SMART MOBILE APPLICATION FOR RESUME BUILDING AND CAREER ADVISORY SERVICES

LEONG MING SHAN

UNIVERSITI TUNKU ABDUL RAHMAN

SMART MOBILE APPLICATION FOR RESUME BUILDING AND CAREER ADVISORY SERVICES

LEONG MING SHAN

A project report submitted in partial fulfilment of the requirements for the award of Bachelor of Software Engineering (Honours)

Lee Kong Chian Faculty of Engineering and Science
Universiti Tunku Abdul Rahman

September 2025

DECLARATION

I hereby declare that this project report is based on my original work except for citations and quotations which have been duly acknowledged. I also declare that it has not been previously and concurrently submitted for any other degree or award at UTAR or other institutions.

Name : Leong Ming Shan

ID No. : 2105781

Date : 18 September 2025

COPYRIGHT STATEMENT

This final year project report is submitted in partial fulfilment of the requirements for the Bachelor of Software Engineering (Honours). This final year project report represents the work of the author, except where due acknowledgement has been made in the text. No part of this final year project report may be reproduced, stored, or transmitted in any form or by any means, whether electronic, mechanical, photocopying, recording, or otherwise, without the prior written permission of the author or UTAR, in accordance with UTAR's Intellectual Property Policy.

ACKNOWLEDGEMENTS

I would like to express my gratitude to Assistant Professor, Dr Nawaf Hassan Mohammed Mohsen Shrifan as my research supervisor for his invaluable advice, recommendations and enormous patience throughout the development of this project. He had support and provided comments to me which allows me to perform well in this project.

Moreover, I would like to convey my appreciation to the Department of Computing (DC) for the access permissions, resources, information and facilities provided. These allowed me to have a smooth experience for my academic study and support me on the completion on this development project.

Next, I would also like to thank wholeheartedly to my university peers for their supportive and helpfulness to this project. Their time and effort from them that contributed to this project represents an important level which enhance this research to meet the requirements and scope by executing UAT.

Lastly, I would like to extend my gratitude to my loving family members and friends who support me and express their kindness and patience to me throughout my academic life although I might disturb them sometimes.

ABSTRACT

The increasing complexity for the current modern job market trend has clearly stated that the importance of professional resumes and career advisory services for job seekers no matter students, graduates, and professionals. By reviewing the current mobile resume building applications, it shows that these applications often lack AI functionality, integration of career advisory features, and sufficient customization options which limits user effectiveness. To address these issues, this project aims to develop a smart Android mobile application, Job Assistant, with integrates Google Gemini AI model for main features. The project follows Scrum methodology framework by conducting the requirement analysis, project designs, development, testing and deployment across five sprints. The developed application used Android Studio, React Native, Visual Studio Code, SQLite database, and integrated Gemini AI by using API key for supporting the professional resume creation and personalized career advice recommendations. The developed key features include resume creation with AI-powered, AI generation career advisory services and various customizable options for resumes. Testing with unit test and UAT gained an overall good result with overall unit test cases passed and UAT 4.82 over 5 rating. These findings highlight that integration of AI into resume-building tools not only increases efficiency but also strengthens user job opportunities with supporting career decisions. To conclude, this project, the smart mobile application with resume building and career advisory services which called Job Assistant represents a valuable community initiative software solution that supports equal access to career development resources and align with Sustainable Development Goals (SDG) such as decent work and no poverty. This Android mobile application can be further expanded with features like multi-languages support, integration with job platforms, iOS version release, offline AI functionality and others to maximize and improve this application.

Keywords: Resume Builder, Resumes, Career Advisory Services, Android Mobile Application, Software Development, React Native, AI, Job Opportunity.

Subject Area: QA76

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF APPENDICES

CHAPTER 1

INTRODUCTION

1.1 Background

A resume is a document that contains personal information, academic credentials, skills, and work experience. It is crucial to use in job seeking and other situations. Therefore, there are many platforms for resume builders on the market nowadays. The main purpose of these platforms is to help users build professional resumes smoothly with a range of features like templates and design tools that ensure resumes are polished