving inventory visibility and traceability. Furthermore, the greedy method is used to assign each product to the best possible position with the least amount of movement. Before the item is located, the available place must first be found. Following that, if the item is in high demand for this time period, it will be transferred to the closest accessible location at the lowest possible cost. While the other item will be moved to other available location in sequence. Finally, the currently occupied place will be designated as unavailable for the time being. The detailed of pseudocode for location assignment implemented in this system is presented on Figure 2.2.2.1. Moreover, [5] had conclude that storage usage of randomized policies is much lower than dedicated policy. The maximum space saving for randomized policy is 26.1% while the average saving is 14.8%.



```
Input: The current production plan X and the required demand D
Output: The storage location assignment
1. For t = 1 to T do
2.   Find the idle storage location during the current period: $\bar{l}_t = \{\bar{l}_t^1, \bar{l}_t^2, \dots\}$
3.   For i = 1 to I do
4.     For j = 1 to $\min\{d_t^i, x_t^i\}$ do
5.       Assign item i to the location with the minimum moving cost from $\bar{l}_t$
6.       Remove the assigned location from $\bar{l}_t$
7.     End for
8.   End for
9.   For i = 1 to I do
10.    For j = $\min\{d_t^i, x_t^i\} + 1$ to $x_t^i$ do
11.      Assign item i to the location with the minimum moving cost from $\bar{l}_t$
12.      Remove the assigned location from $\bar{l}_t$
13.    End for
14.  End for
15. End for
16. Return the storage location assignment
```

Figure 2.2.2.1 The detail of Pseudocode for location assignment.

2.2.3 Design of a reference architecture for developing smart warehouses in industry 4.0

This paper is studied with the objective to propose a smart warehousing system in Industry 4.0. Based on the research done by [7], the incoming items must be uniquely identified in order for the system to properly handle them. The authors propose two methods for identifying the goods: barcode and RFID. They all have distinct characteristics. If only a little amount of data needs to be embedded in the tags and the storage range is limited, a barcode is a good choice. It can be applied in a variety of ways, including stickers, printing, and laser printing. Next, RFID can store a vast amount of data, be read from a long distance, and provide real-time data[7]. It gives a more precise, straightforward, and efficient service. Other than that, scanners can also be used to read the barcode and RFID tags on the items, and the data is then communicated to the warehouse management system for tracking and tracing. To secure the information and limit accessibility, the scanner should be safeguarded with user identification. Besides, [7] mentioned that short distance-reading scanner can avoid errors and speed up the process of scanning. Figure 2.2.3.1 is showing the feature diagram of a scanner.

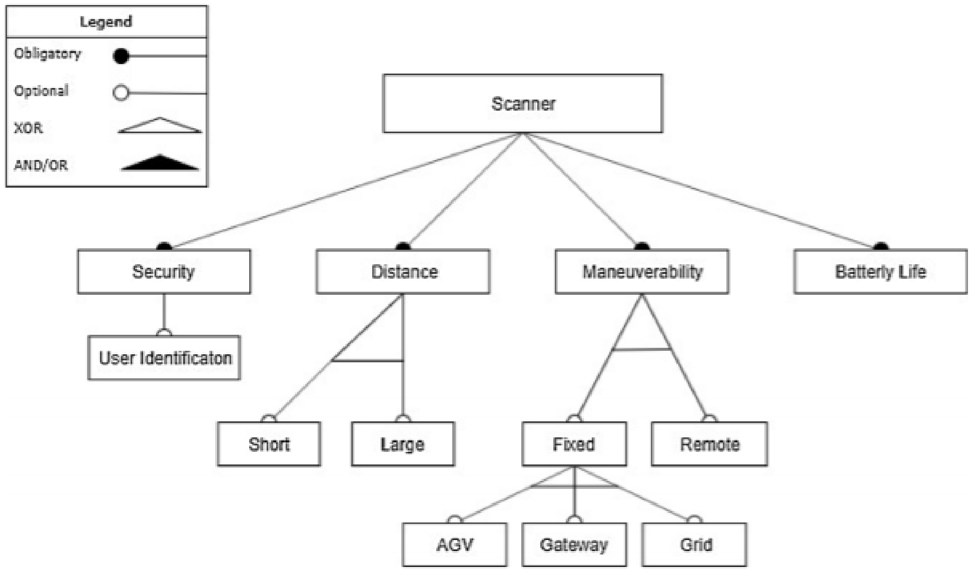


Figure 2.2.3.1 Feature Diagram of Scanner

Moreover, the Warehouse management system (WMS) is implemented in the Smart Warehouses to runs the warehouse while supported by other systems such as

Transport Management System (TMS) and Inventory Management System (IMS). Furthermore, the recording and decision support features is collecting all the data in WMS. Next, the feature diagram of Warehouse Management systems is presented in Figure 2.2.3.2. Last but not least, all the component such as scanner, sensors, and machinery should be connected together under the warehouse communication network.

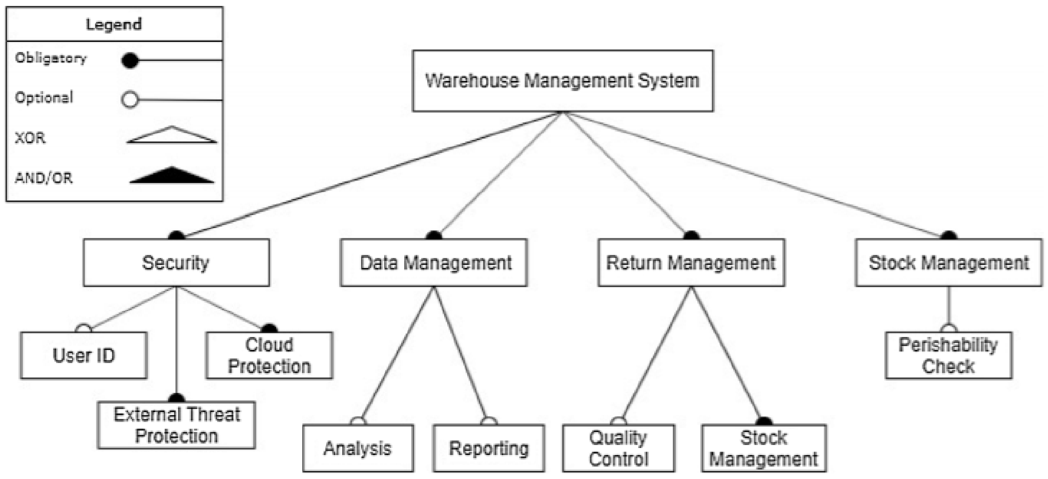


Figure 2.2.3.2 Feature Diagram of Warehouse Management System

2.2.4 The Construction of a Consumer To Consumer E-commerce System Based on Japanese Elderly

The aim of this paper was to develop an elderly community website which act as an online e-commerce system. According to the session on Logistic and Funds Flow of Commodity Trading proposed by [2], private transactions are implemented in this application, where buyers can communicate to establish the price and trade.

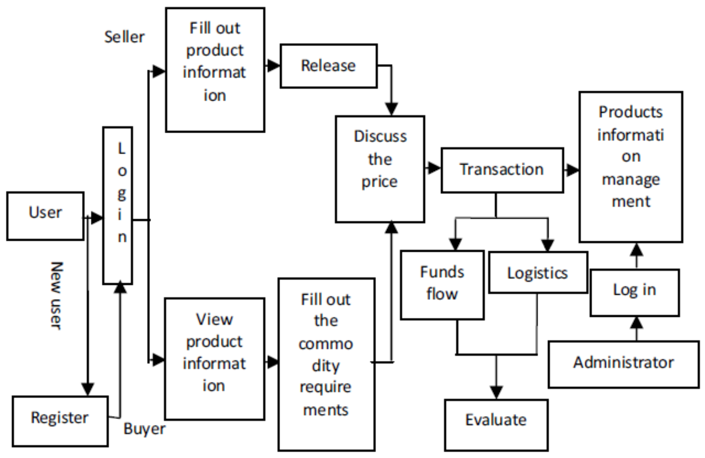


Figure 2.2.4.1 Business Process Online

Next, the main user of the information system is user, goods providers, and administrator. ASP.NET is used for development because it can combine with Microsoft's operating system and backstage database closely. Furthermore, [2] presented a user management subsystem in which users can input personal information, search for commodities using keywords, and obtain full information about each item. Following that, the application can track successful transactions and inventory. In addition, purchase orders and receipt confirmations are kept in the transaction management subsystem, while good providers can maintain their goods information in the commodity management subsystem, such as add, change, and remove goods. However, the administration staff is needed to deal with the users, such as dealing with assistance applications and processing complaints about the seller's items that are unsatisfactory.

2.2.5 Innovation Management And Automated Accounting In The Chaotic Storage Logistics

The goal of this study is to develop an organisational structure for accounting warehouse products that integrates logistical chaotic storage and RFID to automate the warehouse management process. [12] has defined chaotic storage system as a system that randomly places objects on shelves without regard for their characteristics, shelf life, or other attributes. This form of logistical storage system might effectively maximise storage space utilisation while reducing the physical size of the area required. Furthermore, it eliminates the burden of remembering the exact placement of each and every product in the storage site.

The efficacy of controlling the storage site can be greatly enhanced by increasing the accuracy and timeliness of accounting information by implementing a chaotic storage system. According to [12], because only the item's name and storage location will be provided in the later inventory related document, accounting should include multidimensional primary information such as the item's description, exact location, and person in charge. The proposed approach suggests using RFID as a marking tag on things that is embedded with accounting data that records item details such as quantity and weight. After that, those RFID tags are tied to specific location using RFID technology. The RFID tags are then linked to a specific location via RFID technology. The warehouse management system can automatically record the shifting or disposal of objects throughout the inventory process. RFID technology devices can be utilised to gather and process accounting data in this scenario. [12] This methodology, however, has some functional limitations. Firstly, frequently used things may be stored in a position far from the collecting and transportation point, resulting in inconvenient or time-consuming item movement. Second, an initially bunched item, such as raw materials for some goods, may be separated due to random placement.

2.2 Review and Comparison of Previous Work

2.2.1 NFC Part Management System

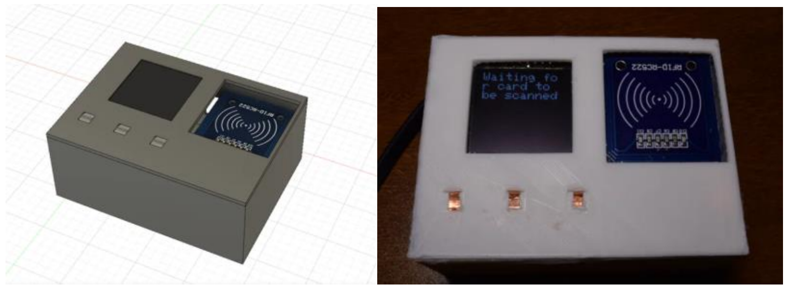


Figure 2.2.1.1 Image of NFC Part Management System

Maker Space NFC Part Management System is design and developed to organize the tools, equipment and ingredient used in makerspace [8]. The tools involved are NFC stickers, Particle Photon with NFC Reader and OLED screen and a web interface. Firstly, the web interface connected to MySQL database could be used to manage the storage bin and objects.

Secondly, according to [8], the NFC sticker is created and attached on each storage bin and serves as label. Every objects stored in the storage bin must be registered and assigned with a storage bin Id via web interface. Lastly, the NFC Reader on the Particle Photon can be used to scan the NFC sticker of the storage bin. A list of items will be displayed on the OLED screen while user can edit the quantity of each object.

The Maker Space NFC Part Management System could effectively manage and track the objects in a simple and clean way. Also, it is flexible in allocating the objects because it can always be managed using the web interface. However, NFC is not suitable for storage management system because the short distance of scanning makes the work inconvenient and it is not necessary to ensure an extremely secure process in storage management system. In other word, it is greater to be used for secure transaction, where the reader must be very near to the tag.

2.2.2 Sortly

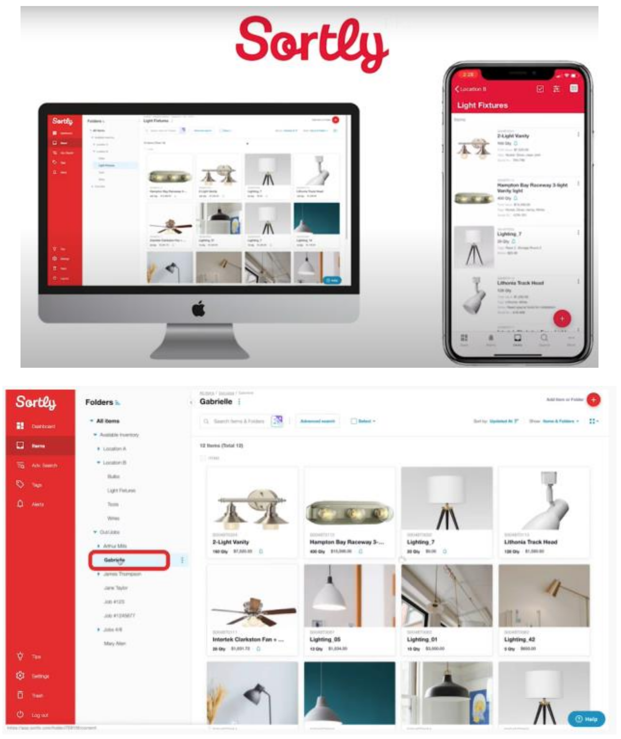


Figure 2.2.2.1 User Interface of Sortly application

Sortly makes it easy for businesses to keep track of their goods. It keeps the user informed about what he has and where it is [11]. Sortly is a cross-platform application. It works with both mobile devices and desktop browsers at the same time, allowing users to keep track of their belongings at all times and from anywhere. It also has a barcode scanner that connects each of the objects stored in Sortly to a QR Code and a BarcodeSortly can link to the item's barcode if it has one. Sortly also has a quick move option that allows users to scan a barcode and move an item to another folder[11]. Not only that, yet this alert feature might notify the user when the quantity of goods is running low. The reminder can be directed at a specific client or team member. Additionally, users can easily add another team member or customer to have access to the inventory, as well as configure permissions to just see a specific item. Sortly, on the other hand, is not a free software, despite the fact that it offers a 14-day trial to new users. The plans start at $25 per month[11].

2.2.3 Cashier Live

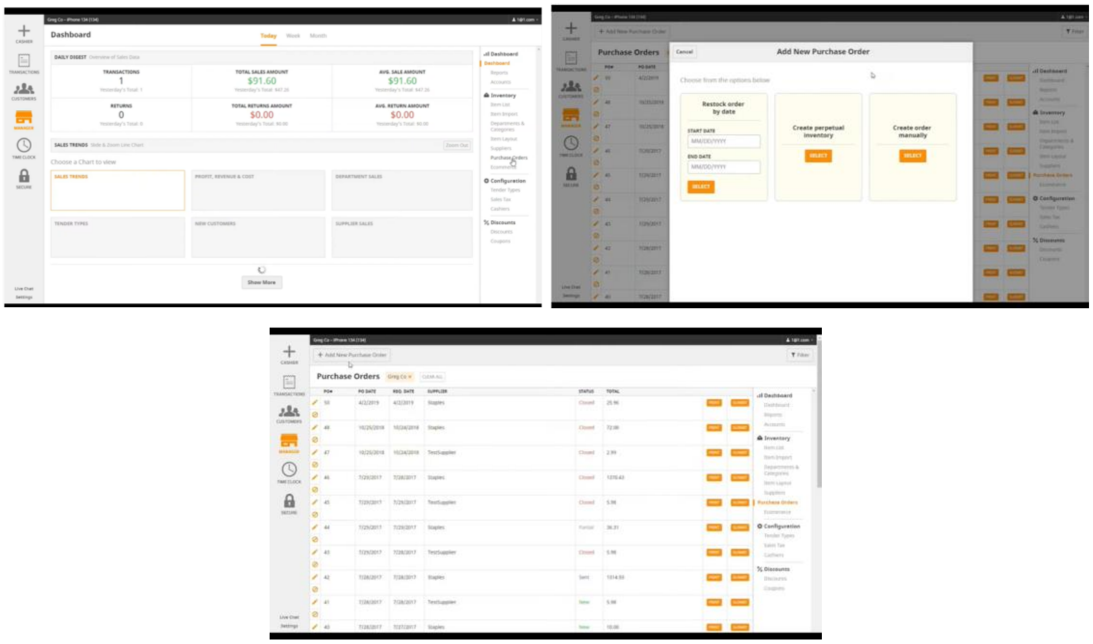


Figure 2.2.3.1 User interface of Cashier Live application

Cashier Live is an inventory management app for iOS and Android [4]. This application could assist shops or sellers in tracking and managing their inventory. Furthermore, Cashier Live is equipped with a barcode scanner that scans the barcodes of the items using the smartphone's camera or Linea-Pro device. The application then allows the user to create a barcode that may be used to uniquely identify inventory items. Aside from that, the inventory file can be exported to an Excel file and emailed. This feature makes it easier to report and summarise inventories over a period of time. However, new users are only given a 14-day trial, after which they must pay to continue using the programme. Besides, the application has no stock replenishment reminder and searching features.

2.2.4 Carousell

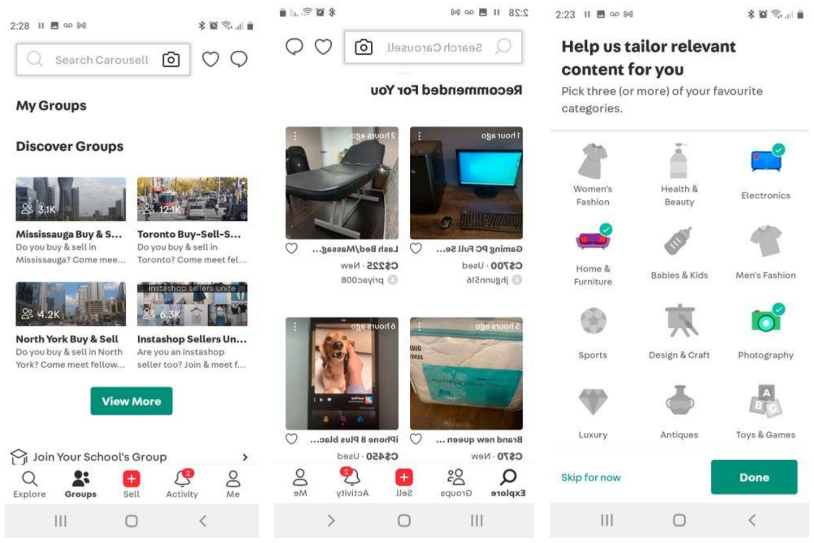
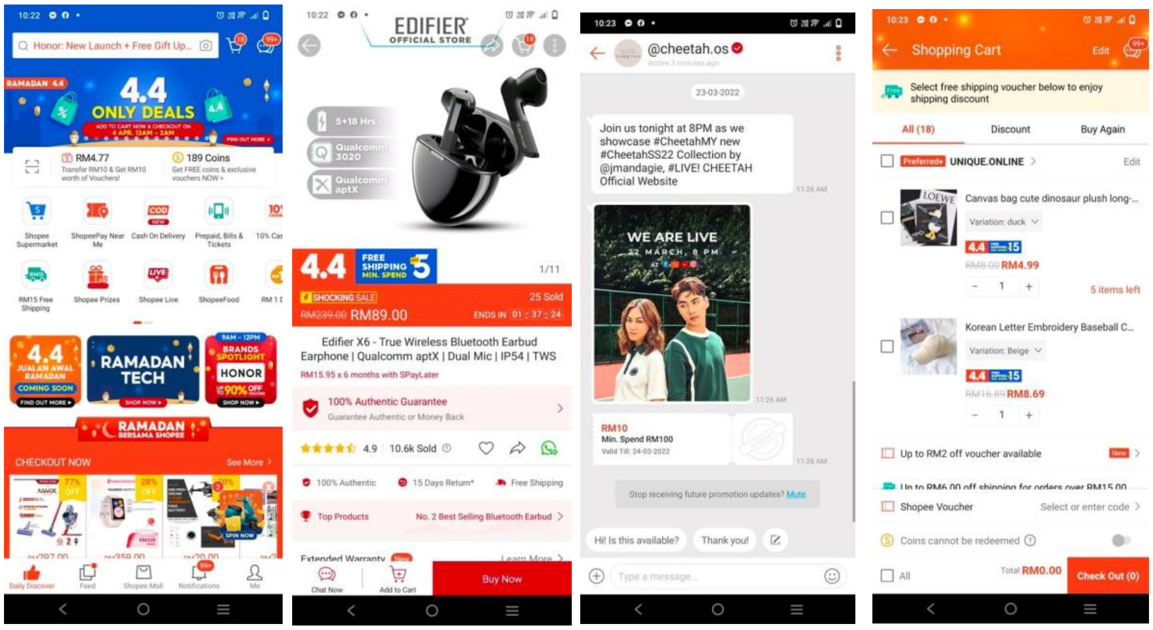


Figure 2.2.4.1 User interface of Carousell

Carousell is a mobile shopping app that offers a wide range of products from local vendors [3]. The sellers are free to sell any thing they want, whether new or secondhand. Furthermore, the user can search for products based on their categories or keywords. Before buying something, users may look at the seller's page and see their star rating as well as buyer reviews. Furthermore, Carousell functions as a social networking platform where users may join various seller groups and discover various things there. Despite the fact that all vendors are in the same country, the item purchased takes a few days to arrive. If they are close by, the delivery time may be shortened. However, it is difficult for a user to locate sellers who are based in the same physical place as him.

2.2.5 Shopee



Join us tonight at 8PM as we showcase #CheetahMY new #CheetahSS22 Collection by @jmandagie, #LIVE! CHEETAH Official Website

Figure 2.2.5.1 User interface of Shopee

Shopee is a cross-platform application that allows Malaysians to sell, buy, and search for items they want [10]. Before shopping for products, Shopee provides a geographical filtering function that allows customers to choose a seller's physical location. Shopee provides payment and shipping services, as well as product advertising. Additionally, the customer can communicate with the vendor via Shopee's webchat feature. Customer can also report their problem by engaging Shopee Live Chat Team anytime[10]. This enables Shopee to respond to client questions at any moment. Buyers, on the other hand, can see other buyers' evaluations and feedback about their purchasing experience and the items they purchased[10].

2.3 Critical Remarks of Previous Works

	NFC Part Management System	Sortly	Cashier Live
Overview of products and storage location	√	√	√
Search Function	X	√	X
Image capturing	X	√	√
Stock Replenishment Notification	X	√	X
Free Application	√	X	X
Cost of Plan	-	$25 per month	$75 per month
Technology used	NFC	Barcode	Barcode
Cross-Platform	X	√	√
Supported OS	-	Android, iOS	Android, iOS

Table 2.3.1 Summaries of Application Feature of Storage Management System

	Carousell	Shopee
Searching feature	√	√
Payment service	√	√
Rating and left a comment	√	√
Delivery service	√	√
Geographical filtering	X	√
Add into cart/favourite	X	√

Table 2.3.2 Summaries of Application Feature of Local E-Commmerce Application

CHAPTER 3 SYSTEM METHODOLOGY AND SYSTEM DESIGN

3.1 System methodology

Agile development adopted with Software Development Life Cycle (SDLC) is applied to the system, as shown in Figure 4.1.1. This methodology is being chose due to its flexibility. The phrases of design, development and validation are inter-leaved in agile development. When the system requirement is subjected to change, it can be solved quickly without unnecessary delays. Thus, it reduces the delivery time for the proposed system and it is more suitable for the project with limited time constraint.

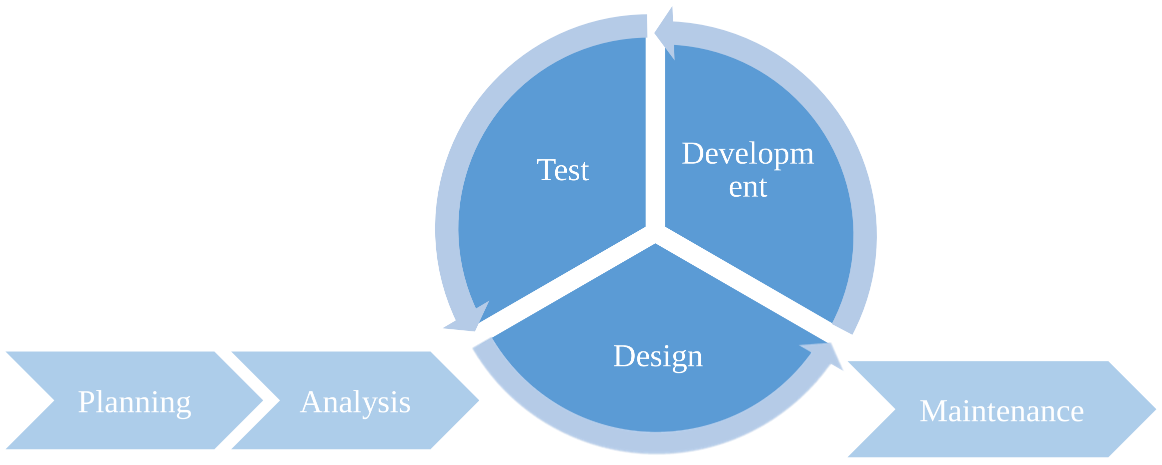


Figure 3.1.1 SDLC – Model of Agile development

<u>Planning Phase</u>

In the planning stage, the business value of the system is being analyzed and the scope of project is finalized. We had understood the basic knowledge on chaotic storage system and analyze the market demands.

<u>Analysis phase</u>

In analysis phase, we analyzed the existing system used to manage the storage in makerspace and point out their strength and limitation. Besides, we also study the research papers that highlights the current practices towards the chaotic storage. According to the literature review, we investigate and define new requirement for developing the new system.

Design phase

In design phase, system design diagram such as use case diagram, activity diagram and class diagram is being created. Besides, the system UI is designed and high fidelity prototype is presented.

Implementation phase

In implementation phase, coding and system testing are done simultaneously so that the error can be solved as quick as possible if existed. The system requirement may be changed if the system does not work well.

3.2 System Design Diagram

3.2.1 System Architecture Diagram

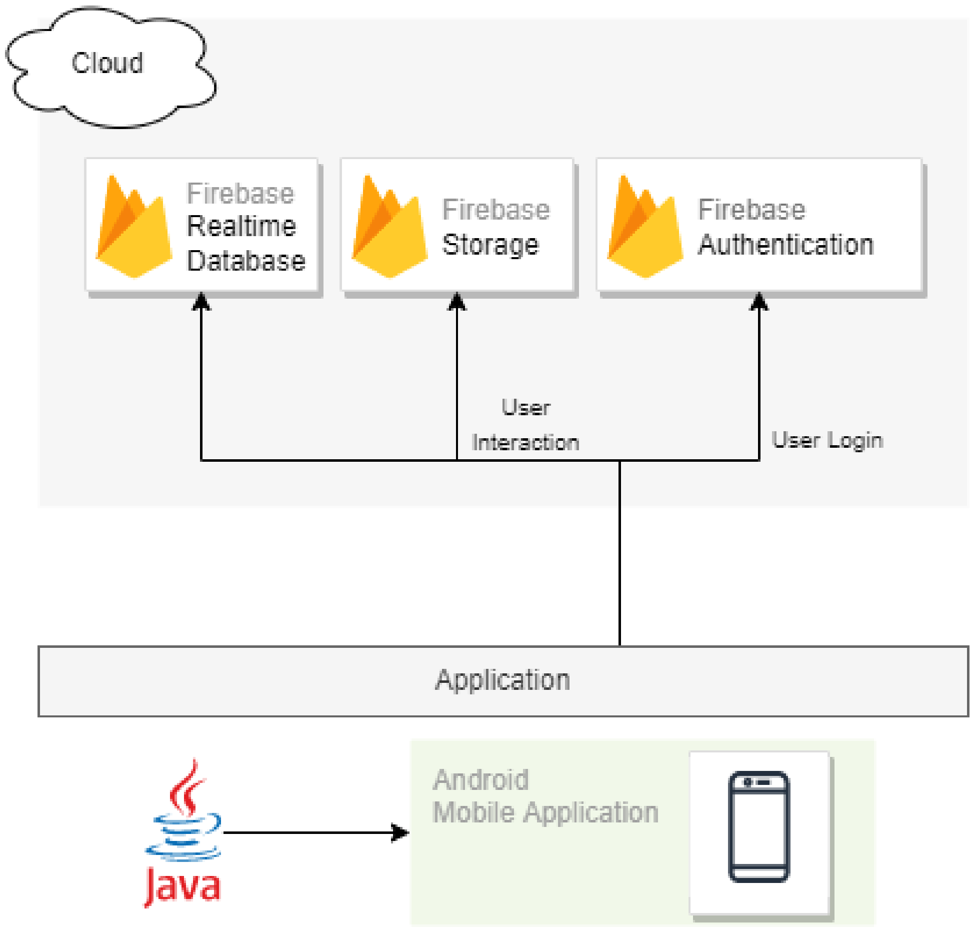


Figure 3.2.1.1 Architecture Diagram

Android Studio has been chosen as the tools to develop our Android Mobile Application. In our case, Java language has been used for app development. Next, Firebase Authentication Besides, Firebase is used in the project to provide backend services such as authentication of account, real-time database, file storage and so on. In this case, email and password based authentication is used to authenticate users. In addition, all data of the system are stored in Firebase realtime database as JSON and synchronized in realtime to every connected client while the images are stored in Firebase Storage.

3.2.2 Use Case Diagram

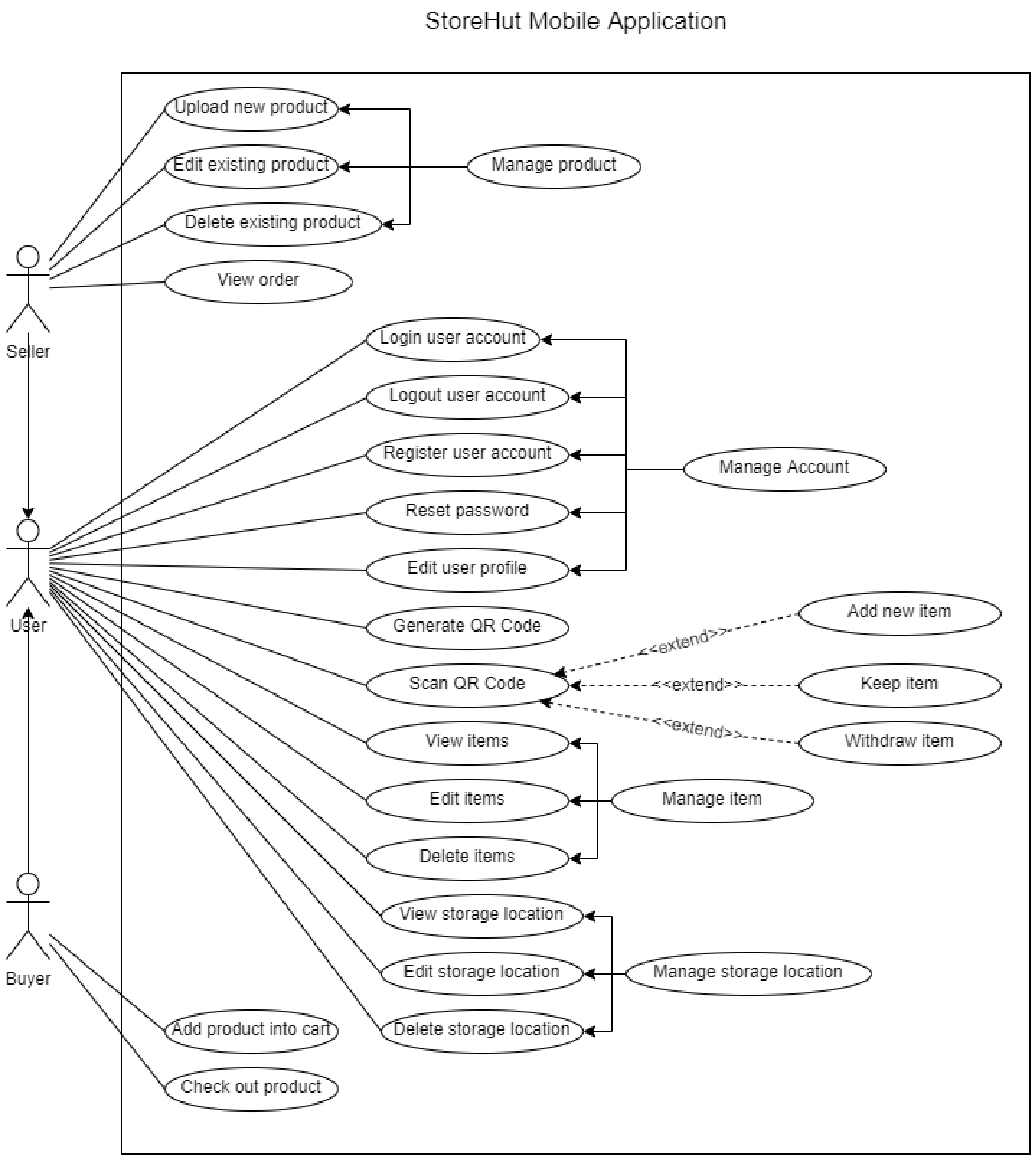


Figure 3.2.2.1 Use Case Diagram

3.1.3 Use Case Description

Use Case ID	UC001		**Version**	1.0
Use Case	Generate QR Code			
Purpose	To generate QR Code for storage location registration.			
Actor	User			
Trigger	Click QR Code icon in homepage, and click "Generate QR Code".			
Precondition	User is logged in.			
Scenario Name	**Step**	**Action**		
Main Flow	1	System requests user to input the name of storage bin, location, and extra description which is optional to be filled in.		
	2	User enters name of storage bin, location, and extra description(optional).		
	3	User click on "Generate" button.		
	4	System generate a QR Code based on the information entered, and display it on the screen while automatically saved into device's local storage.		
Alternate Flow – Empty storage bin or location	2.1	User does not enter name of storage bin or location.		
	2.3	System displays error message "Storage bin must be defined" or "Location must be defined".		
	2.4	Back to Main Flow Step 1.		

Table 3.2.3.1 QR Code's Use Case Description

Use Case ID	UC002		**Version**	1.0
Use Case	Add new item			
Purpose	To register new item into the system.			
Actor	User			
Trigger	Click QR Code icon in homepage, then click "Scan QR Code" and scan the QR Code attached to storage location. After that, select "Add New Item"			
Precondition	User is logged in.			
Scenario Name	**Step**	**Action**		
Main Flow	1	System requests user to input the item name, description and quantity as well as insert a relevant image.		
	2	User enters item name, description(optional), quantity, and insert an image.		

	3	User clicks on "Add Item" button.
	4	System saves the record into database.
Alternate Flow – Empty item name or quantity	2.1	User does not enter item name or quantity.
	2.3	System displays error message "Item name must be defined" or "Quantity must be defined".
	2.4	Back to Main Flow Step 1.

Table 3.2.3.2 Add new item's Use Case Description

Use Case ID	UC003	**Version**	1.0
Use Case	Keep Item		
Purpose	To keep item into the storage location or increase the stock.		
Actor	User		
Trigger	Click QR Code icon in homepage, then click "Scan QR Code" button and scan the QR Code attached to storage location. After that, select "Keep Item"		
Precondition	User is logged in.		
Scenario Name	**Step**	**Action**	
Main Flow	1	System display a list of item that are registered or stored in this storage location. System request user to select an item.	
	2	User selects an item from the list.	
	3	System request user to enter the quantity of item to be kept.	
	4	User enter quantity of item to be kept.	
	5	System save the record into database and update the quantity of item.	
Alternate Flow – Empty quantity	2.1	User does not enter quantity of item to be kept.	
	2.3	System displays error message "Quantity must be defined".	
	2.4	Back to Main Flow Step 1.	

Table 3.2.3.3 Keep item's Use Case Description

Use Case ID	UC004	**Version**	1.0
Use Case	Withdraw Item		
Purpose	To withdraw item from the storage location or decrease the stock.		
Actor	User		

Trigger	Click QR Code icon in homepage, then click "Scan QR Code" button and scan the QR Code attached to storage location. After that, select "Withdraw Item"	
Precondition	User is logged in.	
Scenario Name	**Step**	**Action**
Main Flow	1	System display a list of item that are registered or stored in this storage location. System request user to select an item.
	2	User selects an item from the list.
	3	System request user to enter the quantity of item to be withdraw.
	4	User enter quantity of item to be withdraw.
	5	System save the record into database and update the quantity of item.
Alternate Flow – Empty quantity	2.1	User does not enter quantity of item to be kept.
	2.2	System displays error message "Quantity must be defined".
	2.3	Back to Main Flow Step 1.
Alternate Flow – Quantity more than the stock amount	2.1	User enter quantity of item which is more than the stock available
	2.2	System displays error message "Stock not enough".
	2.3	Back to Main Flow Step 1.

Table 3.2.3.4 Withdraw item's Use Case Description

Use Case ID	UC005	**Version**	1.0
Use Case	View item list		
Purpose	To view the list of item.		
Actor	User		
Trigger	Click "My List" icon on homepage. Toggle the tab above to "Item".		
Precondition	User is logged in.		
Scenario Name	**Step**	**Action**	
Main Flow	1	System displays a list of item that are registered.	
	2	User selects an item from the list.	
	3	System displays all of the information regarding this item such as image, item name, item description, current quantity, and storage location.	
Sub Flow – See location of item	4.1	User clicks on the green color link "Click here" next to the text "Want to see location?".	
	4.2	System navigate to storage location page.	

Sub Flow – Edit item	5.1	User clicks on “Edit item” button.
	5.2	System displays the current record of the item.
	5.3	User updates the information of the item.
	5.4	User clicks on “Confirm” button
Sub Flow – Delete item	6.1	User clicks on “Delete item” button.
	6.2	System delete the item.
Alternate Flow – Empty item name or quantity	5.3.1	User does not enter item name or quantity.
	5.3.2	System displays error message “Item name must be defined” or “Quantity must be defined”.
	5.3.3	Back to Main Flow Step 1.

Table 3.2.3.5 View item list’s Use Case Description

Use Case ID	UC006	**Version**	1.0
Use Case	View storage location list		
Purpose	To view the list of storage location.		
Actor	User		
Trigger	Click “My List” icon on homepage. Toggle the tab above to “Storage Location”.		
Precondition	User is logged in.		
Scenario Name	**Step**	**Action**	
Main Flow	1	System displays a list of storage location that are registered.	
	2	User selects a storage location from the list.	
	3	System displays all of the information regarding this storage location such as image of QR Code, storage bin name, storage description, storage location, total amount of item and list of item which are stored inside the storage location.	
Sub Flow – See item stored inside it	4.1	User clicks on item which is listed out at the lower part of the page.	
	4.2	System navigate to item page.	
Sub Flow – Edit storage location	5.1	User clicks on “Edit storage” button.	
	5.2	System displays the current record of the storage location.	
	5.3	User updates the information of the storage location.	
	5.4	User clicks on “Confirm” button	
Sub Flow – Delete item	6.1	User clicks on “Delete storage” button.	
	6.2	System delete the storage location as well as the item stored inside it.	

Alternate Flow – Empty storage bin or storage location	5.3.1	User does not enter name of storage bin and storage description.
	5.3.2	System displays error message "Storage bin must be defined" or "Storage location must be defined".
	5.3.3	Back to Main Flow Step 1.

Table 3.2.3.6 View storage location list's Use Case Description

Use Case ID	UC007	**Version**	1.0
Use Case	Add product into cart		
Purpose	To add the product in StoreHut Mart into cart.		
Actor	User - Buyer		
Trigger	Click "StoreHut Mart" icon on homepage and select a product.		
Precondition	User is logged in.		
Scenario Name	**Step**	**Action**	
Main Flow	1	System displays the information of the product.	
	2	User click on "Add To Cart" Button.	
	3	System save product into cart.	

Table 3.2.3.7 Add product into cart's Use Case Description

Use Case ID	UC008	**Version**	1.0
Use Case	Contact seller of product		
Purpose	To contact the seller who sell or rent the product.		
Actor	User - Buyer		
Trigger	User click on "StoreHut Mart" icon on homepage and select a product.		
Precondition	User is logged in.		
Scenario Name	**Step**	**Action**	
Main Flow	1	System displays the information of the product.	
	2	User click on "Contact Seller" Button.	
	3	System navigate to device's phone call screen.	

Table 3.2.3.8 Contact seller of product's Use Case Description

Use Case ID	UC009	**Version**	1.0
Use Case	View on-shelf products		

Purpose	To view the on-shelf products.	
Actor	User - Seller	
Trigger	Click “My Store” icon at homepage and select “My Product” button.	
Precondition	User is logged in.	
Scenario Name	**Step**	**Action**
Main Flow	1	System displays the list of products.
	2	User clicks on the product.
	3	System displays the information of the product.
Sub Flow – Edit Product	4.1	User clicks on “Edit” button.
	4.2	System displays the current record of the products.
	4.3	User updates the information of the products.
	4.4	User clicks on “Save” button
Sub Flow – Delete item	5.1	User clicks on “Delete” button.
	5.2	System delete the product.
Alternate Flow – Empty product name, product details, stock quantity, selling price, daily renting price, product image	4.3.1	User does not enter product name, product details, stock quantity, selling price or daily renting price
	4.3.2	System displays error message “Product name must be defined” or “Product details must be defined” or “Stock quantity must be defined” or “Selling price must be defined” or “Daily renting price must be defined” or “Please take a photo of your products”.
	4.3.3	Back to Main Flow Step 1.

Table 3.2.3.9 View on-shelf products’ Use Case Description

Use Case ID	UC010	**Version**	1.0
Use Case	Add new product to shelf		
Purpose	To register new product to be sell or rent.		
Actor	User - Seller		
Trigger	Click “My Store” icon at homepage and select “Add Product” button.		
Precondition	User is logged in.		
Scenario Name	**Step**	**Action**	
Main Flow	1	System requests user to input the product name, product details, stock quantity, selling price, daily renting price, selling mode, and add a relevant product’s image.	

	2	User enters product name, product details, stock quantity, selling price, daily renting price, selects a selling mode and inserts a relevant image.
	3	User clicks on “Save” button.
	4	System saves the record into database.
Alternate Flow – Empty product name, product details, stock quantity, selling price, daily renting price, product image	2.1	User does not enter product name, product details, stock quantity, selling price or daily renting price
	2.2	System displays error message “Product name must be defined” or “Product details must be defined” or “Stock quantity must be defined” or “Selling price must be defined” or “Daily renting price must be defined” or “Please take a photo of your products”.
	2.3	Back to Main Flow Step 1.

Table 3.2.3.10 Add new product to shelf’s Use Case Description

3.2.4 Sequence Diagram

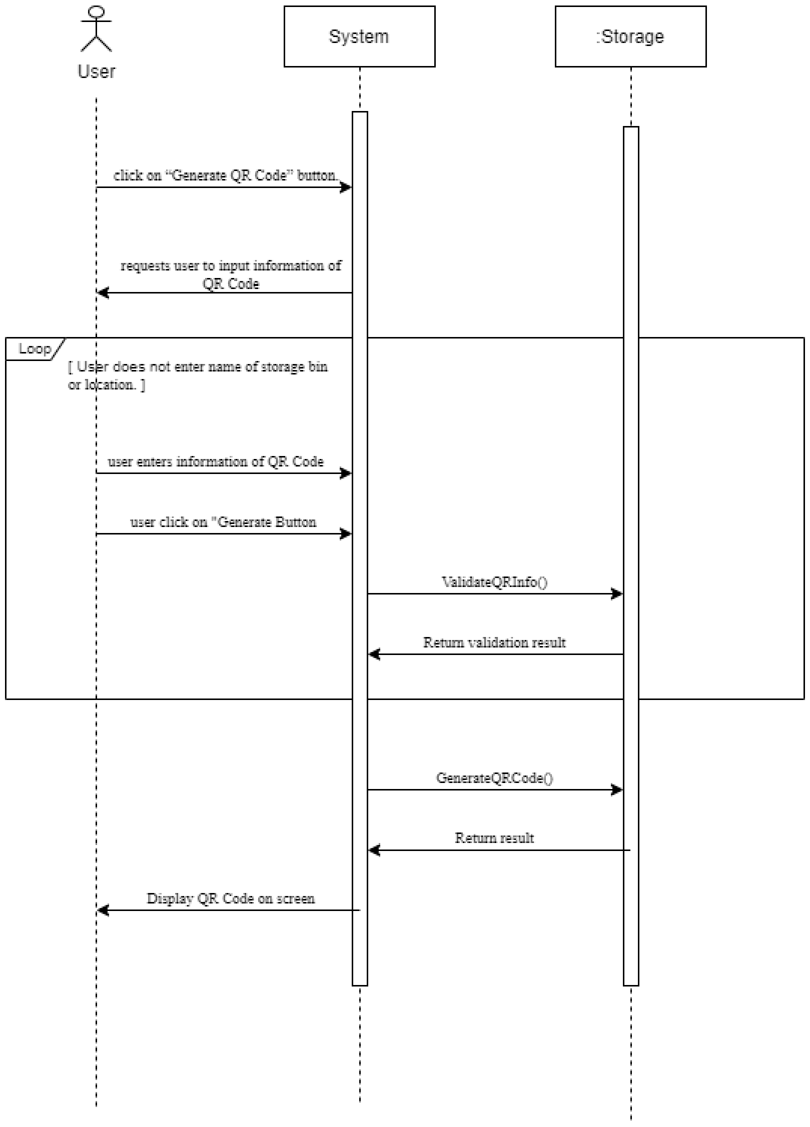

User
System
:Storage
click on "Generate QR Code" button.
requests user to input information of QR Code
Loop
[User does not enter name of storage bin or location.]
user enters information of QR Code
user click on "Generate Button
ValidateQRInfo()
Return validation result
GenerateQRCode()
Return result
Display QR Code on screen

Figure 3.2.4.1 Generate QR Code's Sequence Diagram

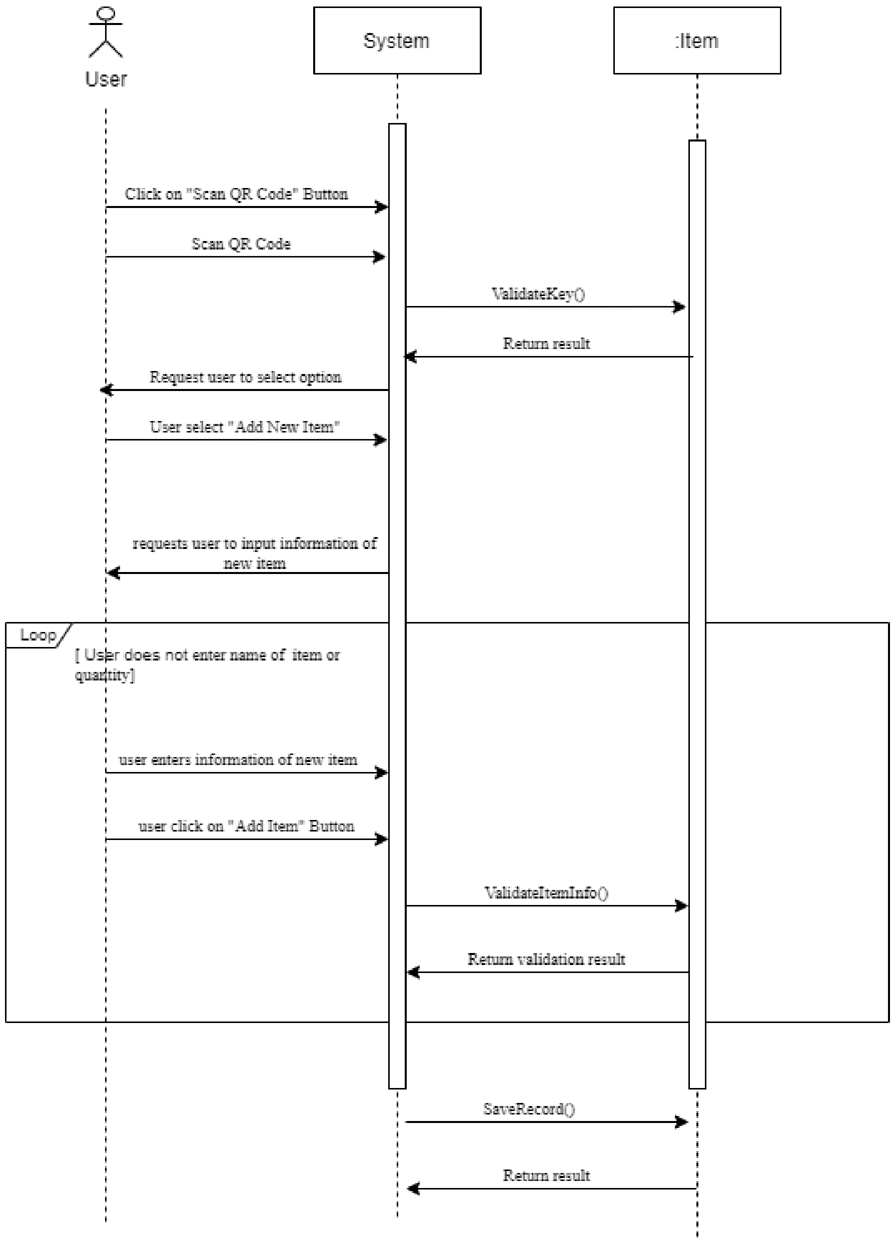

User
System
:Item
Click on "Scan QR Code" Button
Scan QR Code
ValidateKey()
Return result
Request user to select option
User select "Add New Item"
requests user to input information of new item
Loop
[User does not enter name of item or quantity]
user enters information of new item
user click on "Add Item" Button
ValidateItemInfo()
Return validation result
SaveRecord()
Return result

Figure 3.2.4.2 Scan QR Code' Sequence Diagram

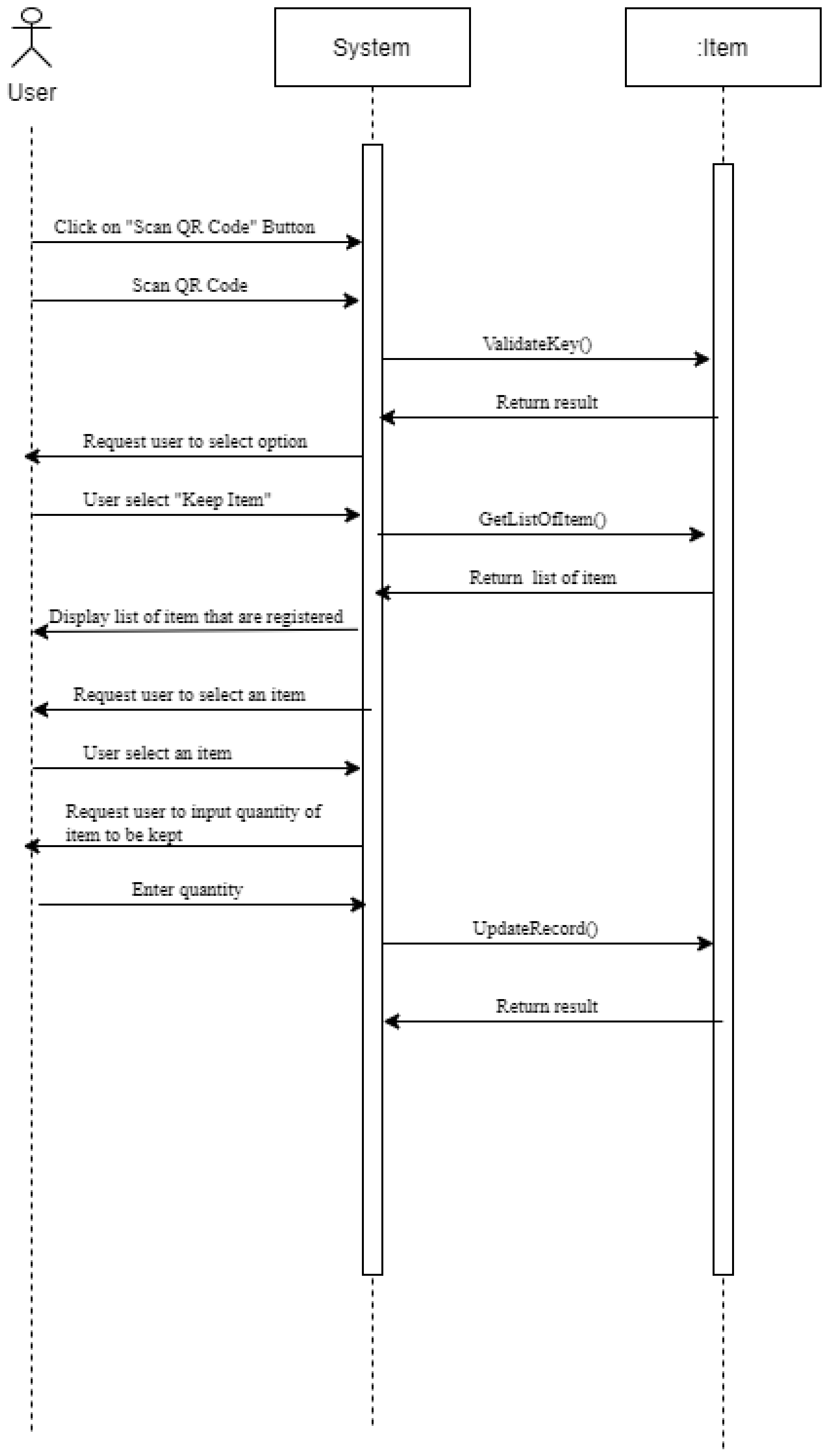

User
System
:Item
Click on "Scan QR Code" Button
Scan QR Code
ValidateKey()
Return result
Request user to select option
User select "Keep Item"
GetListOfItem()
Return list of item
Display list of item that are registered
Request user to select an item
User select an item
Request user to input quantity of item to be kept
Enter quantity
UpdateRecord()
Return result

Figure 3.2.4.3 Keep Item Sequence Diagram

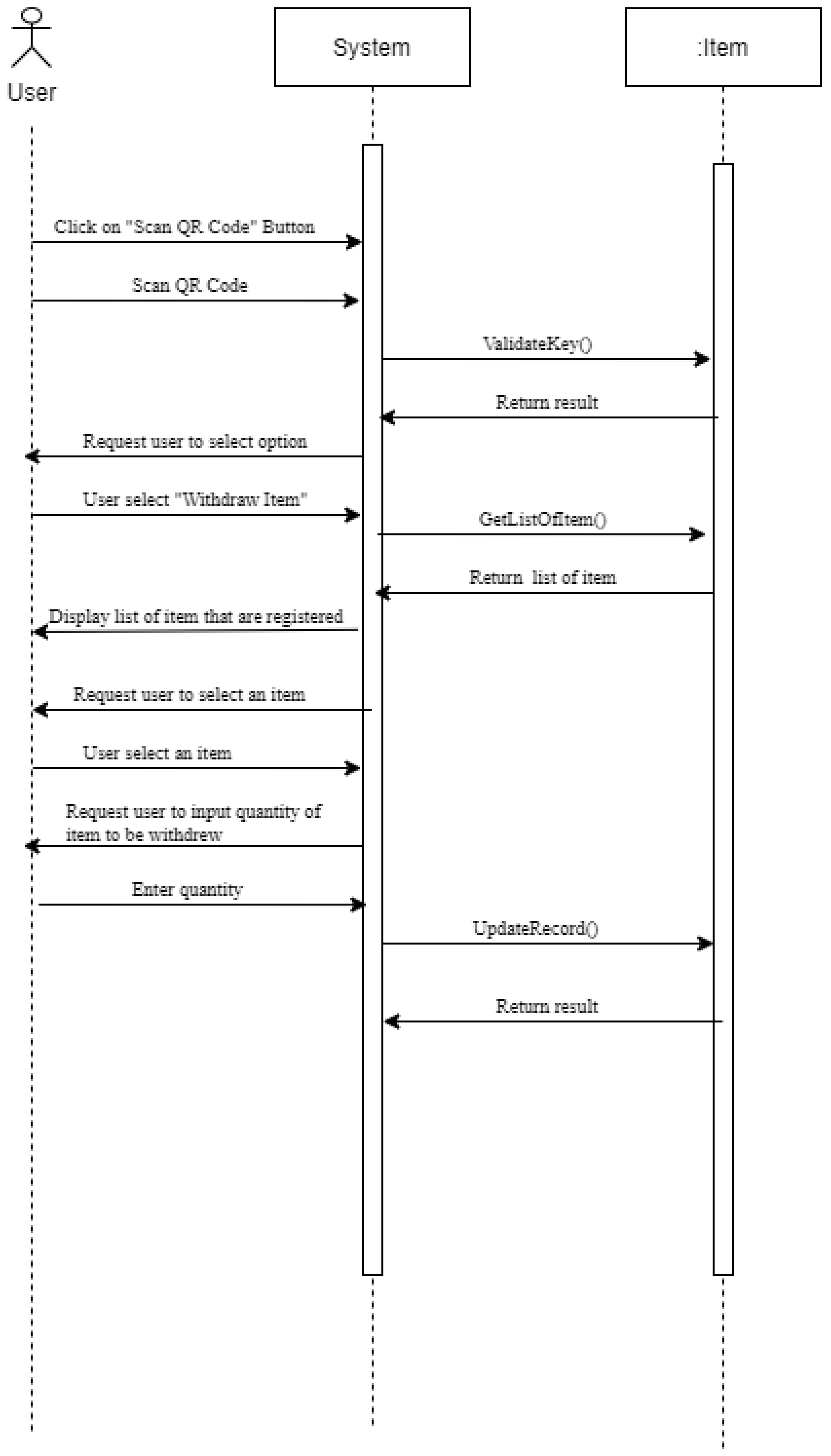

User
System
:Item
Click on "Scan QR Code" Button
Scan QR Code
ValidateKey()
Return result
Request user to select option
User select "Withdraw Item"
GetListOfItem()
Return list of item
Display list of item that are registered
Request user to select an item
User select an item
Request user to input quantity of item to be withdrew
Enter quantity
UpdateRecord()
Return result

Figure 3.2.4.4 Withdraw Item Sequence Diagram

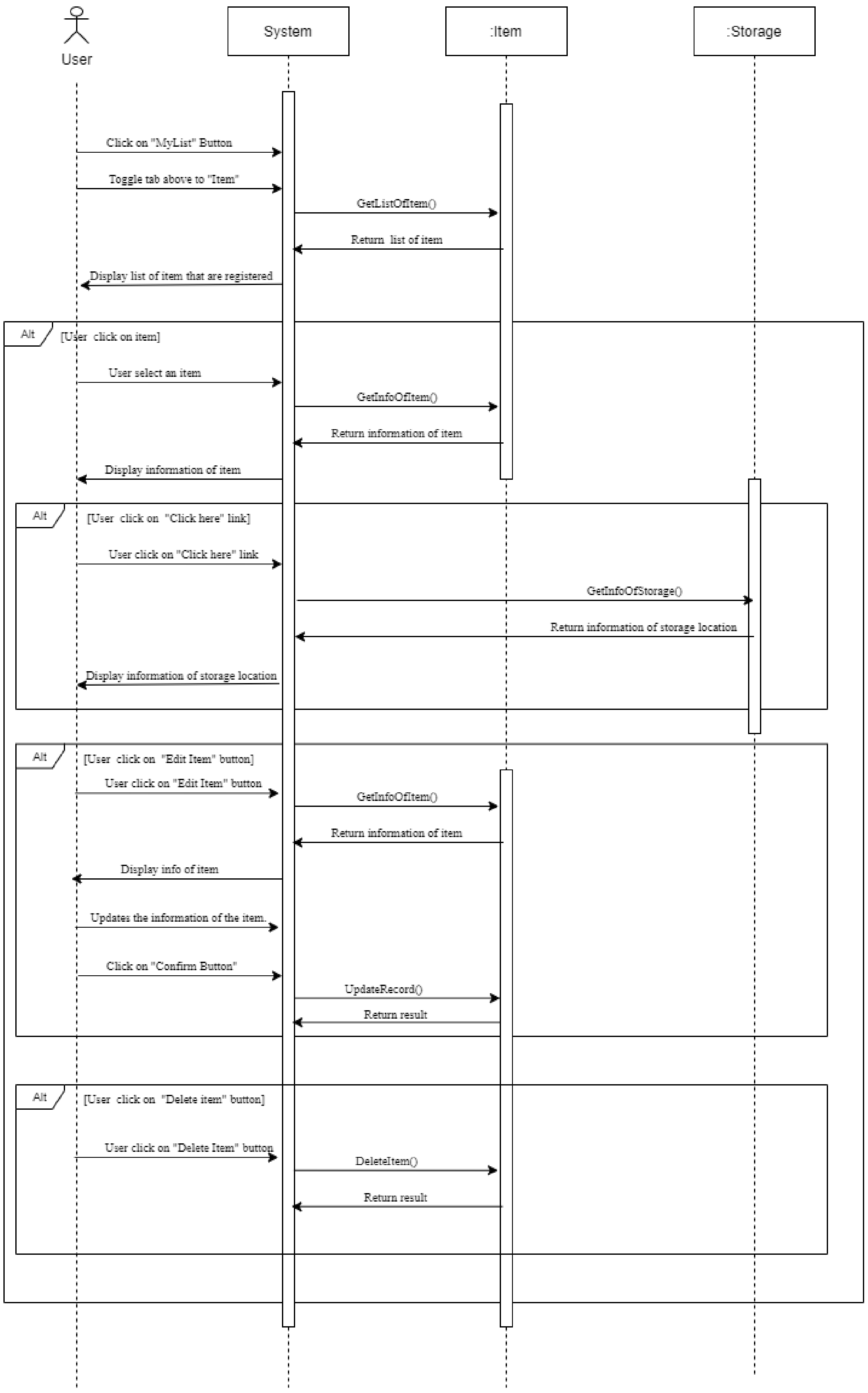

User
System
:Item
:Storage
Click on "MyList" Button
Toggle tab above to "Item"
GetListOfItem()
Return list of item
Display list of item that are registered
Alt
[User click on item]
User select an item
GetInfoOfItem()
Return information of item
Display information of item
Alt
[User click on "Click here" link]
User click on "Click here" link
GetInfoOfStorage()
Return information of storage location
Display information of storage location
Alt
[User click on "Edit Item" button]
User click on "Edit Item" button
GetInfoOfItem()
Return information of item
Display info of item
Updates the information of the item.
Click on "Confirm Button"
UpdateRecord()
Return result
Alt
[User click on "Delete item" button]
User click on "Delete Item" button
DeleteItem()
Return result

Figure 3.2.4.5 View Item List Sequence Diagram

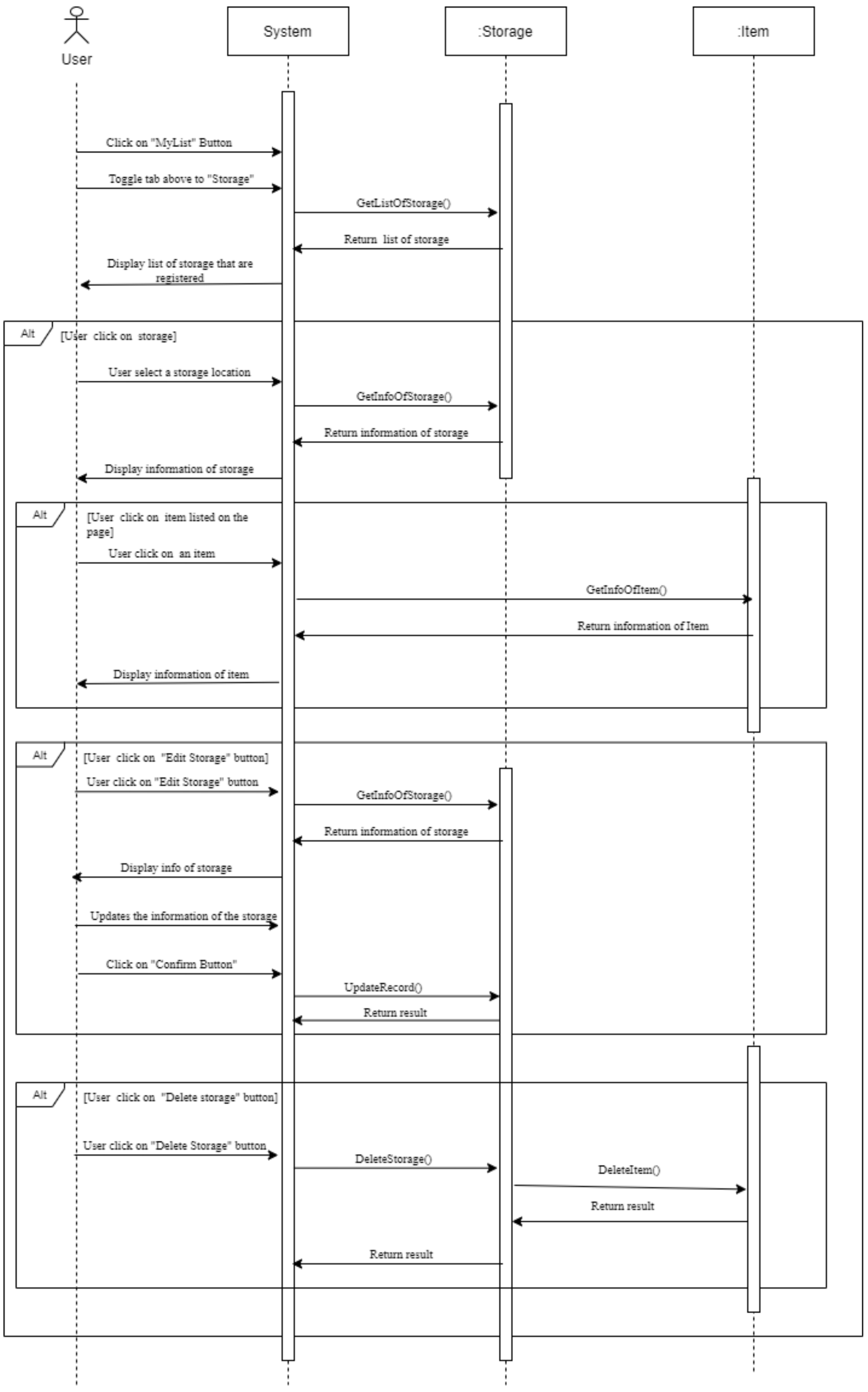

User
System
:Storage
:Item
Click on "MyList" Button
Toggle tab above to "Storage"
GetListOfStorage()
Return list of storage
Display list of storage that are registered
Alt
[User click on storage]
User select a storage location
GetInfoOfStorage()
Return information of storage
Display information of storage
Alt
[User click on item listed on the page]
User click on an item
GetInfoOfItem()
Return information of Item
Display information of item
Alt
[User click on "Edit Storage" button]
User click on "Edit Storage" button
GetInfoOfStorage()
Return information of storage
Display info of storage
Updates the information of the storage
Click on "Confirm Button"
UpdateRecord()
Return result
Alt
[User click on "Delete storage" button]
User click on "Delete Storage" button
DeleteStorage()
DeleteItem()
Return result
Return result

Figure 3.2.4.6 View Storage Location List Sequence Diagram

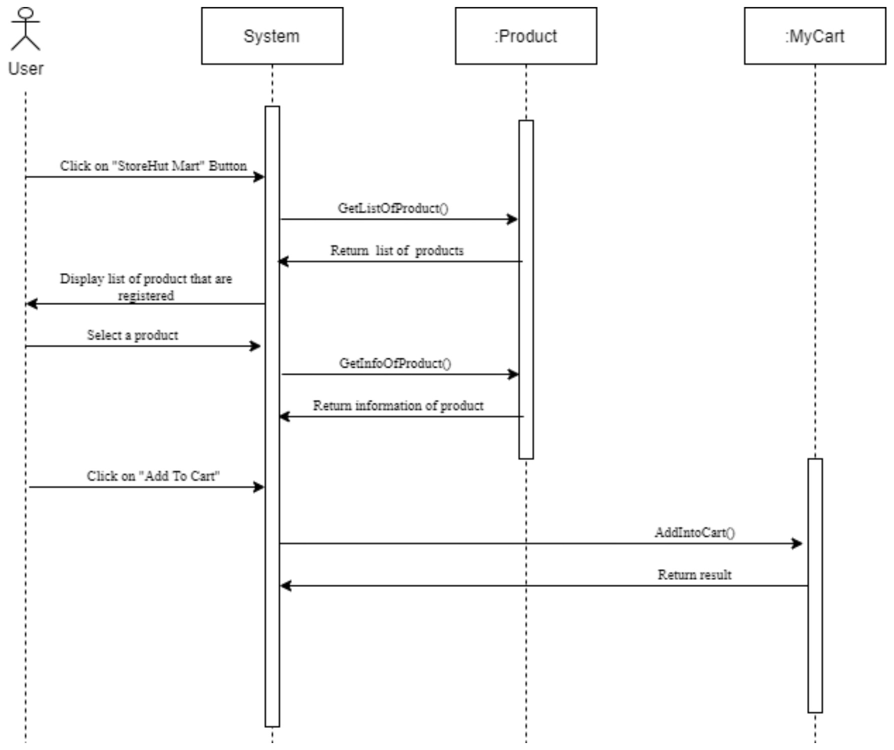


Figure 3.2.4.7 Add Product Into Cart Sequence Diagram

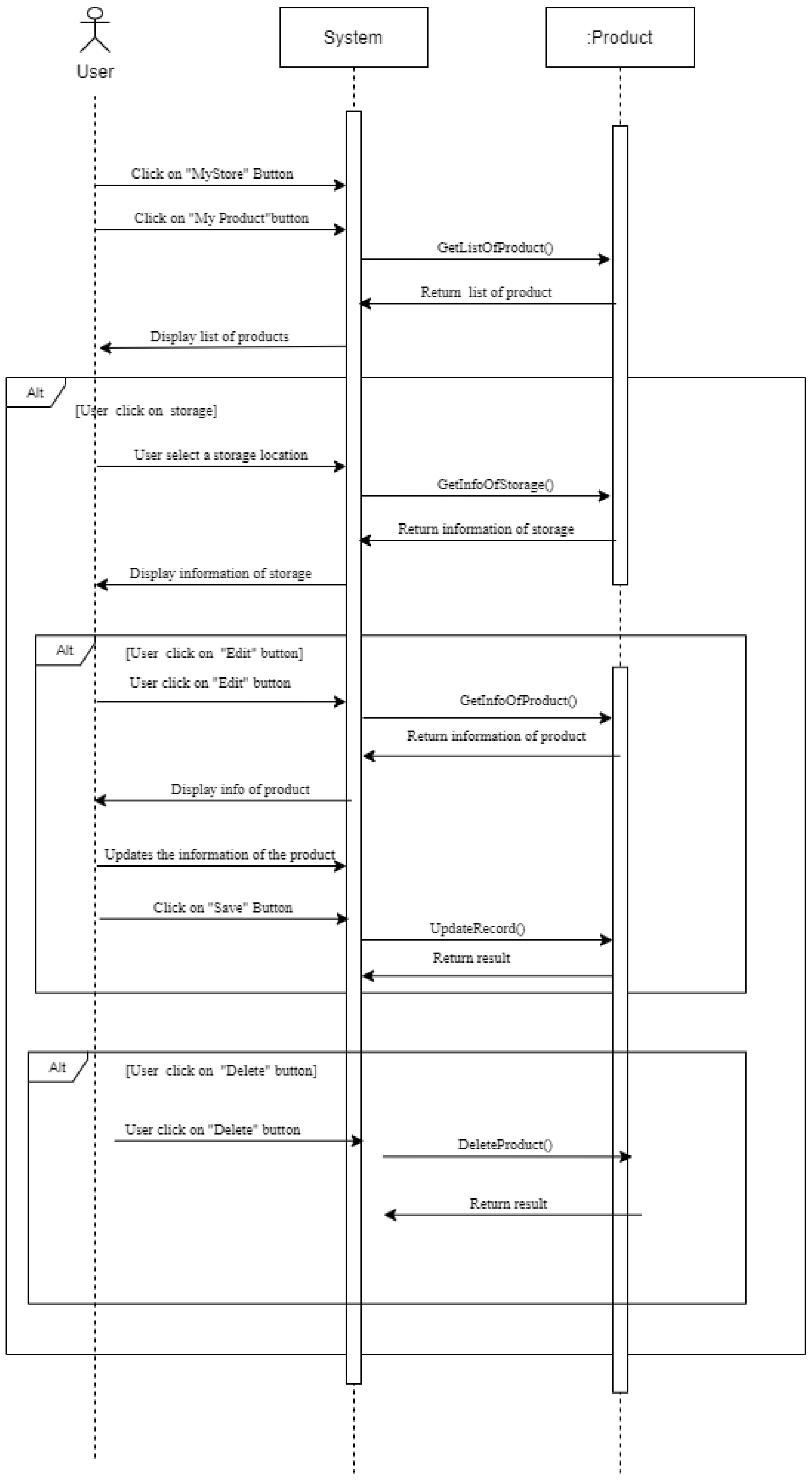

User
System
:Product
Click on "MyStore" Button
Click on "My Product"button
GetListOfProduct()
Return list of product
Display list of products
Alt
[User click on storage]
User select a storage location
GetInfoOfStorage()
Return information of storage
Display information of storage
Alt
[User click on "Edit" button]
User click on "Edit" button
GetInfoOfProduct()
Return information of product
Display info of product
Updates the information of the product
Click on "Save" Button
UpdateRecord()
Return result
Alt
[User click on "Delete" button]
User click on "Delete" button
DeleteProduct()
Return result

Figure 3.2.4.9 View On-Shelf Product Sequence Diagram

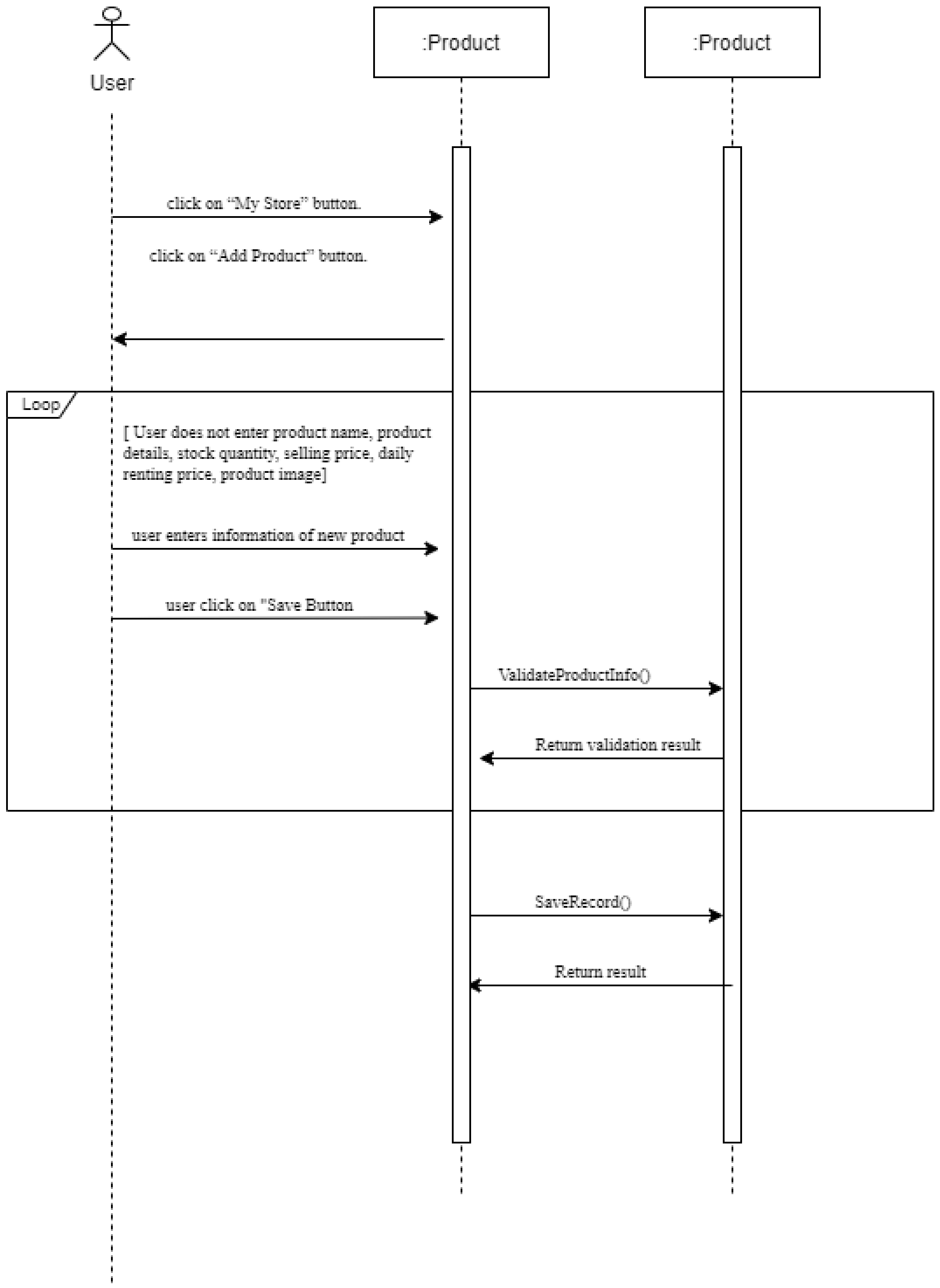

User
:Product
:Product
click on "My Store" button.
click on "Add Product" button.
Loop
[User does not enter product name, product details, stock quantity, selling price, daily renting price, product image]
user enters information of new product
user click on "Save Button
ValidateProductInfo()
Return validation result
SaveRecord()
Return result

Figure 3.2.4.10 Add New Product to Shelf Sequence Diagram

3.3 System Design

3.3.1 System Flow Chart

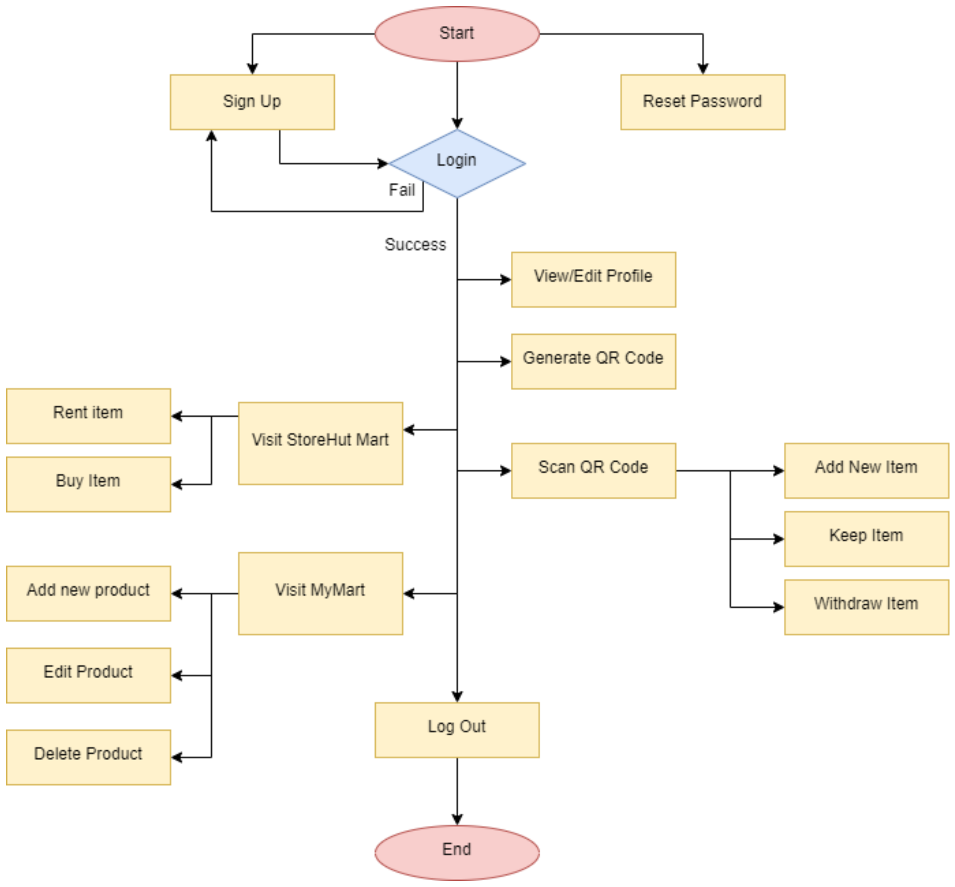


Figure 3.3.1.1 System Flow Chart

Our proposed approach is a mobile application that could help the users to track the storage location of the items, at the same time showing an overall view of the items owned by the users. Besides, we had built a simple e-commerce system in the application to allow users to buy and rent equipment, tools, ingredients and others product from the seller around the country. The figure above is showing the system flow chart of StoreHut Application which displays how the data flows in the system and how the decisions are made to control events. By referring to Figure 1.5.1, user must register a user account before accessing the features in the application. If user has forgotten the password of an existing account, he can request for password resetting. After successfully login to the application, user can select the desired features, such as view or edit user profile, generate QR Code, Scan QR Code, visit StorehutMart, and visit MyMart.

Firstly, generating QR Code is a function to generate a QR Code which is embedded with the information of storage location. After generating the QR Code, it can be saved or printed out to be pasted to the relevant storage location. By selecting this function, user will be prompted with some input regarding to the general information of the storage location or buckets before the QR Code can be generated.

Next, after scanning the QR Code which is attached to a storage location such as a storage box, the system will validate the QR Code. After that, the system will display three different functions such as add new item, keep item and withdraw item on the screen. By selecting add new item, user will be navigated to a page which display a form to add new item into the particular storage location. After submitting the form, user can place the item into the storage location. Hence, the record of that item will be stored under the storage location which is corresponding to the scanned QR Code.

Another interesting feature in this application is the e-commerce system named StoreHut Mart. User can find various type of tools and equipment by scrolling through the page and they can add the item into the cart if they found the desired one. The special service provided by this system is the renting or buying features. In this case, seller can choose to rent or sell the items to his customer. Next, user can check out the item by making payment to the owner while the seller will send out the item to his customer after receiving the payment. On the other side, every user can sell or rent their products on StoreHutMart by registering the products on MyMart page. Other than creating a new product, user can edit or delete an existing products according to their needs.

3.3.2 ERD Diagram

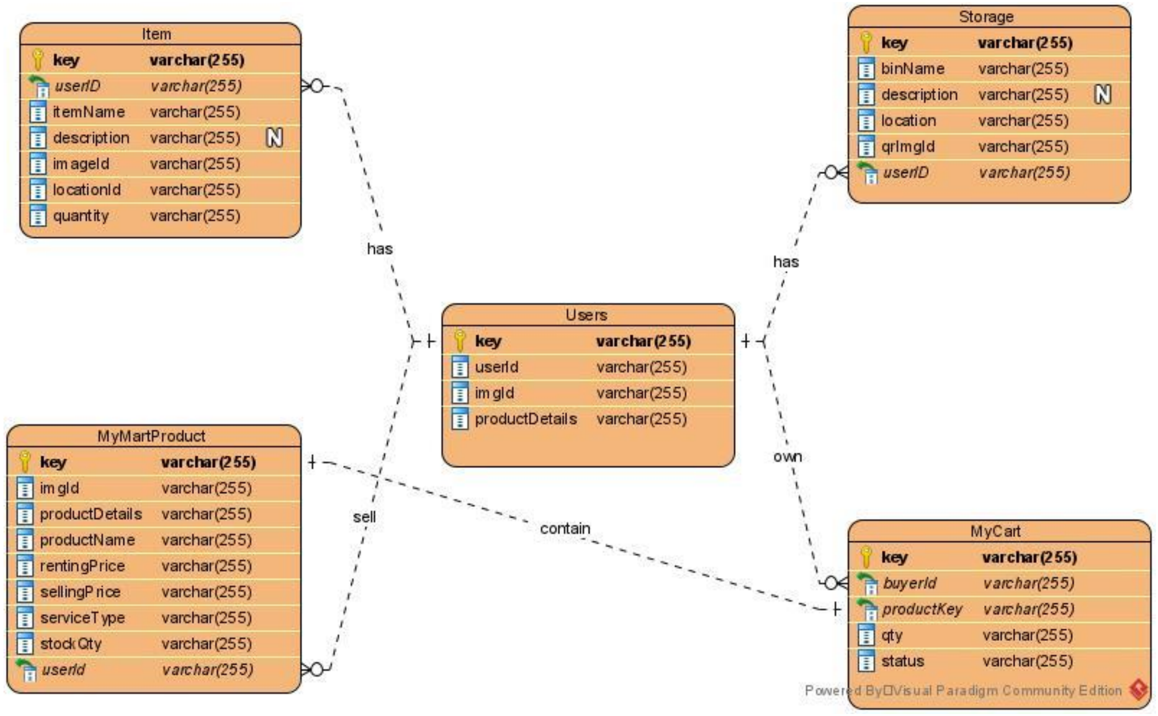


Figure 3.3.2.1 Entity Relationship Diagram

3.3.3 Data Dictionary

Users

Field Name	Data Type	Field Size	Null ?	PK/ FK	Description	Example
key	VARCHAR2	255	No	PK	Unique ID represented each user	9V8NMwTz0wchCtquggLfroe2iey1
username	VARCHAR2	255	No	PK	Display name of user	Happy1234
email	VARCHAR2	255	No	-	Email address of user	Wanjing2000@gmail.com

phone	VARCHAR2	255	No	-	Contact number of user	0126678900
token	VARCHAR2	255	No	-	Firebase auth token which is used to authenticate via the signInWithCustomToken() method.	fYLL94LqTZqq_jIh9bJOKT:APA91bHVkVLbqIPNv_UmMvzC3914rkM0tmkMJHTRvmrXtFu84nAlLZzCaDEuHHSdZoQzSuoWFqPI-oBuHPWAeXVF107DAjlZ4NOCIc40RFlR__AuollzAwoyiorFYNG1CjXQzaRstWnd

Figure 3.3.3.1 Data dictionary of Users table

Item

Field Name	Data Type	Field Size	Null?	PK/ FK	Description	Example
key	VARCHAR2	255	No	PK	Unique ID represented each item	-MxjojBZUJW1ji0cHdxt
itemName	VARCHAR2	255	No	PK	Display name of item	Hammer
description	VARCHAR2	255	Yes	-	Description of the item	Strip rubber hammer
quantity	VARCHAR2	255	No	-	Stock quantity of the item	2

imageId	VARCHAR2	255	Yes	-	The image id generated by system	9044950_no_image_icon.png
locationId	VARCHAR2	255	No	-	The corresponding location id which is storing the item	-MxjoT2GOyesz1W4Yc0H
userID	VARCHAR2	255	No	PK	Unique ID represented each user	9V8NMwTz0whCtquggLfroe2iey1

Figure 3.3.3.2 Data dictionary of Item table

Storage

Field Name	**Data Type**	**Field Size**	**Null?**	**PK/ FK**	**Description**	**Example**
key	VARCHAR2	255	No	PK	Unique ID represented each storage location	-MxjojBZUJW1ji0cHdxt
binName	VARCHAR2	255	No	PK	Display name of storage location	MyBucket 1
description	VARCHAR2	255	Yes	-	Description of the storage location	Red bucket without cover
qrImgId	VARCHAR2	255	No	-	The image id of	1646848500386.jpg

					corresponding QR Code	
location	VARCHAR2	255	No	-	The description of exact physical location that the storage container placed at	-MxjoT2GOyesz1W4Yc0H
userID	VARCHAR2	255	No	PK	Unique ID represented each user	9V8NMwTz0wchCtquggLfroe2iey1

Figure 3.3.3.3 Data dictionary of Storage table

MyMartProduct

Field Name	Data Type	Field Size	Null?	PK/ FK	Description	Example
key	VARCHAR2	255	No	PK	Unique ID represented each on-shelf products	-MxjojBZUJW1ji0cHdxt
productName	VARCHAR2	255	No	PK	Display name of product	SENSUI JAPAN High Quality Hok Tsui Steel Claw Hammer Magnetic Tukul Hammer
productDetails	VARCHAR2	255	No	-	Description of the product	# Durable Steel Claw handle #

						Strip rubber # Forged steel head
imgId	VARCHAR2	255	No	-	The image id of product	1646848500386.jpg
renting Price	VARCHAR2	255	No	-	The renting price per day	23
selling Price	VARCHAR2	255	No	-	The selling price of products	5
service Type	VARCHAR2	255	No	-	The mode of service such as only rent, only sell and also rent and sell	rent
stockQty	VARCHAR2	255	No	-	The quantity of stock available	59
userID	VARCHAR2	255	No	PK	Unique ID represented each seller	9V8NMwTz0wchCtquggLfroe2iey1

Figure 3.3.3.4 Data dictionary of MyMartProduct table

MyCart

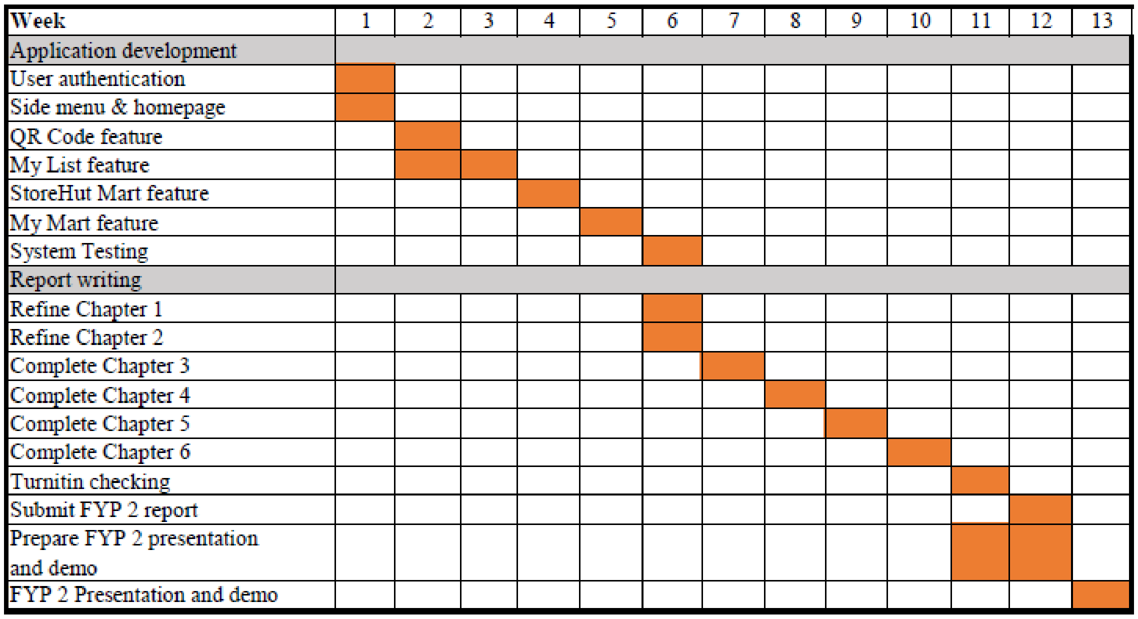
Field Name	**Data Type**	**Field Size**	**Null?**	**PK/ FK**	**Description**	**Example**
key	VARCHAR2	255	No	PK	Unique ID represented each product in the cart	-MxjojBZUJW1ji0cHdxt

productKey	VARCHAR2	255	No	PK	Unique ID represented each product	-MxjojBZUJW1ji0cHdxt
qty	VARCHAR2	255	No	-	Quantity of the product in cart	2
status	VARCHAR2	255	No	-	Status of the product	In_cart
buyerId	VARCHAR2	255	No	PK	Unique ID represented the user who add it into cart	9V8NMwTz0wchCtquggLfroe2iey1

Figure 3.3.3.5 Data dictionary of MyCart table

CHAPTER 4 SYSTEM IMPLEMENTATION

4.1 Project Timeline



Week	1	2	3	4	5	6	7	8	9	10	11	12	13
Application development													
User authentication	🟧												
Side menu & homepage	🟧												
QR Code feature		🟧											
My List feature		🟧	🟧										
StoreHut Mart feature				🟧									
My Mart feature					🟧								
System Testing						🟧							
Report writing													
Refine Chapter 1						🟧							
Refine Chapter 2						🟧							
Complete Chapter 3							🟧						
Complete Chapter 4								🟧					
Complete Chapter 5									🟧				
Complete Chapter 6										🟧			
Turnitin checking											🟧		
Submit FYP 2 report												🟧	
Prepare FYP 2 presentation and demo											🟧	🟧	
FYP 2 Presentation and demo													🟧

Table 4.1.1 Project Timeline

4.2 Technologies and tools involved

4.2.1 Hardware requirement

Description	Minimum Requirement
Operating System	- 64-bit Microsoft® Windows® 8/10 or - MacOS® 10.14 (Mojave) or higher or - Any 64-bit Linux distribution that supports Gnome, KDE, or Unity DE; GNU C Library (glibc) 2.31 or later.
Processor	2nd generation Intel Core or newer
RAM	8GBytes or more
Available disk space	8GBytes or more
Screen resolution	At least 1280 x 800

Table 4.2.1.1 Hardware requirement

4.2.2 Software requirement

- Android Studio

The software required for the development of the proposed application is Android Studio. which provides all tools needed for Android App development. The latest version of Android Studio is 4.2.2, which is updated in June 2021. Furthermore, various programming skills are needed to build an Android application, such as understanding Java, XML, Android SDK, databases, utilisation of Android Studio, and User Interface Design.

- Device’s camera

The user must allow the mobile application's camera permission to scan the QR Code in order to catch the information embedded in the QR Code attached to the storage box.

- Files in device

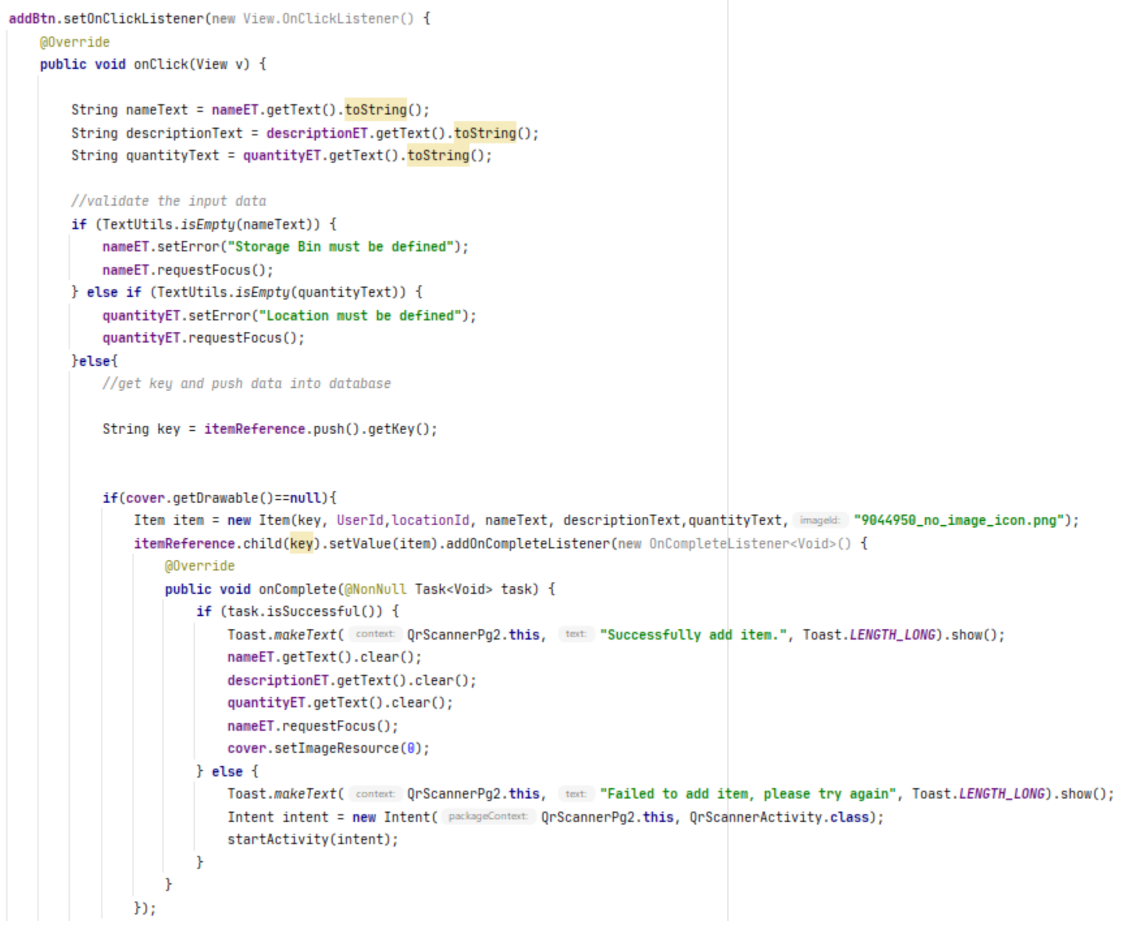
The photo album and files will be accessed when user attaches the images into the application. Besides, the QR Code generated in application can be downloaded into the local storage of device.

4.2.3 Connectivity Requirement

The internet connectivity is needed for the proposed application to function well. The proposed application is using the Google Firebase for backend data storage, real-time synchronization, and user-event logging. Although the firebase application can work when it temporarily loses internet connection, most of the features of the proposed application can only be accessible online, for example displaying list of products and storages, generate QR code, and the feature of take and keep item. Hence, an internet connection is needed for the better operation of the application.

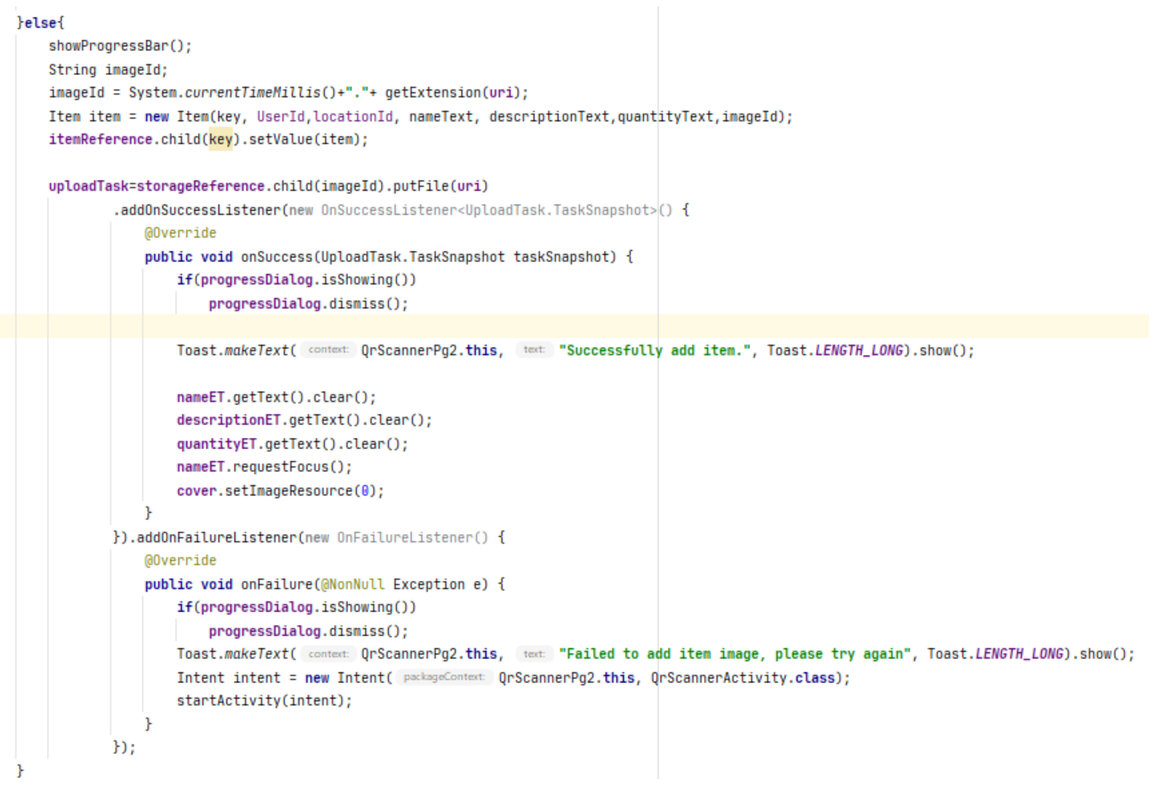
4.3 Code Snippet

4.3.1 Code snippet of the procedure on uploading data and image to Firebase Realtime Database and Firebase Storage respectively.



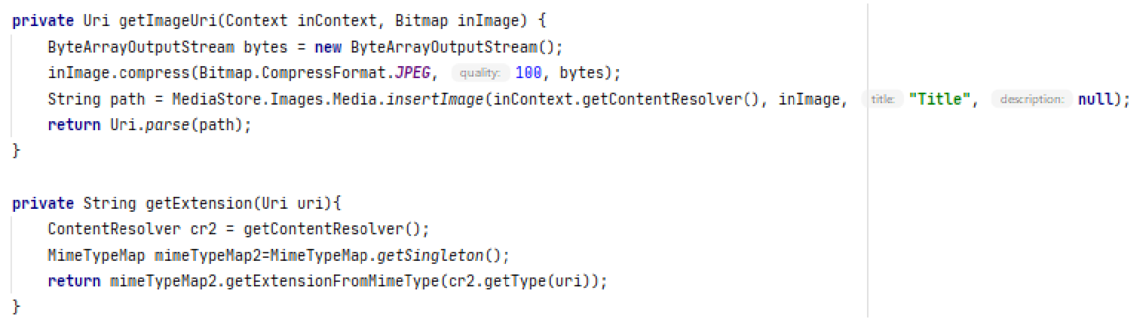
```
addBtn.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {

        String nameText = nameET.getText().toString();
        String descriptionText = descriptionET.getText().toString();
        String quantityText = quantityET.getText().toString();

        //validate the input data
        if (TextUtils.isEmpty(nameText)) {
            nameET.setError("Storage Bin must be defined");
            nameET.requestFocus();
        } else if (TextUtils.isEmpty(quantityText)) {
            quantityET.setError("Location must be defined");
            quantityET.requestFocus();
        }else{
            //get key and push data into database

            String key = itemReference.push().getKey();


            if(cover.getDrawable()==null){
                Item item = new Item(key, UserId,locationId, nameText, descriptionText,quantityText, imageId: "9044950_no_image_icon.png");
                itemReference.child(key).setValue(item).addOnCompleteListener(new OnCompleteListener<Void>() {
                    @Override
                    public void onComplete(@NonNull Task<Void> task) {
                        if (task.isSuccessful()) {
                            Toast.makeText( context: QrScannerPg2.this,  text: "Successfully add item.", Toast.LENGTH_LONG).show();
                            nameET.getText().clear();
                            descriptionET.getText().clear();
                            quantityET.getText().clear();
                            nameET.requestFocus();
                            cover.setImageResource(0);
                        } else {
                            Toast.makeText( context: QrScannerPg2.this,  text: "Failed to add item, please try again", Toast.LENGTH_LONG).show();
                            Intent intent = new Intent( packageContext: QrScannerPg2.this, QrScannerActivity.class);
                            startActivity(intent);
                        }
                    }
                });
```

Figure 4.3.1.1 Code snippet of uploading data to Firebase Realtime Database



```
}else{
    showProgressBar();
    String imageId;
    imageId = System.currentTimeMillis()+"."+ getExtension(uri);
    Item item = new Item(key, UserId,locationId, nameText, descriptionText,quantityText,imageId);
    itemReference.child(key).setValue(item);

    uploadTask=storageReference.child(imageId).putFile(uri)
            .addOnSuccessListener(new OnSuccessListener<UploadTask.TaskSnapshot>() {
                @Override
                public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {
                    if(progressDialog.isShowing())
                        progressDialog.dismiss();

                    Toast.makeText( context: QrScannerPg2.this,  text: "Successfully add item.", Toast.LENGTH_LONG).show();

                    nameET.getText().clear();
                    descriptionET.getText().clear();
                    quantityET.getText().clear();
                    nameET.requestFocus();
                    cover.setImageResource(0);
                }
            }).addOnFailureListener(new OnFailureListener() {
                @Override
                public void onFailure(@NonNull Exception e) {
                    if(progressDialog.isShowing())
                        progressDialog.dismiss();
                    Toast.makeText( context: QrScannerPg2.this,  text: "Failed to add item image, please try again", Toast.LENGTH_LONG).show();
                    Intent intent = new Intent( packageContext: QrScannerPg2.this, QrScannerActivity.class);
                    startActivity(intent);
                }
            });
}
```

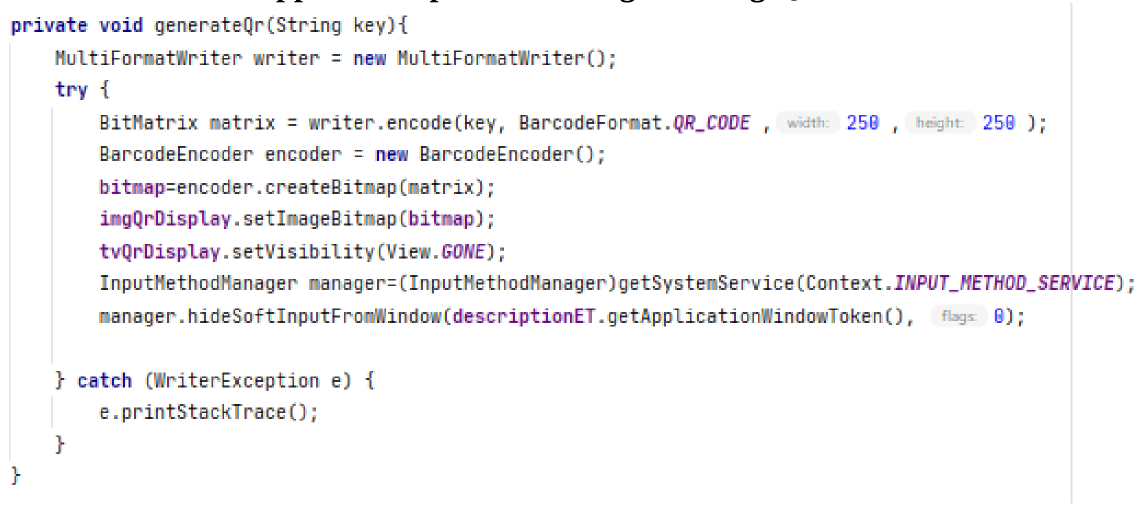
Figure 4.3.1.2 Code snippet of uploading Image to Firebase Storage



```
private Uri getImageUri(Context inContext, Bitmap inImage) {
    ByteArrayOutputStream bytes = new ByteArrayOutputStream();
    inImage.compress(Bitmap.CompressFormat.JPEG,  quality: 100, bytes);
    String path = MediaStore.Images.Media.insertImage(inContext.getContentResolver(), inImage,  title: "Title",  description: null);
    return Uri.parse(path);
}

private String getExtension(Uri uri){
    ContentResolver cr2 = getContentResolver();
    MimeTypeMap mimeTypeMap2=MimeTypeMap.getSingleton();
    return mimeTypeMap2.getExtensionFromMimeType(cr2.getType(uri));
}
```

Figure 4.3.1.3 Code snippet of getting image Uri

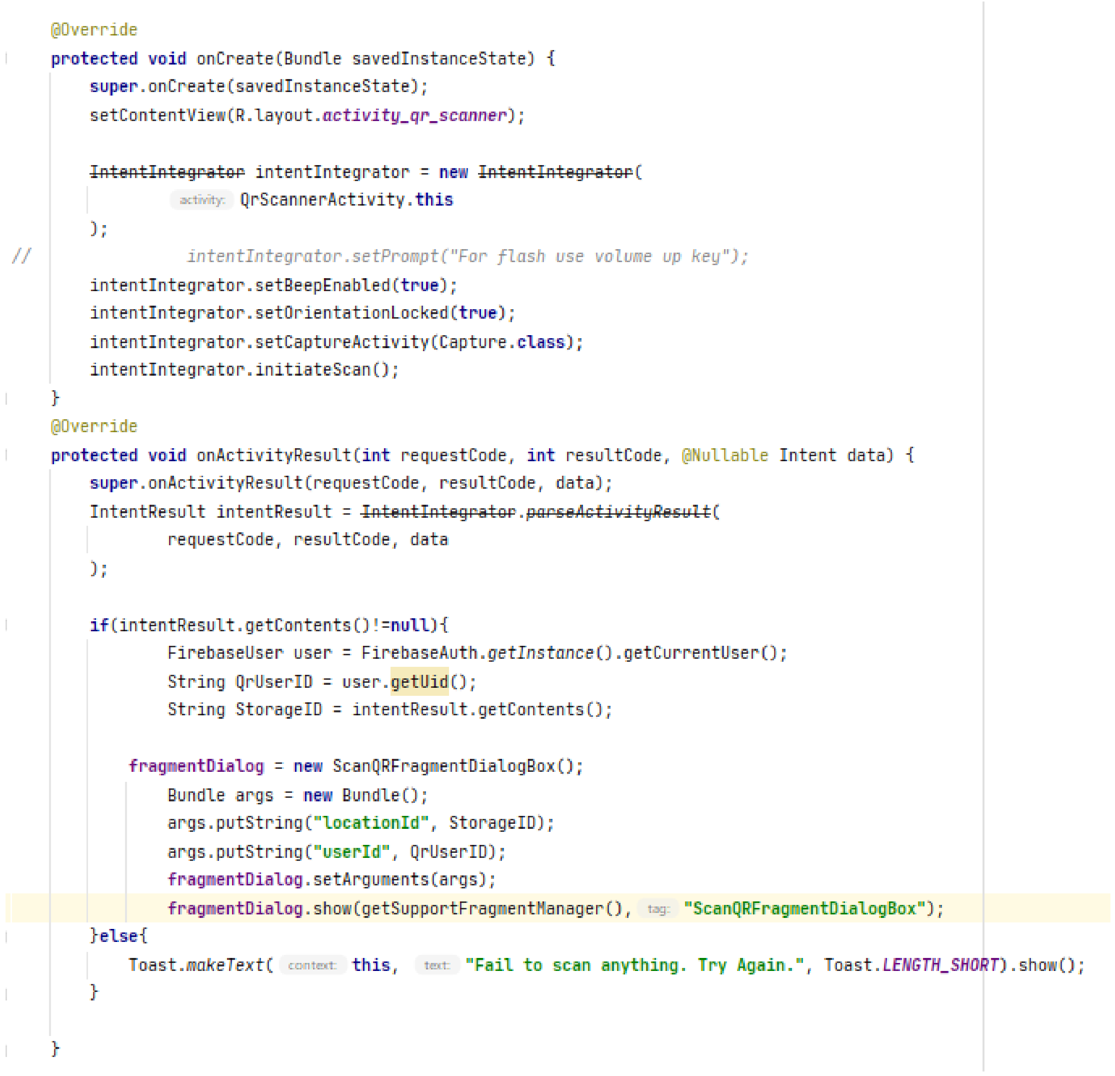
4.3.2 Code snippet of the procedure on generating QR Code



```
private void generateQr(String key){
    MultiFormatWriter writer = new MultiFormatWriter();
    try {
        BitMatrix matrix = writer.encode(key, BarcodeFormat.QR_CODE , width: 250 , height: 250 );
        BarcodeEncoder encoder = new BarcodeEncoder();
        bitmap=encoder.createBitmap(matrix);
        imgQrDisplay.setImageBitmap(bitmap);
        tvQrDisplay.setVisibility(View.GONE);
        InputMethodManager manager=(InputMethodManager)getSystemService(Context.INPUT_METHOD_SERVICE);
        manager.hideSoftInputFromWindow(descriptionET.getApplicationWindowToken(), flags: 0);

    } catch (WriterException e) {
        e.printStackTrace();
    }
}
```

Figure 4.3.2.1 Code snippet of generating QR Code with a given key of storage location

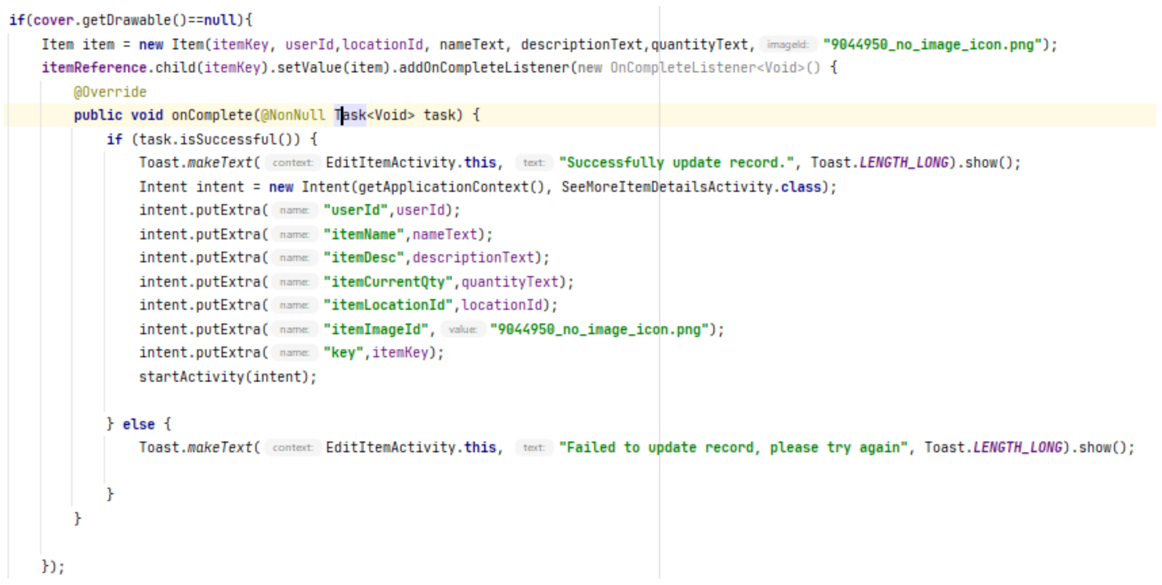
4.3.3 Code snippet of the procedure on creating QR Code Scanner



```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_qr_scanner);

    ~~IntentIntegrator~~ intentIntegrator = new ~~IntentIntegrator~~(
            activity: QrScannerActivity.this
    );
//              intentIntegrator.setPrompt("For flash use volume up key");
    intentIntegrator.setBeepEnabled(true);
    intentIntegrator.setOrientationLocked(true);
    intentIntegrator.setCaptureActivity(Capture.class);
    intentIntegrator.initiateScan();
}
@Override
protected void onActivityResult(int requestCode, int resultCode, @Nullable Intent data) {
    super.onActivityResult(requestCode, resultCode, data);
    IntentResult intentResult = ~~IntentIntegrator.parseActivityResult~~(
            requestCode, resultCode, data
    );

    if(intentResult.getContents()!=null){
            FirebaseUser user = FirebaseAuth.getInstance().getCurrentUser();
            String QrUserID = user.getUid();
            String StorageID = intentResult.getContents();

        fragmentDialog = new ScanQRFragmentDialogBox();
            Bundle args = new Bundle();
            args.putString("locationId", StorageID);
            args.putString("userId", QrUserID);
            fragmentDialog.setArguments(args);
            fragmentDialog.show(getSupportFragmentManager(), tag: "ScanQRFragmentDialogBox");
    }else{
        Toast.makeText( context: this,  text: "Fail to scan anything. Try Again.", Toast.LENGTH_SHORT).show();
    }

}
```

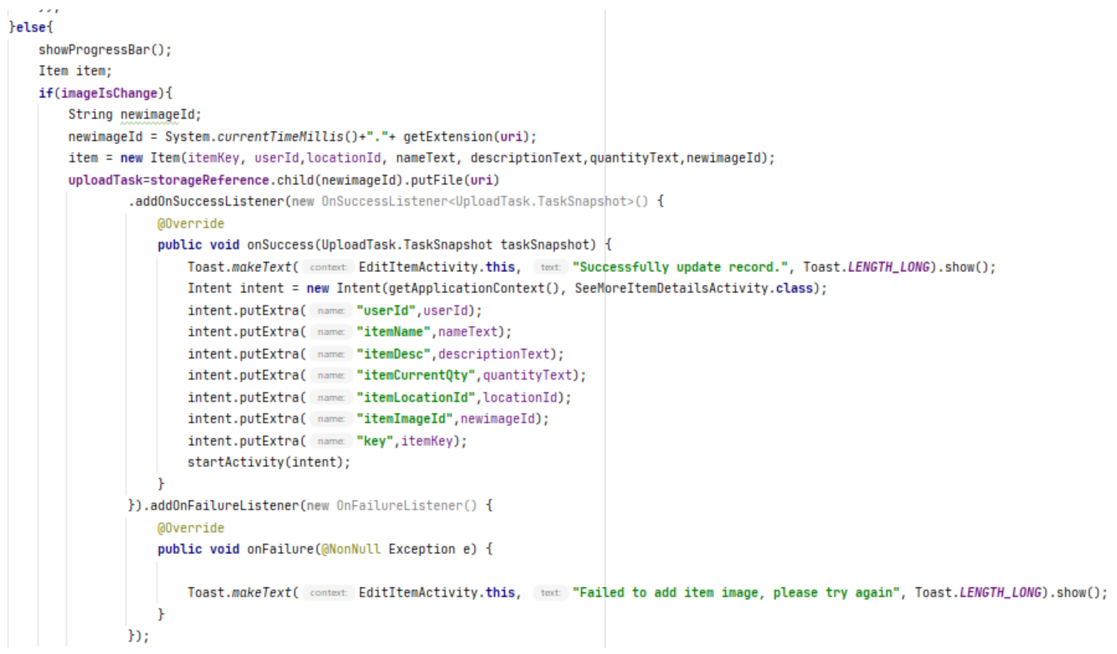
Figure 4.3.3.1 Code snippet of creating QR Scanner

4.3.4 Code snippet of procedure on handling three different data updating situation – original image is changed to another one, no changes on the original image and no image attached



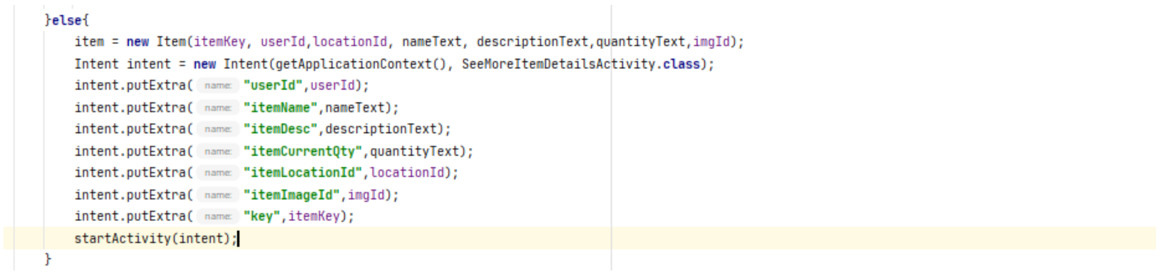
```
if(cover.getDrawable()==null){
    Item item = new Item(itemKey, userId,locationId, nameText, descriptionText,quantityText, imageId: "9044950_no_image_icon.png");
    itemReference.child(itemKey).setValue(item).addOnCompleteListener(new OnCompleteListener<Void>() {
        @Override
        public void onComplete(@NonNull Task<Void> task) {
            if (task.isSuccessful()) {
                Toast.makeText( context: EditItemActivity.this,  text: "Successfully update record.", Toast.LENGTH_LONG).show();
                Intent intent = new Intent(getApplicationContext(), SeeMoreItemDetailsActivity.class);
                intent.putExtra( name: "userId",userId);
                intent.putExtra( name: "itemName",nameText);
                intent.putExtra( name: "itemDesc",descriptionText);
                intent.putExtra( name: "itemCurrentQty",quantityText);
                intent.putExtra( name: "itemLocationId",locationId);
                intent.putExtra( name: "itemImageId", value: "9044950_no_image_icon.png");
                intent.putExtra( name: "key",itemKey);
                startActivity(intent);

            } else {
                Toast.makeText( context: EditItemActivity.this,  text: "Failed to update record, please try again", Toast.LENGTH_LONG).show();

            }
        }

    });
```

Figure 4.3.4.1 Code snippet of handling situation where no image attached



```
}else{
    showProgressBar();
    Item item;
    if(imageIsChange){
        String newimageId;
        newimageId = System.currentTimeMillis()+"."+ getExtension(uri);
        item = new Item(itemKey, userId,locationId, nameText, descriptionText,quantityText,newimageId);
        uploadTask=storageReference.child(newimageId).putFile(uri)
                .addOnSuccessListener(new OnSuccessListener<UploadTask.TaskSnapshot>() {
                    @Override
                    public void onSuccess(UploadTask.TaskSnapshot taskSnapshot) {
                        Toast.makeText( context: EditItemActivity.this,  text: "Successfully update record.", Toast.LENGTH_LONG).show();
                        Intent intent = new Intent(getApplicationContext(), SeeMoreItemDetailsActivity.class);
                        intent.putExtra( name: "userId",userId);
                        intent.putExtra( name: "itemName",nameText);
                        intent.putExtra( name: "itemDesc",descriptionText);
                        intent.putExtra( name: "itemCurrentQty",quantityText);
                        intent.putExtra( name: "itemLocationId",locationId);
                        intent.putExtra( name: "itemImageId",newimageId);
                        intent.putExtra( name: "key",itemKey);
                        startActivity(intent);
                    }
                }).addOnFailureListener(new OnFailureListener() {
                    @Override
                    public void onFailure(@NonNull Exception e) {

                        Toast.makeText( context: EditItemActivity.this,  text: "Failed to add item image, please try again", Toast.LENGTH_LONG).show();
                    }
                });
```

Figure 4.3.4.2 Code snippet of handling situation where original image is changed to another one.



```
}else{
    item = new Item(itemKey, userId,locationId, nameText, descriptionText,quantityText,imgId);
    Intent intent = new Intent(getApplicationContext(), SeeMoreItemDetailsActivity.class);
    intent.putExtra( name: "userId",userId);
    intent.putExtra( name: "itemName",nameText);
    intent.putExtra( name: "itemDesc",descriptionText);
    intent.putExtra( name: "itemCurrentQty",quantityText);
    intent.putExtra( name: "itemLocationId",locationId);
    intent.putExtra( name: "itemImageId",imgId);
    intent.putExtra( name: "key",itemKey);
    startActivity(intent);
}
```

Figure 4.3.4.2 Code snippet of handling situation where original image is changed to another one.

4.3.5 Code snippet of producing scrollable grid view.

The RecyclerView is used in providing a scrollable list. There are a few major components in RecyclerView. Firstly, the Adapter is the component that connect the dataset to the item views displayed in RecyclerView by associating the position of each item view with the specific location of data source. Next, LayoutManager can positions the items within the RecyclerView. Lastly, the ViewHolder is a mandatory component in RecyclerView which build up the user interface of each items.

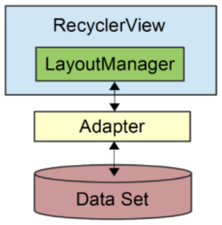
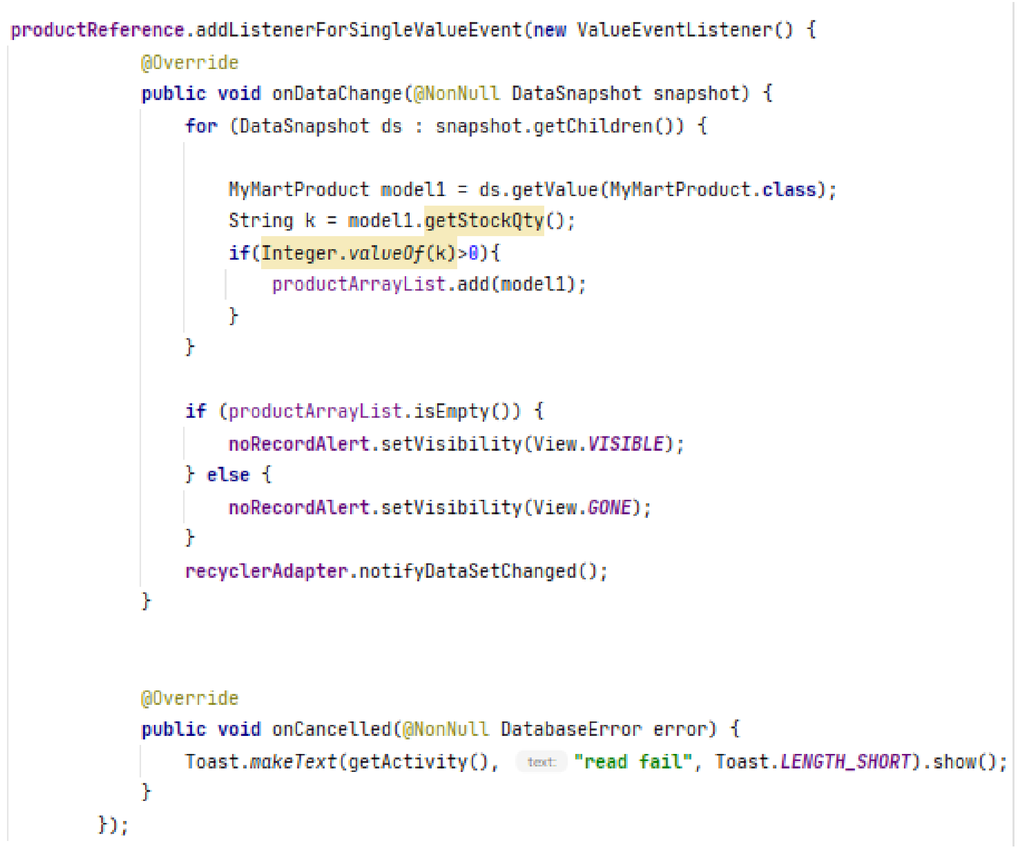
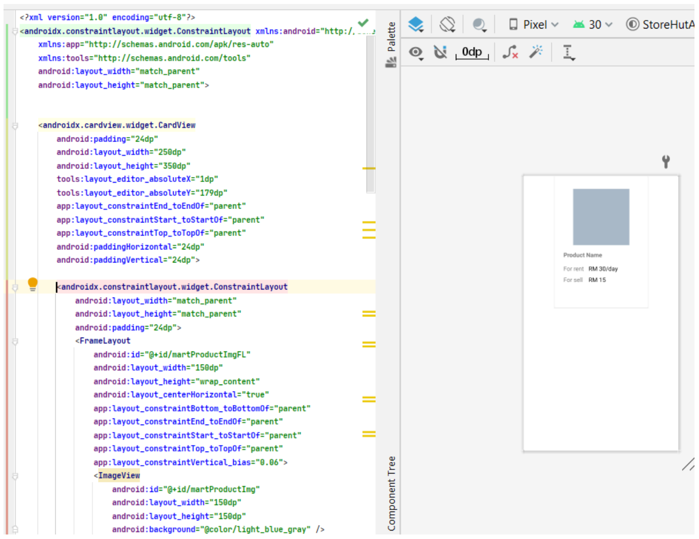


Figure 4.3.5.1 Components of RecyclerView



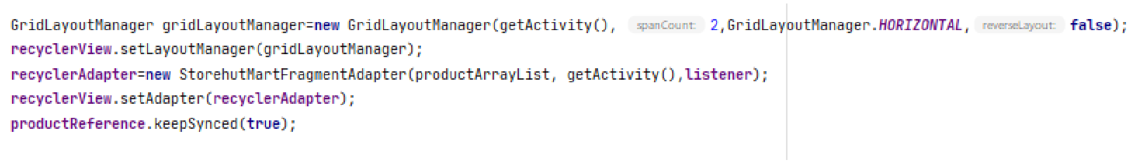
```
productReference.addListenerForSingleValueEvent(new ValueEventListener() {
            @Override
            public void onDataChange(@NonNull DataSnapshot snapshot) {
                for (DataSnapshot ds : snapshot.getChildren()) {

                    MyMartProduct model1 = ds.getValue(MyMartProduct.class);
                    String k = model1.getStockQty();
                    if(Integer.valueOf(k)>0){
                        productArrayList.add(model1);
                    }
                }

                if (productArrayList.isEmpty()) {
                    noRecordAlert.setVisibility(View.VISIBLE);
                } else {
                    noRecordAlert.setVisibility(View.GONE);
                }
                recyclerAdapter.notifyDataSetChanged();
            }


            @Override
            public void onCancelled(@NonNull DatabaseError error) {
                Toast.makeText(getActivity(), text: "read fail", Toast.LENGTH_SHORT).show();
            }
        });
```

Figure 4.3.5.2 Code snippet of getting the available product from Realtime Database and add into the array list for further usage.



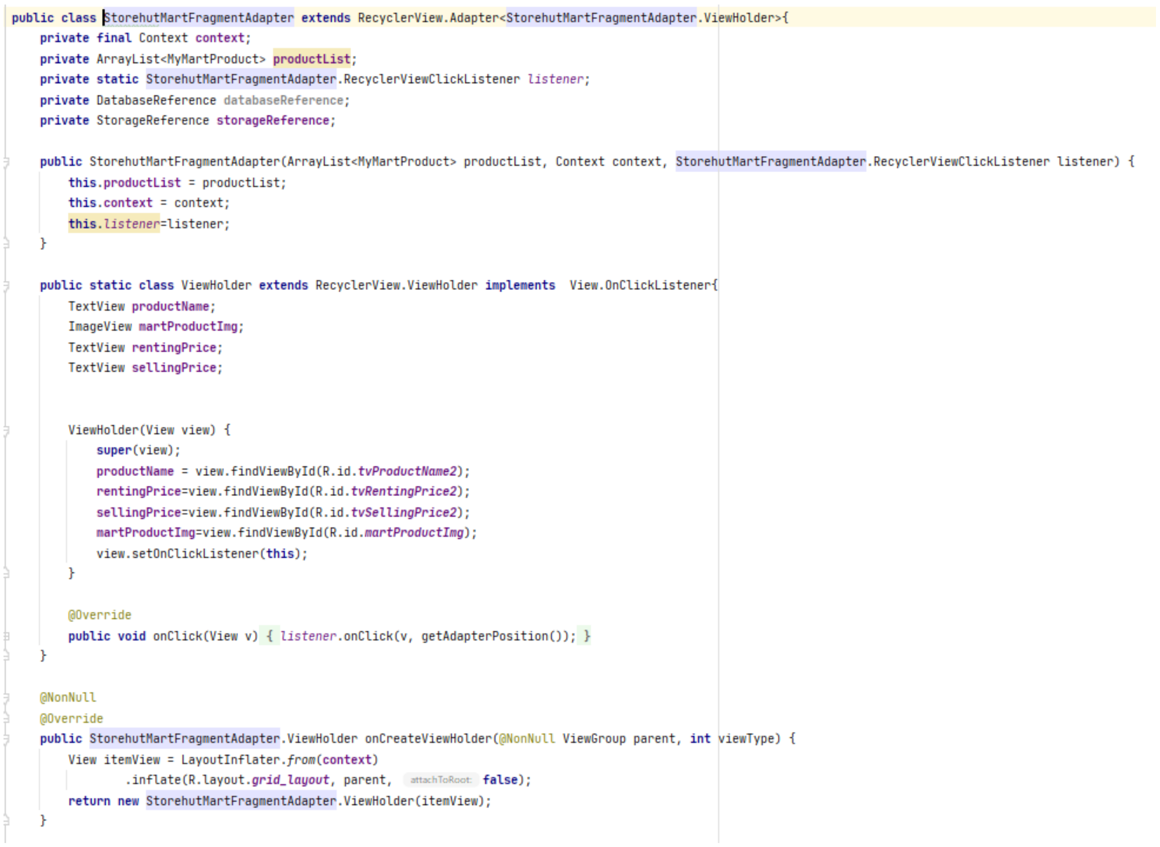
```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout xmlns:android="http:/
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent">


    <androidx.cardview.widget.CardView
        android:padding="24dp"
        android:layout_width="250dp"
        android:layout_height="350dp"
        tools:layout_editor_absoluteX="1dp"
        tools:layout_editor_absoluteY="179dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        android:paddingHorizontal="24dp"
        android:paddingVertical="24dp">

        <androidx.constraintlayout.widget.ConstraintLayout
            android:layout_width="match_parent"
            android:layout_height="match_parent"
            android:padding="24dp">
            <FrameLayout
                android:id="@+id/martProductImgFL"
                android:layout_width="150dp"
                android:layout_height="wrap_content"
                android:layout_centerHorizontal="true"
                app:layout_constraintBottom_toBottomOf="parent"
                app:layout_constraintEnd_toEndOf="parent"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toTopOf="parent"
                app:layout_constraintVertical_bias="0.06">
                <ImageView
                    android:id="@+id/martProductImg"
                    android:layout_width="150dp"
                    android:layout_height="150dp"
                    android:background="@color/light_blue_gray" />
```

Figure 4.3.5.3 Code snippet of designing the UI of Grid Layout which is used to show the products in StoreHut Mart



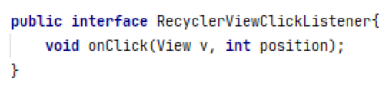
```
GridLayoutManager gridLayoutManager=new GridLayoutManager(getActivity(),  spanCount: 2,GridLayoutManager.HORIZONTAL,  reverseLayout: false);
recyclerView.setLayoutManager(gridLayoutManager);
recyclerAdapter=new StorehutMartFragmentAdapter(productArrayList, getActivity(),listener);
recyclerView.setAdapter(recyclerAdapter);
productReference.keepSynced(true);
```

Figure 4.3.5.4 Implementation of GridLayoutManager and RecyclerAdapter into recycler view

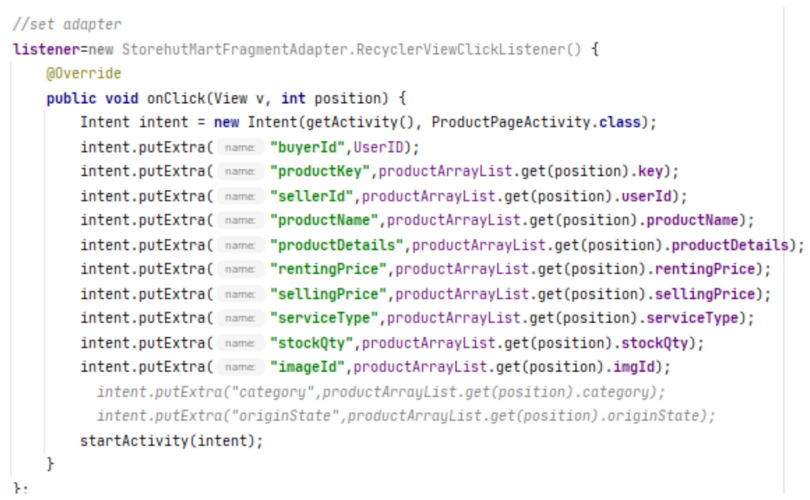


```
public class StorehutMartFragmentAdapter extends RecyclerView.Adapter<StorehutMartFragmentAdapter.ViewHolder>{
    private final Context context;
    private ArrayList<MyMartProduct> productList;
    private static StorehutMartFragmentAdapter.RecyclerViewClickListener listener;
    private DatabaseReference databaseReference;
    private StorageReference storageReference;

    public StorehutMartFragmentAdapter(ArrayList<MyMartProduct> productList, Context context, StorehutMartFragmentAdapter.RecyclerViewClickListener listener) {
        this.productList = productList;
        this.context = context;
        this.listener=listener;
    }

    public static class ViewHolder extends RecyclerView.ViewHolder implements  View.OnClickListener{
        TextView productName;
        ImageView martProductImg;
        TextView rentingPrice;
        TextView sellingPrice;


        ViewHolder(View view) {
            super(view);
            productName = view.findViewById(R.id.tvProductName2);
            rentingPrice=view.findViewById(R.id.tvRentingPrice2);
            sellingPrice=view.findViewById(R.id.tvSellingPrice2);
            martProductImg=view.findViewById(R.id.martProductImg);
            view.setOnClickListener(this);
        }

        @Override
        public void onClick(View v) { listener.onClick(v, getAdapterPosition()); }
    }

    @NonNull
    @Override
    public StorehutMartFragmentAdapter.ViewHolder onCreateViewHolder(@NonNull ViewGroup parent, int viewType) {
        View itemView = LayoutInflater.from(context)
                .inflate(R.layout.grid_layout, parent,  attachToRoot: false);
        return new StorehutMartFragmentAdapter.ViewHolder(itemView);
    }
```

Figure 4.3.5.5 Code snippet of ViewHolder of adapter class.



```
public interface RecyclerViewClickListener{
    void onClick(View v, int position);
}
```



```
//set adapter
listener=new StorehutMartFragmentAdapter.RecyclerViewClickListener() {
    @Override
    public void onClick(View v, int position) {
        Intent intent = new Intent(getActivity(), ProductPageActivity.class);
        intent.putExtra( name: "buyerId",UserID);
        intent.putExtra( name: "productKey",productArrayList.get(position).key);
        intent.putExtra( name: "sellerId",productArrayList.get(position).userId);
        intent.putExtra( name: "productName",productArrayList.get(position).productName);
        intent.putExtra( name: "productDetails",productArrayList.get(position).productDetails);
        intent.putExtra( name: "rentingPrice",productArrayList.get(position).rentingPrice);
        intent.putExtra( name: "sellingPrice",productArrayList.get(position).sellingPrice);
        intent.putExtra( name: "serviceType",productArrayList.get(position).serviceType);
        intent.putExtra( name: "stockQty",productArrayList.get(position).stockQty);
        intent.putExtra( name: "imageId",productArrayList.get(position).imgId);
          intent.putExtra("category",productArrayList.get(position).category);
          intent.putExtra("originState",productArrayList.get(position).originState);
        startActivity(intent);
    }
};
```

Figure 4.3.5.6 Code snippet of onClick() listener implemented on each component of recycler view.

4.4 User Interface

4.4.1 Dashboard

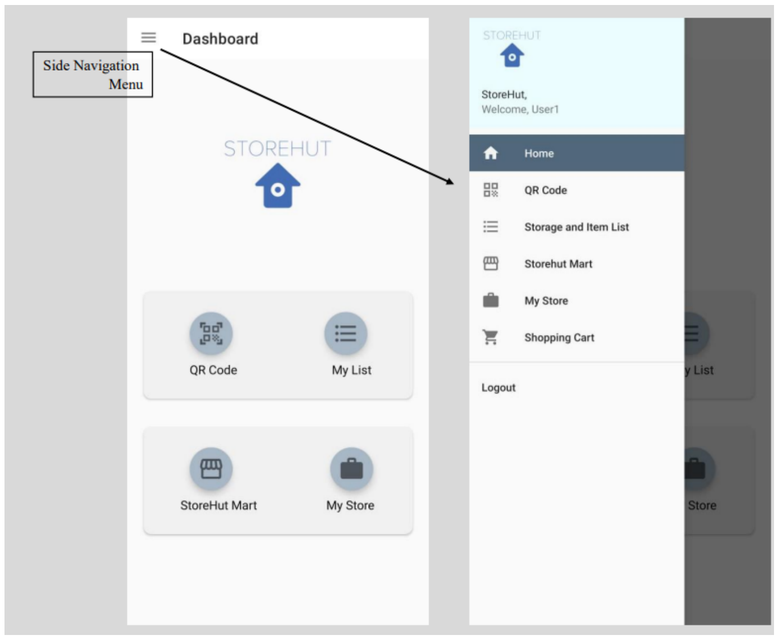


Figure 4.4.1.1 UI of Dashboard

4.4.2 QR Code Generator

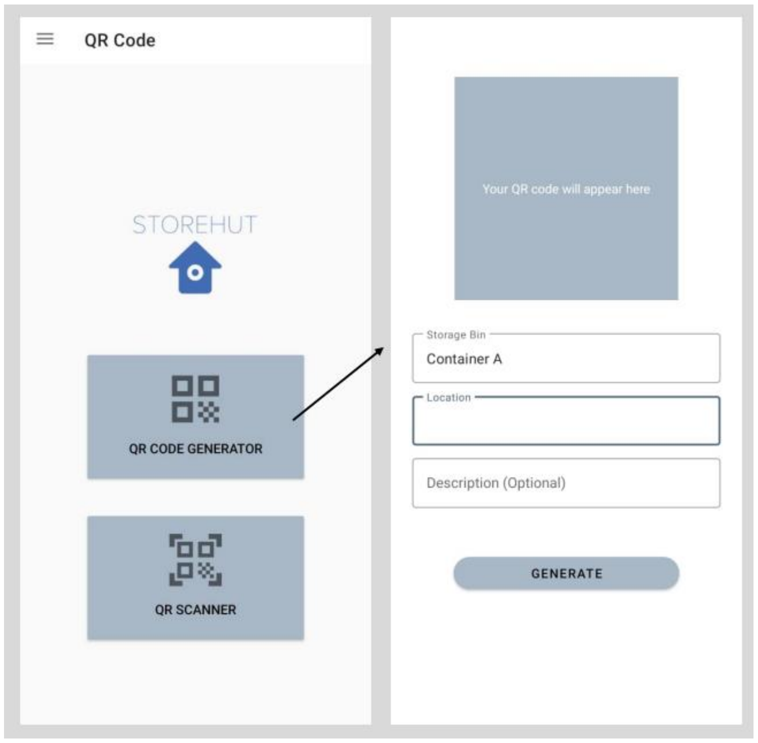


Figure 4.4.2.1 UI of QR Code Generator

4.4.3 QR Code Scanner

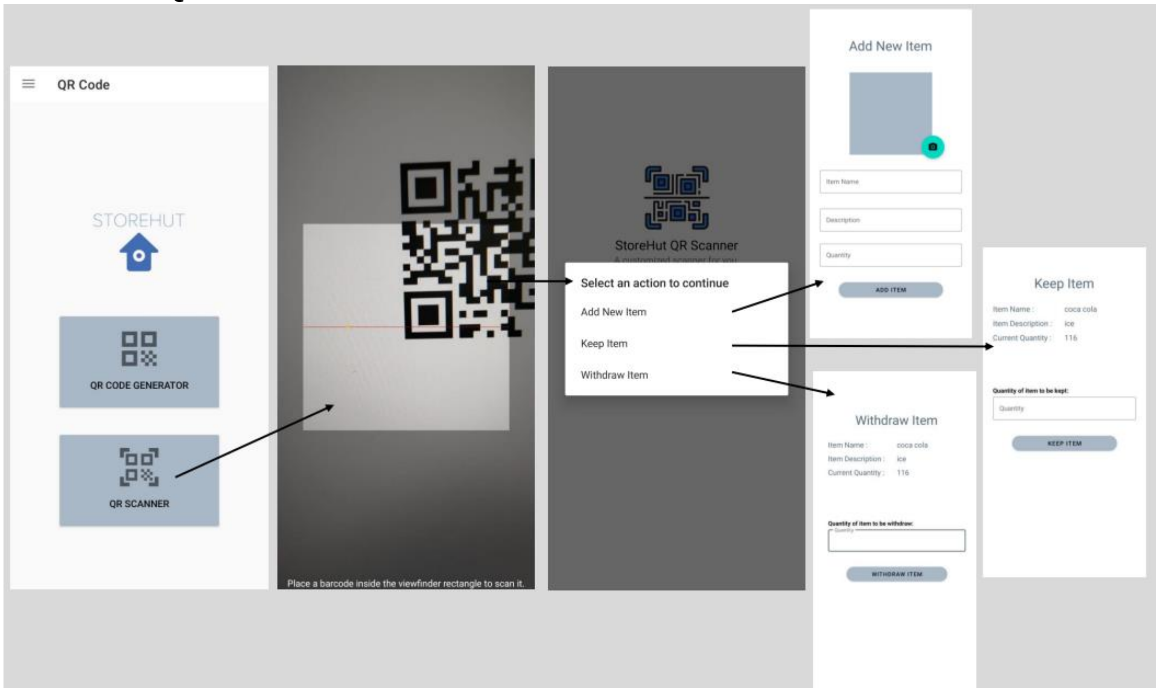


Figure 4.4.3.1 UI of QR Code Scanner

4.4.4 Item and Storage Location List

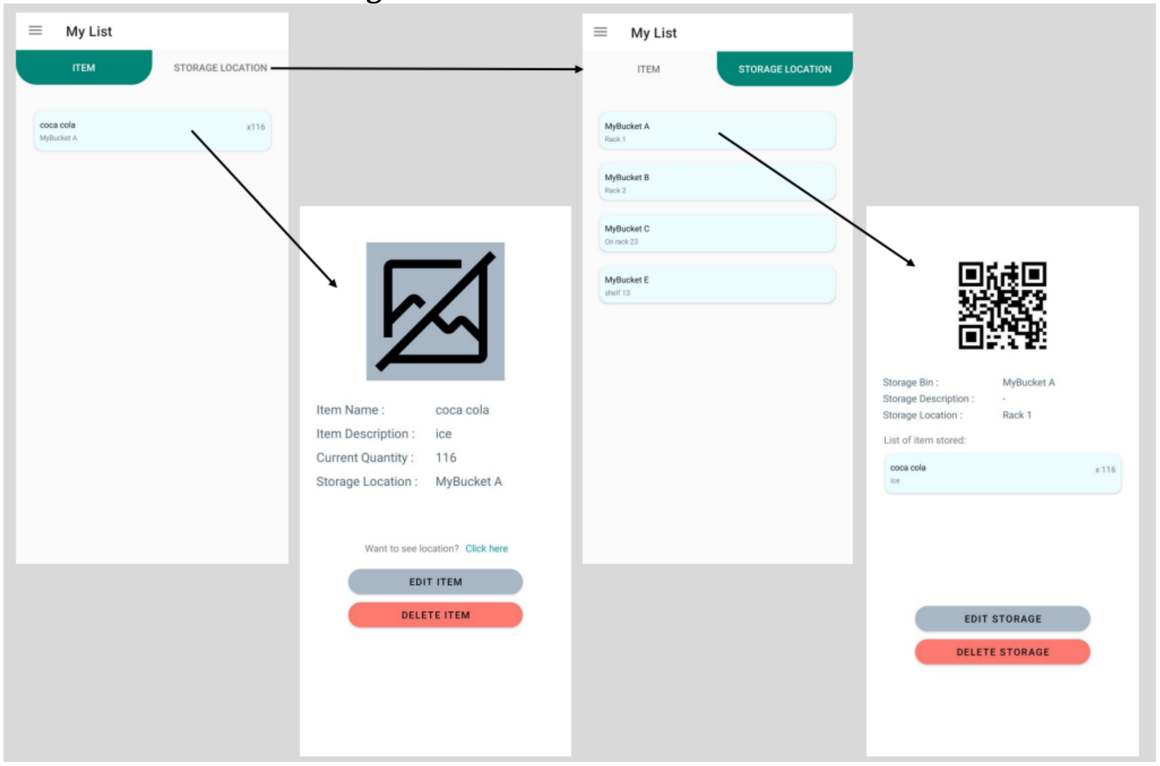


Figure 4.4.4.1 UI of Item and Storage Location List

4.4.5 StoreHut Mart Interface

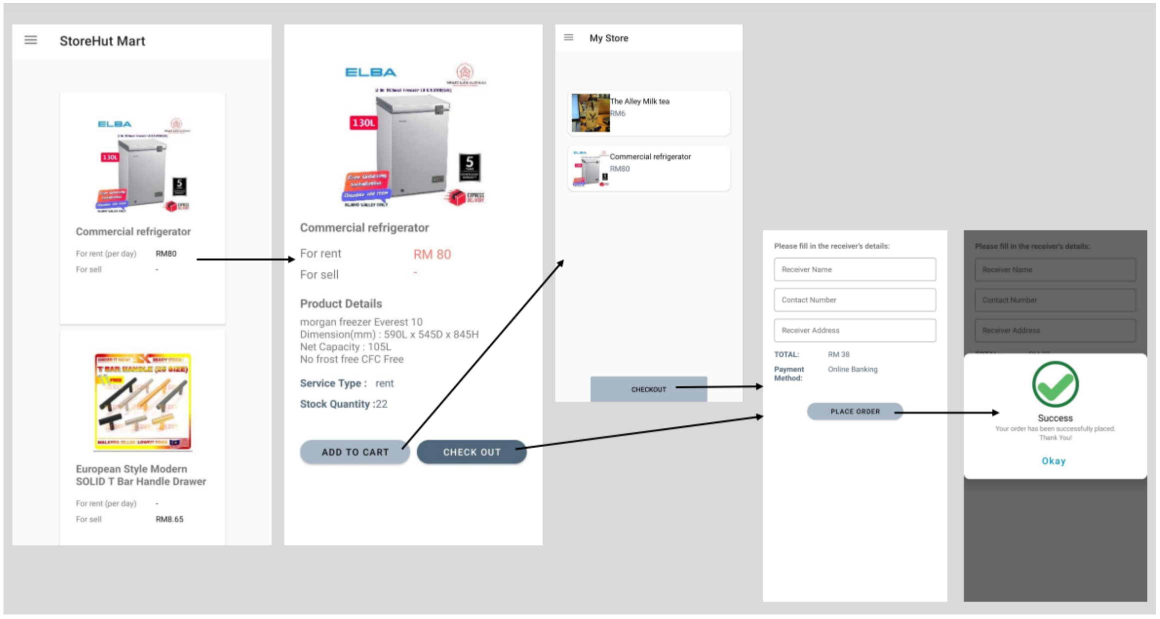
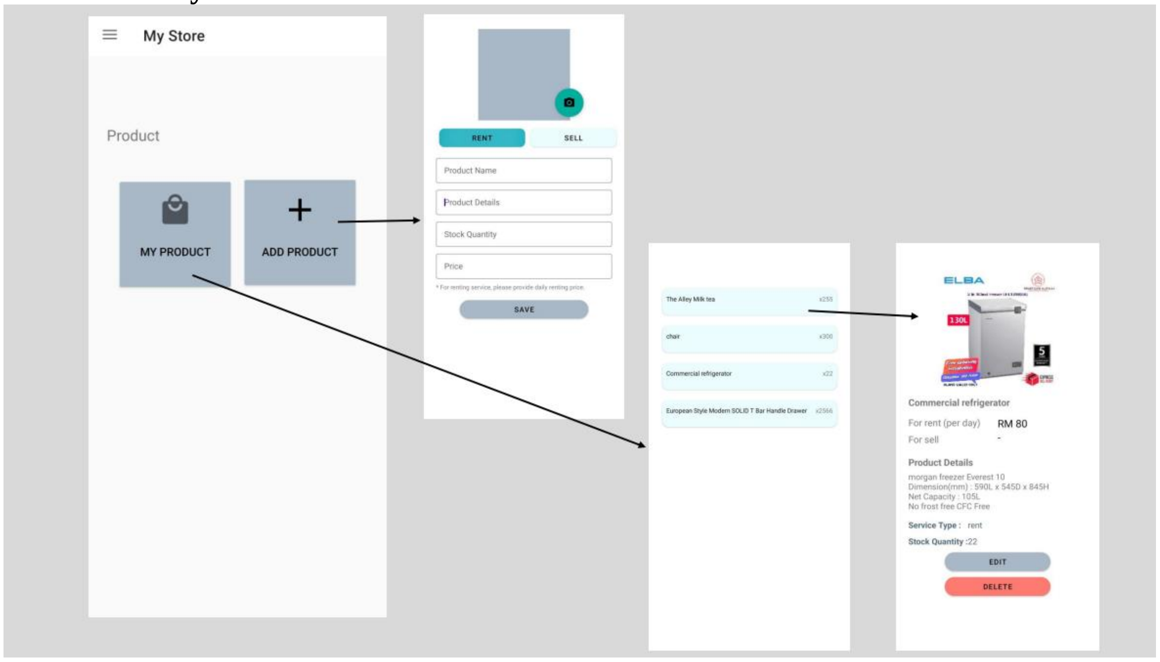


Figure 4.4.5.1 UI of StoreHut Mart Interface

4.4.6 MyStore Interface



The Alley Milk tea x255
chair x300
Commercial refrigerator x22
European Style Modern SOLID T Bar Handle Drawer x2566

Commercial refrigerator
For rent (per day) RM 80
For sell -
Product Details
morgan freezer Everest 10
Dimension(mm) : 590L x 545D x 845H
Net Capacity : 105L
No frost free CFC Free
Service Type : rent
Stock Quantity :22

Figure 4.4.6.1 UI of MyStore Interface

CHAPTER 5 SYSTEM EVALUATION AND DISCUSSION

5.1 Software Testing

After the application is completely developed, we have performed software testing which includes component test, integration test, system test, and acceptance test. In component testing, each software component is examined in isolation, such as code, data model and component specifications. This is to find out the incorrect functionality and data flow, as well as verifying the code and logic. After every component is examined, we start to perform integration testing, where the interface and interaction between components are tested. This is to verify whether the functional and non-functional behaviours of interfaces are as designed and specified. In this case, we had performed Top Down Depth First Integration Test.

Next, in the system test level, we try to find bugs in the overall functions and responses of the system under test as a whole. The techniques that we use is use case test, which test the individual use cases and its interactions by following the use cases one by one. The last step is performing acceptance testing. The system has been tested from the perspective of users to validate that the system is complete and work as expected. In the User Acceptance Test (UAT), the generated apk file of this application is distributed to 10 friends and relatives, who met the problem of managing their equipment or stocks, for testing purpose. The feedbacks from each user are collected and analysed to figure out the fitness for use of the system by intended users.

5.2 User Feedback Analysis

In the User Acceptance Test (UAT), the generated apk file of this application is distributed to 10 friends and relatives, who met the problem of managing their equipment or stocks. After 5 days of testing period, a google form is being distributed to each user to collect their feedback regarding the completeness, usability, requirement fulfilment and satisfaction.

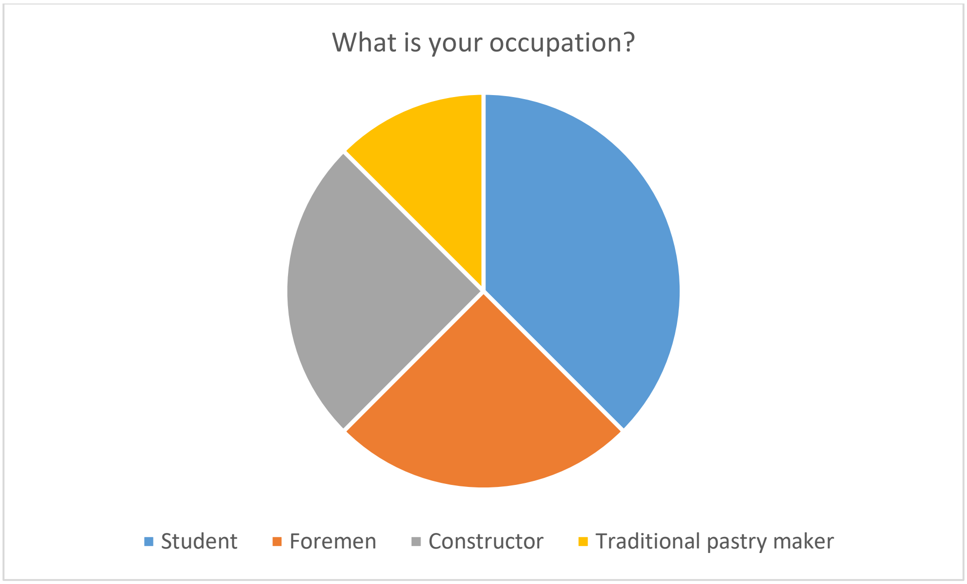
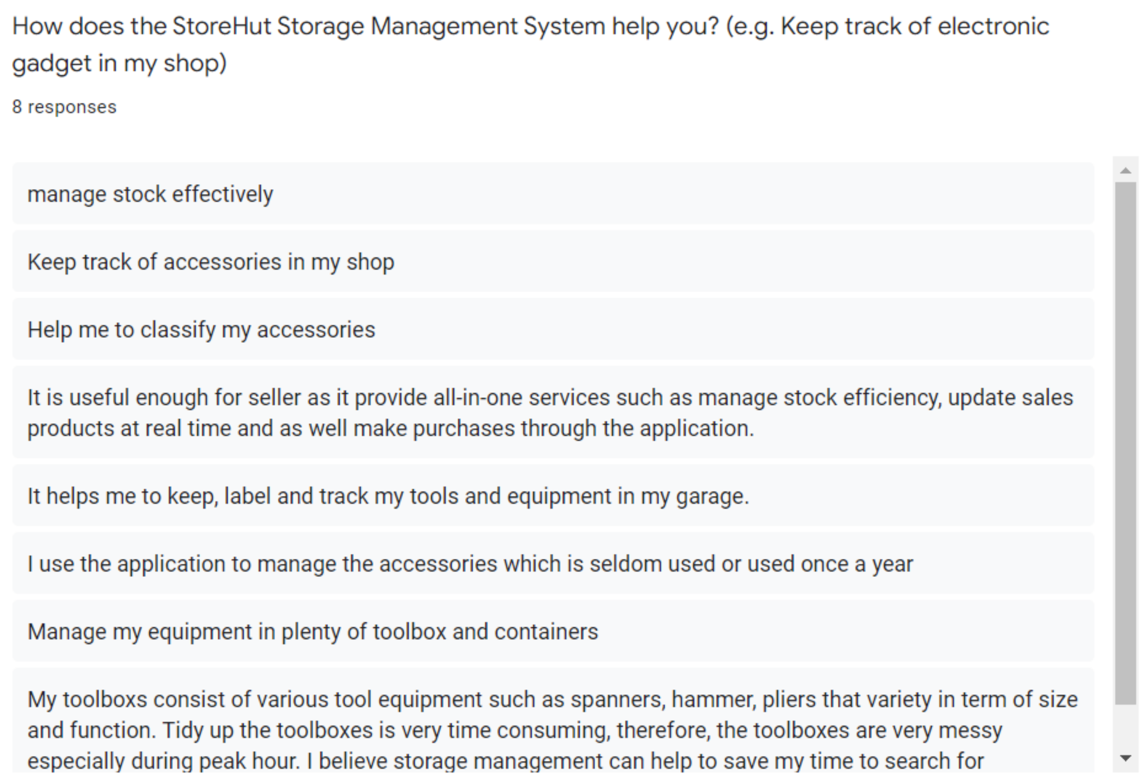


Figure 5.2.1 Respondent's Occupation

By referring to the Pie chart above, we know that our respondents are come from different background, some of them are student, while others are working adults, which are foremen, constructor and traditional pastry maker.



How does the StoreHut Storage Management System help you? (e.g. Keep track of electronic gadget in my shop)

8 responses

manage stock effectively

Keep track of accessories in my shop

Help me to classify my accessories

It is useful enough for seller as it provide all-in-one services such as manage stock efficiency, update sales products at real time and as well make purchases through the application.

It helps me to keep, label and track my tools and equipment in my garage.

I use the application to manage the accessories which is seldom used or used once a year

Manage my equipment in plenty of toolbox and containers

My toolboxs consist of various tool equipment such as spanners, hammer, pliers that variety in term of size and function. Tidy up the toolboxes is very time consuming, therefore, the toolboxes are very messy especially during peak hour. I believe storage management can help to save my time to search for

Figure 5.2.2 The feedback given on how does StoreHut Management System help the respondents

The next question is asking how the respondents implement the StoreHut system in their works or life, and solve the problems they have met. According to the list above, our respondents are using the application to manage, keep track, classify, and labels their tools, accessories, equipment and stocks. One of the respondents claimed that the StoreHut Storage Management System is excellent in providing all-in-one services, from managing the stock, update products details to the process of purchasing. Besides, one of them is manage to save his time from searching the correct tools from the messy toolboxes in the foreman shop.

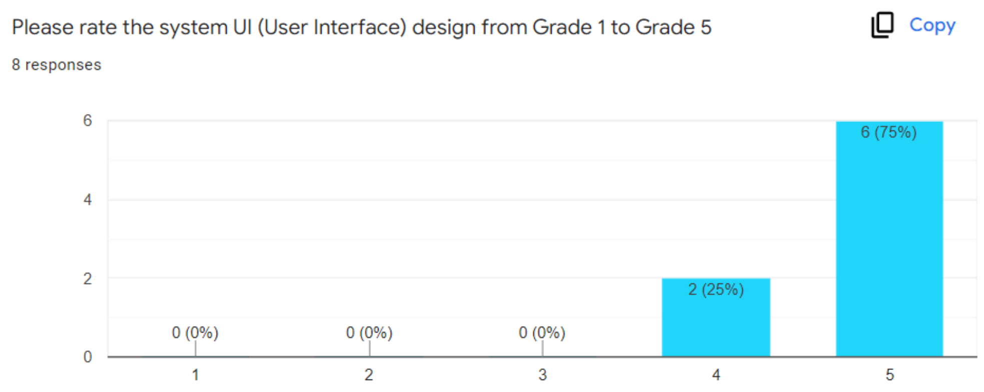


Figure 5.2.3 The result of rating the system UI design.

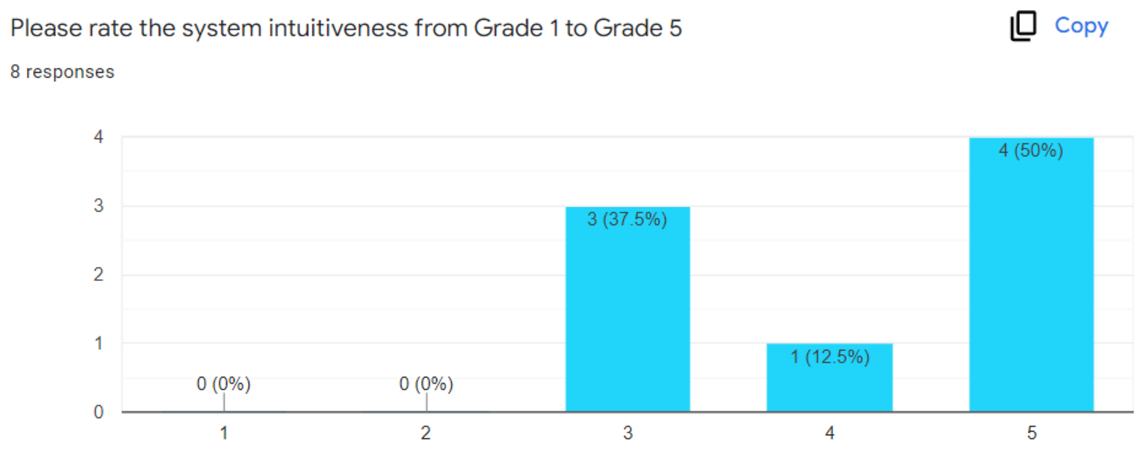


Figure 5.2.4 The result of rating the system intuitiveness.

The third and fourth question are requesting the respondents to rate the system user interface (UI) and intuitiveness from Grade 1 to Grade 5. Grade 1 is indicating that the UI or intuitiveness of the system is not pleasant at all, while Grade 5 is indicating they are very satisfying with that. The reason behind of designing this question is to make sure that the UI design of the application satisfies most people's aesthetics while easily understand how to use the application without much effort. By referring to the Figure 5.2.3 and 5.2.4, most of the respondents are satisfying with the UI design and intuitiveness.

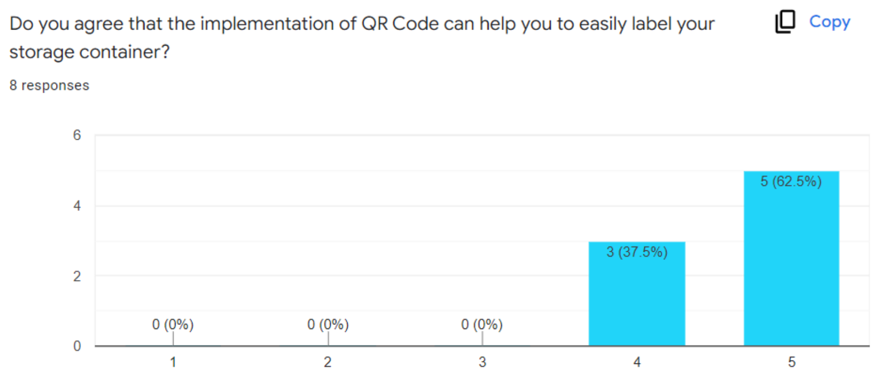


Figure 5.2.5 The result of rating the usefulness of QR Code implementation in the system

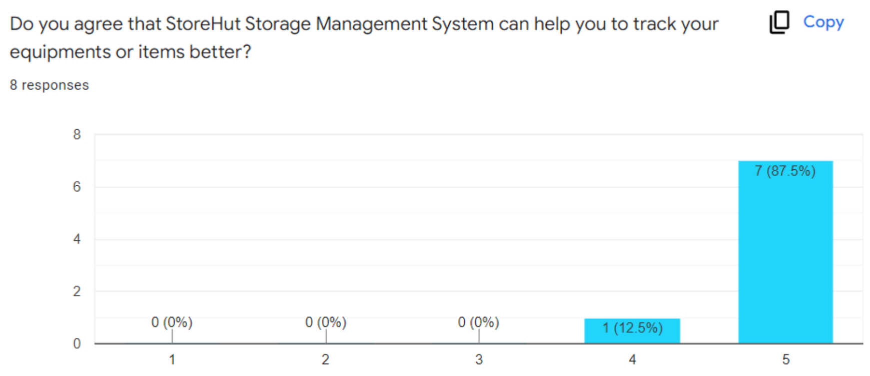


Figure 5.2.6 The result of rating the effectiveness of the system on tracking the equipment or items.

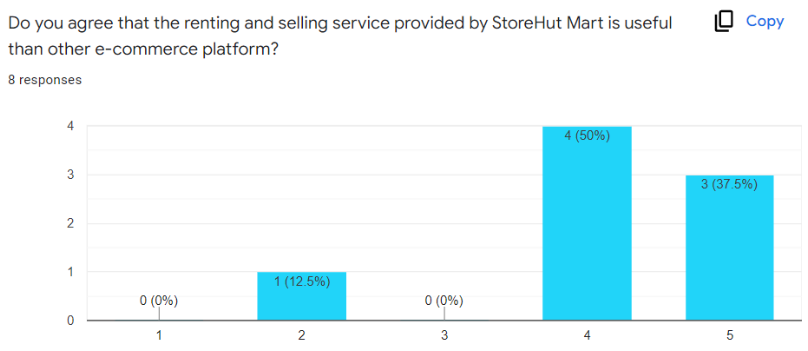
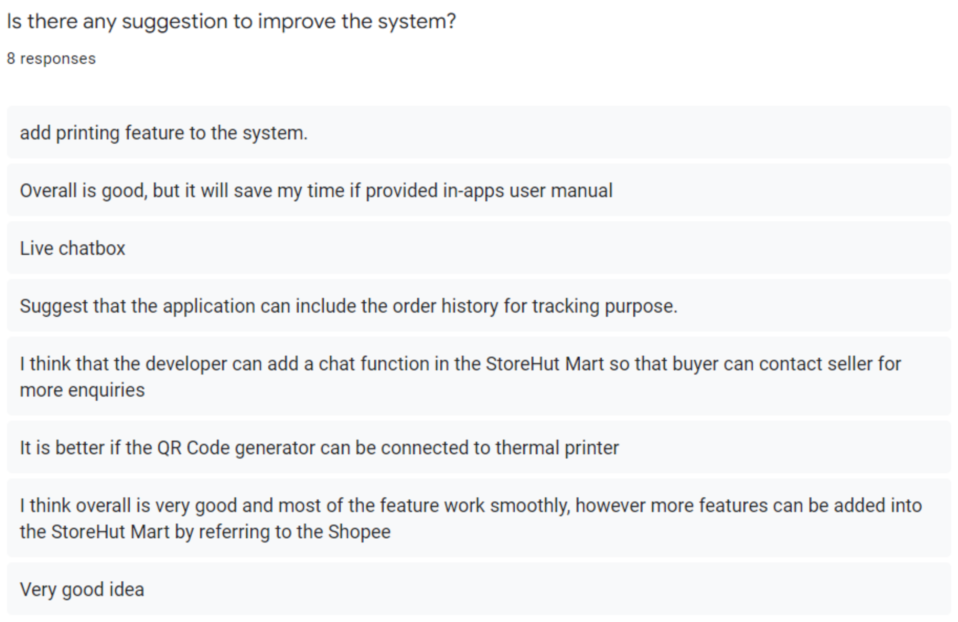


Figure 5.2.7 The result of rating the usefulness of renting and selling service provided.

The next three questions are asking the usefulness and effectiveness of the features in StoreHut Storage Management System. First of all, most of the respondents are enjoying the convenience provided by the application to their life or works. However, 12.5% of the respondents are not very agree with the statement of renting and selling service provided by StoreHut Mart is useful than other e-commerce platform. Hence, we should make improvement on this section of the application to increase user involvement and perfect the functionality.


Is there any suggestion to improve the system?

8 responses

add printing feature to the system.

Overall is good, but it will save my time if provided in-apps user manual

Live chatbox

Suggest that the application can include the order history for tracking purpose.

I think that the developer can add a chat function in the StoreHut Mart so that buyer can contact seller for more enquiries

It is better if the QR Code generator can be connected to thermal printer

I think overall is very good and most of the feature work smoothly, however more features can be added into the StoreHut Mart by referring to the Shopee

Very good idea

Figure 5.2.8 The list of suggestion provided by each of the respondents.

According to the suggestion provided by all of the respondent, we can conclude that, 50% of them 8 respondents hope that we can make adjustment and improvement in the StoreHut Mart features, such as providing live chat service, chat function between buyer and seller, order history for tracking purpose, and others. Besides, two of the respondents claimed that it is better to connect the QR Code generator to printer to improve the user experience. Furthermore, a respondent is giving a useful feedback, which is providing an in-apps user manual for the new users so that they are able to be familiar with the system more efficiently.

CHAPTER 6 CONCLUSION

6.1 Review of Project

In conclusion, the result of development has fulfilled the three objectives of study defined initially. Firstly, we aimed to analyse the existing approaches used in tracking object location in random chaotic storage system at makerspace. We were able to accomplish this goal by analysing the strengths and weaknesses of various technologies and researches based on our findings from the literature review. Additionally, the second objective which is to design a mobile application that helps tracking object location in random chaotic storage system, is achieved by implementing the technology of QR Code Scanning to track the storage location of every single items. Furthermore, the system provides a clear view of the items stored in each storage site and description of each item's storage location, as well as the stock available for each item. Next, the objective of designing a mobile application that helps user to purchase or rent equipment and ingredients is accomplish by creating an e-commerce platform in the application which allow user to sell, rent, and buy equipment, ingredients, machine or other products. All in all, the system testing is attaining the last objective of evaluating the usability of the mobile application in tracking object location in random chaotic storage system at makerspace while renting and buying products from other users.

6.2 Novelties and Contributions

We stood out from the previous makerspace management system due to the development and combining of several modules with a simple but effective approach of managing the makerspace storage. First and foremost, the use of QR Code technology, which required fewer resources, allowed for the efficient tracking of each item's storage position as well as the display of the things stored in each storage place on the screen. Users may soon understand about all of the storage in the makerspace, including the stock available for each item, with only a few taps. Aside from that, the development of e-commerce platform in the makerspace management system is another highlight of our project. It provides renting and selling service for the user to share and sell their tools and equipment among the community.

6.3 Limitation

The function of Storehut Mart can be alternate by the existing e-commerce provider (i.e. Shopee, Lazada, Zalora as the competitors). As comparable, there is limited competitive advantage and low product diversification in the early stage of Storehut Mart develpopment. This is because of existence of strong customer base in the market, well-established distribution channel, and operational efficiency of the competitors. Hence, some of the equipment or ingredients can be found at a cheaper price, unfavorable price quoted in Storehut Mart leads to lower liquidity, where buyers and sellers are not willing to participate in Storehut Mart.

6.4 Future Work

There were numerous enhancements and improvements that may be made or attained for future work throughout this project. In-app user manual could be added into the mobile application to improve the intuitiveness of the system and increase user experience. Next, chatbox can be included into the StoreHut Mart to build communication between seller and buyer while reducing the problems of misunderstanding of products. Besides, future applications may include more useful features that will aid the maker in their works. For example, a recommendation of nearby physical stores based on the user's search terms can provide the user with more options when it comes to selecting equipment, ingredients, and other resources. It not only assists users in locating wanted things, but it also assists sellers in increasing revenue while promoting their businesses.

REFERENCES

[1]C. Roser, "Storage Strategies – Fixed Location | AllAboutLean.com", *AllAboutLean.com*, 2022. [Online]. Available: https://www.allaboutlean.com/storage-strategies-fixed-location. [Accessed: 13- Apr- 2022].

[2]C. Shi, "The Construction of a Consumer to Consumer E-commerce System Based on Japanese Elderly Community", *2012 International Conference on Computer Science and Electronics Engineering*, pp. 253-258, 2012. Available: 10.1109/ICCSEE.2012.414.

[3]"Carousell Malaysia - Online Marketplace to Buy and Sell Items for Free Anywhere in Malaysia", *Carousell*, 2022. [Online]. Available: https://www.carousell.com.my. [Accessed: 13- Apr- 2022].

[4]"CashierLive | Affordable, easy to use web based point-of-sale software.", *CashierLive*, 2022. [Online]. Available: https://www.cashierlive.com/inventory. [Accessed: 13- Apr- 2022].

[5]G. Zhang, X. Shang, F. Alawneh, Y. Yang and T. Nishi, "Integrated production planning and warehouse storage assignment problem: An IoT assisted case", *International Journal of Production Economics*, vol. 234, p. 108058, 2021. Available: 10.1016/j.ijpe.2021.108058.

[6]"List of Hacker Spaces - HackerspaceWiki", *Wiki.hackerspaces.org*, 2020. [Online]. Available: https://wiki.hackerspaces.org/List_of_Hacker_Spaces. [Accessed: 02- Aug- 2020].

[7]M. van Geest, B. Tekinerdogan and C. Catal, "Design of a reference architecture for developing smart warehouses in industry 4.0", *Computers in Industry*, vol. 124, p. 103343, 2021. Available: 10.1016/j.compind.2020.103343.

[8]"Makerspace NFC Part Management System", *Hackster.io*, 2022. [Online]. Available: https://www.hackster.io/gatoninja236/makerspace-nfc-part-management-system-e4f5f3. [Accessed: 13- Apr- 2022].

[9]P. Espinoza-Camino, I. Macassi-Jaurequi, C. Raymundo-Ibañez and F. Dominguez, "Warehouse management model using FEFO, 5s, and chaotic storage to improve product loading times in small- and medium-sized non-metallic mining companies", *IOP Conference Series: Materials Science and Engineering*, vol. 796, no. 1, p. 012012, 2020. Available: 10.1088/1757-899x/796/1/012012.

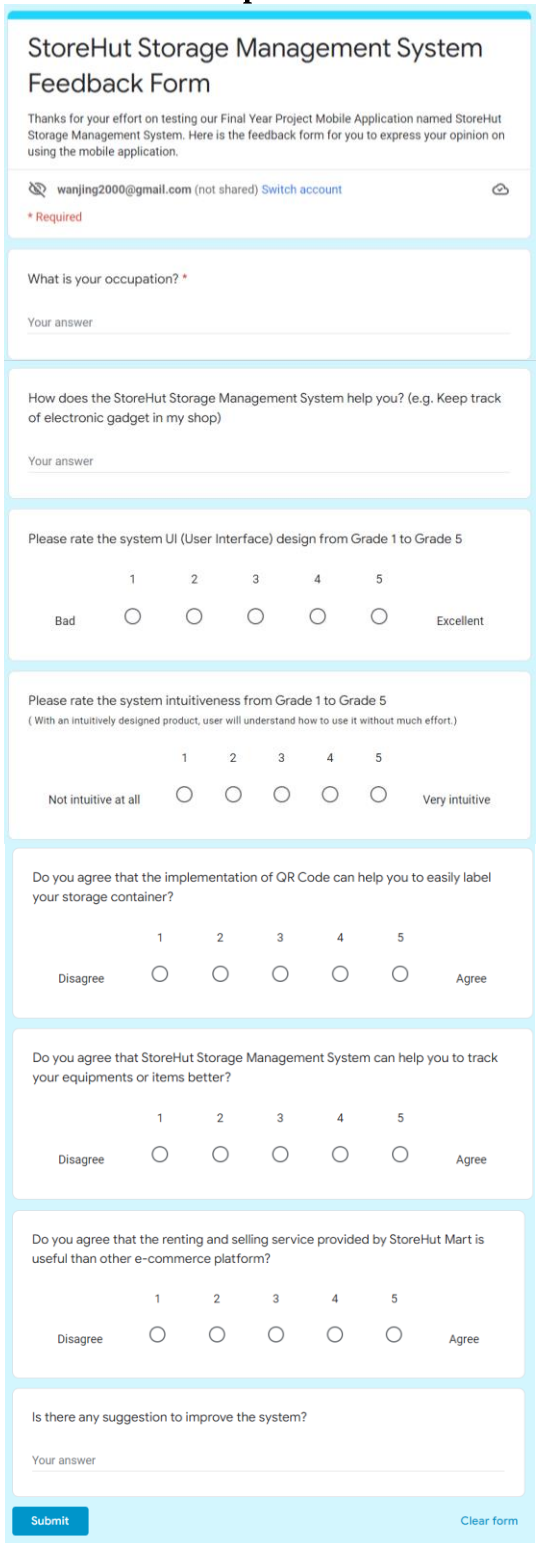
[10]*Shopee MY*. Singapore: Shopee, 2022.

[11]"Sortly: Simple Inventory Management Software for Small Businesses", *Sortly*, 2022. [Online]. Available: https://www.sortly.com. [Accessed: 13- Apr- 2022].

[12]Z. Zadorozhnyi, V. Muravskyi, N. Pochynok and A. Hrytsyshyn, "Innovation Management and Automated Accounting in the Chaotic Storage Logistics", *Marketing and Management of Innovations*, no. 2, pp. 313-323, 2020. Available: 10.21272/mmi.2020.2-23. Li, F., 2022

APPENDIX

Appendix A : User Feedback Form Template


StoreHut Storage Management System Feedback Form

Thanks for your effort on testing our Final Year Project Mobile Application named StoreHut Storage Management System. Here is the feedback form for you to express your opinion on using the mobile application.

wanjing2000@gmail.com (not shared) Switch account

* Required

What is your occupation? *

Your answer

How does the StoreHut Storage Management System help you? (e.g. Keep track of electronic gadget in my shop)

Your answer

Please rate the system UI (User Interface) design from Grade 1 to Grade 5

	1	2	3	4	5	
Bad	○	○	○	○	○	Excellent

Please rate the system intuitiveness from Grade 1 to Grade 5
(With an intuitively designed product, user will understand how to use it without much effort.)

	1	2	3	4	5	
Not intuitive at all	○	○	○	○	○	Very intuitive

Do you agree that the implementation of QR Code can help you to easily label your storage container?

	1	2	3	4	5	
Disagree	○	○	○	○	○	Agree

Do you agree that StoreHut Storage Management System can help you to track your equipments or items better?

	1	2	3	4	5	
Disagree	○	○	○	○	○	Agree

Do you agree that the renting and selling service provided by StoreHut Mart is useful than other e-commerce platform?

	1	2	3	4	5	
Disagree	○	○	○	○	○	Agree

Is there any suggestion to improve the system?

Your answer

Figure A-1 User feedback form

Appendix B : Final Year Project Weekly Report
(Project II)

Trimester, Year: 3,3	**Study week no.:2**
Student Name & ID: Tan Wan Jing 18ACB01284	
Supervisor: Ts Dr Ooi Boon Yaik	
Project Title: Development Of Maker Space Storage Management System	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

Application development:

1) User authentication (implementing Firebase Authentication)
2) Side menu & homepage

2. WORK TO BE DONE

Application development:

1) QR Code feature – QR Code generator and scanner
2) Listing of items and storage location

3. PROBLEMS ENCOUNTERED

-

4. SELF EVALUATION OF THE PROGRESS

- Efficient and performs time management properly

[illegible] [illegible]

Supervisor's signature Student's signature

Weekly Report

Trimester, Year: 3,3	**Study week no.:4**
Student Name & ID: Tan Wan Jing 18ACB01284	
Supervisor: Ts Dr Ooi Boon Yaik	
Project Title: Development Of Maker Space Storage Management System	

1. WORK DONE [Please write the details of the work done in the last fortnight.] Application development: 1) QR Code feature – QR Code generator and scanner 2) Listing of items and storage location
2. WORK TO BE DONE Application development: 3) My Mart feature – Add new product, edit and delete existing product
3. PROBLEMS ENCOUNTERED - Not able to follow the Wireframe created in FYP 1 because have better idea on developing this feature
4. SELF EVALUATION OF THE PROGRESS - Efficient and performs time management properly

Alex

Supervisor's signature

[illegible]

Student's signature

Weekly Report

Trimester, Year: 3,3	**Study week no.:6**
Student Name & ID: Tan Wan Jing 18ACB01284	
Supervisor: Ts Dr Ooi Boon Yaik	
Project Title: Development Of Maker Space Storage Management System	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

Application development:

1) My Mart feature – Add new product, edit and delete existing product
2) StoreHut Mart feature – Create a grid view that lists the products and let user to add them into shopping cart.
3) Shopping Cart feature

2. WORK TO BE DONE

- Perform system testing
- Refine FYP 2 report (Chapter 1, 2, and 3)

3. PROBLEMS ENCOUNTERED

-

4. SELF EVALUATION OF THE PROGRESS

- Efficient and performs time management properly

Alex

Supervisor's signature

[illegible]

Student's signature

Weekly Report

Trimester, Year: 3,3	**Study week no.:8**
Student Name & ID: Tan Wan Jing 18ACB01284	
Supervisor: Ts Dr Ooi Boon Yaik	
Project Title: Development Of Maker Space Storage Management System	

1. WORK DONE [Please write the details of the work done in the last fortnight.] - Software testing is performed and done. - Chapter 1, 2 and 3 is completed
2. WORK TO BE DONE - Do documentation on System Implementation and Evaluation
3. PROBLEMS ENCOUNTERED - When performing software testing, quite a number of feature are not performed as expected, so much more time is used to solve the errors and bugs, and test few more time.
4. SELF EVALUATION OF THE PROGRESS - Efficient and performs time management properly

Alex [illegible]

Supervisor's signature Student's signature

Weekly Report

Trimester, Year: 3,3	**Study week no.:10**
Student Name & ID: Tan Wan Jing 18ACB01284	
Supervisor: Ts Dr Ooi Boon Yaik	
Project Title: Development Of Maker Space Storage Management System	

1. WORK DONE

[Please write the details of the work done in the last fortnight.]

- The report is completed.

2. WORK TO BE DONE

- Do turnitin checking
- Design poster
- Double check on the report content and formatting
- Prepare presentation slides and demo video

3. PROBLEMS ENCOUNTERED

-

4. SELF EVALUATION OF THE PROGRESS

- Efficient and performs time management properly

Alex

Supervisor's signature

[illegible]

Student's signature

Appendix C : Poster

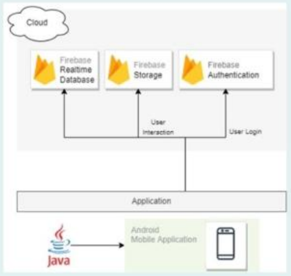
Makerspace Storage Management System

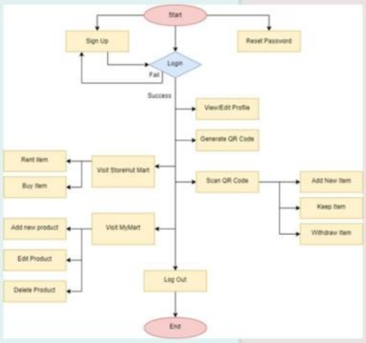
Introduction

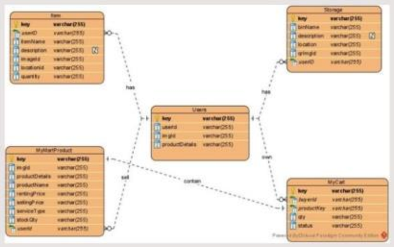
It is a mobile application that is specifically designed for makers to manage and track the item and its storage location from time to time. The storage approach used is Random Chaotic Storage, where the objects are placed randomly and disorderedly in a storage location.

Features:
- QR Code generator
- QR Code scanner
- StoreHut Mart
- MyStore

Methodology

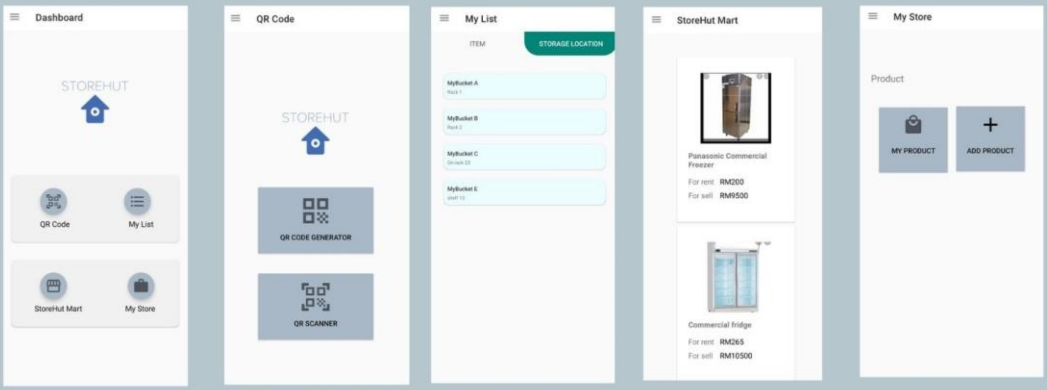






Result

STOREHUT MOBILE APPLICATION



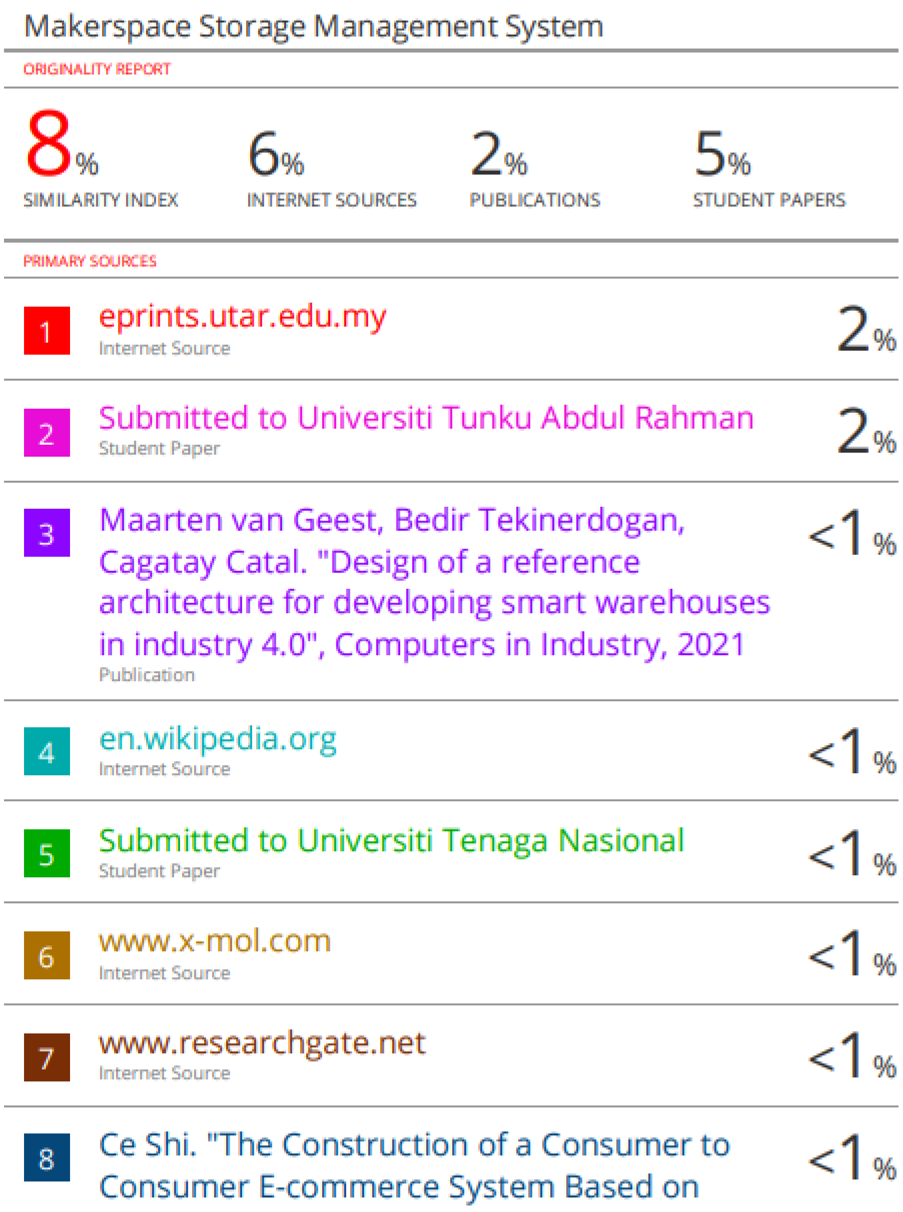
Conclusion

In conclusion, the result of development has fulfilled the three objectives of the study defined initially. However, As comparable, there is a limited competitive advantage and low product diversification in the early stage of Storehut Mart development. Throughout this project, there were plenty of enhancements and improvements that could be made or achieved for future work. For example, in-app user manual, chatbox, recommendation of nearby physical stores

BACHELOR OF COMPUTER SCIENCE (HONS)
Faculty of Information and Communication Technology
FINAL YEAR PROJECT

Done By: Tan Wan Jing

Appendix D: Plagiarism Check Result



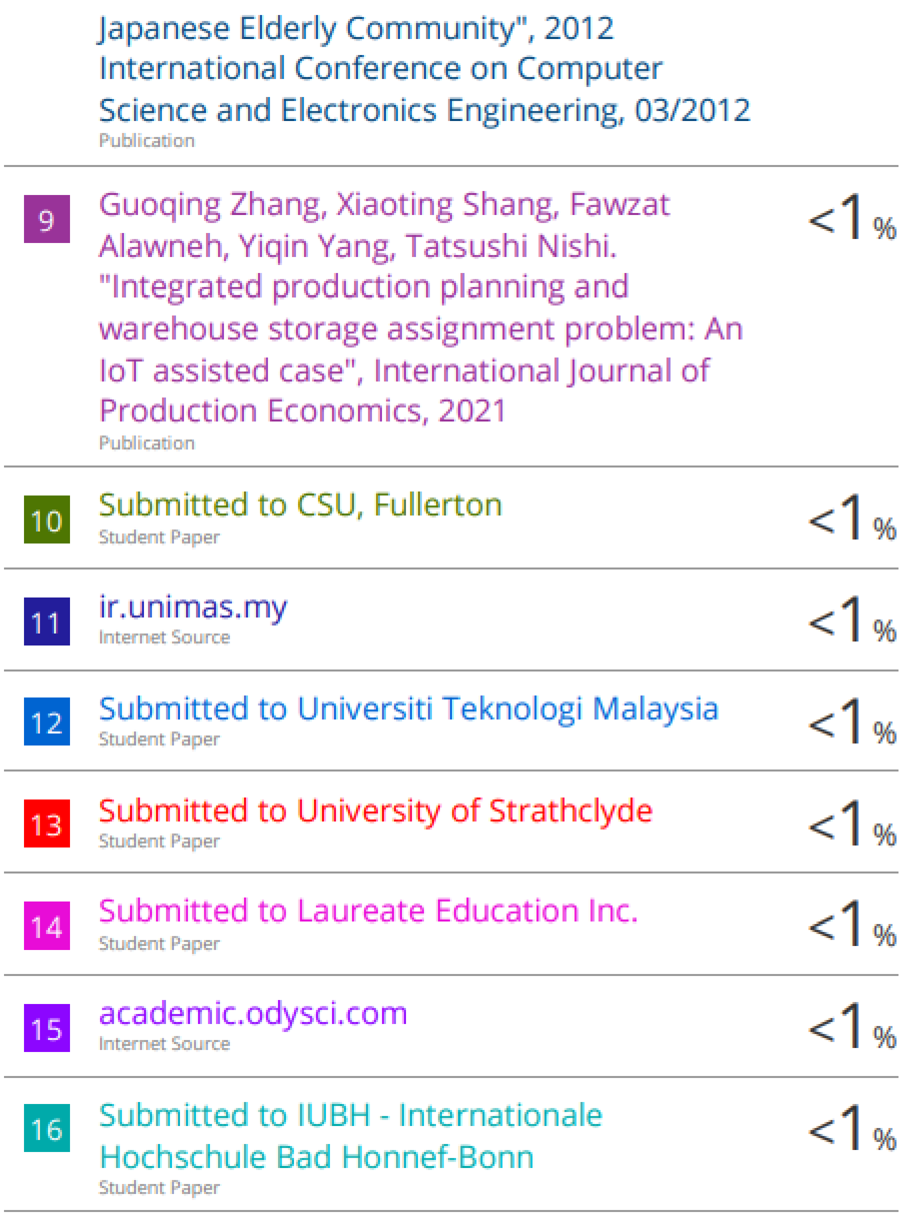
Makerspace Storage Management System

ORIGINALITY REPORT

8%	6%	2%	5%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

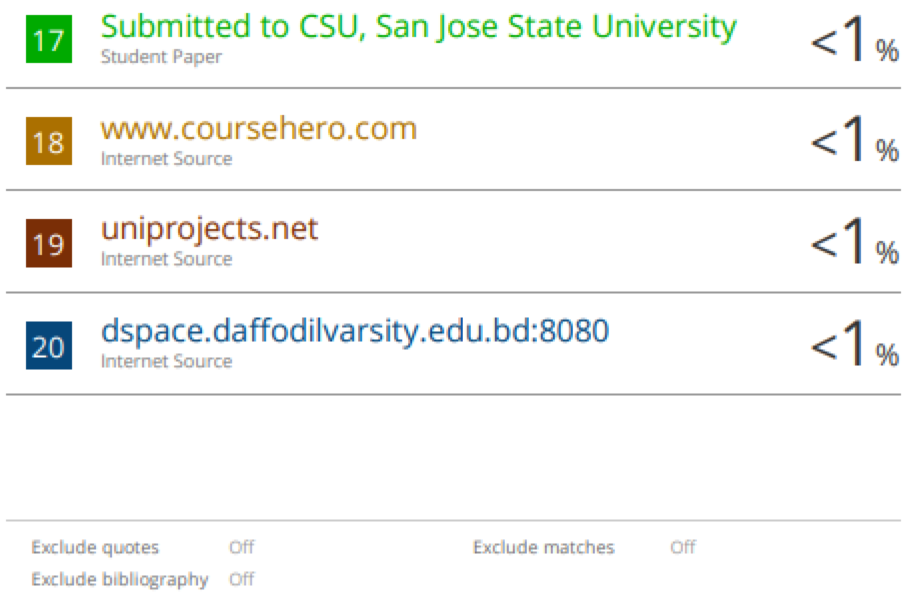
PRIMARY SOURCES

1	eprints.utar.edu.my Internet Source	2%
2	Submitted to Universiti Tunku Abdul Rahman Student Paper	2%
3	Maarten van Geest, Bedir Tekinerdogan, Cagatay Catal. "Design of a reference architecture for developing smart warehouses in industry 4.0", Computers in Industry, 2021 Publication	<1%
4	en.wikipedia.org Internet Source	<1%
5	Submitted to Universiti Tenaga Nasional Student Paper	<1%
6	www.x-mol.com Internet Source	<1%
7	www.researchgate.net Internet Source	<1%
8	Ce Shi. "The Construction of a Consumer to Consumer E-commerce System Based on	<1%



	Japanese Elderly Community", 2012 International Conference on Computer Science and Electronics Engineering, 03/2012 Publication	
9	Guoqing Zhang, Xiaoting Shang, Fawzat Alawneh, Yiqin Yang, Tatsushi Nishi. "Integrated production planning and warehouse storage assignment problem: An IoT assisted case", International Journal of Production Economics, 2021 Publication	<1 %
10	Submitted to CSU, Fullerton Student Paper	<1 %
11	ir.unimas.my Internet Source	<1 %
12	Submitted to Universiti Teknologi Malaysia Student Paper	<1 %
13	Submitted to University of Strathclyde Student Paper	<1 %
14	Submitted to Laureate Education Inc. Student Paper	<1 %
15	academic.odysci.com Internet Source	<1 %
16	Submitted to IUBH - Internationale Hochschule Bad Honnef-Bonn Student Paper	<1 %

Plagiarism Check Result



17	Submitted to CSU, San Jose State University Student Paper	<1 %
18	www.coursehero.com Internet Source	<1 %
19	uniprojects.net Internet Source	<1 %
20	dspace.daffodilvarsity.edu.bd:8080 Internet Source	<1 %

Exclude quotes	Off	Exclude matches	Off
Exclude bibliography	Off		

Plagiarism Check Result

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)	Tan Wan Jing
ID Number(s)	18ACB01284
Programme / Course	Bachelor Of Computer Science (Honours)
Title of Final Year Project	Makerspace Storage Management

Similarity	**Supervisor's Comments (Compulsory if parameters of originality exceeds the limits approved by UTAR)**
Overall similarity index: 8 **%** **Similarity by source** Internet Sources: 6 % Publications: 2 % Student Papers: 5 %	
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: Ts Dr Ooi Boon Yaik	Name:
Date: 21/4/2022	Date:

Appendix E: FYP 2 Checklist

UNIVERSITI TUNKU ABDUL RAHMAN

FACULTY OF INFORMATION & COMMUNICATION TECHNOLOGY (KAMPAR CAMPUS)

CHECKLIST FOR FYP2 THESIS SUBMISSION

Student Id	18ACB01284
Student Name	Tan Wan Jing
Supervisor Name	Ts Dr Ooi Boon Yaik

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.
	Front Plastic Cover (for hardcopy)
√	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.

[illegible]

(Signature of Student)
Date: 21/04/2022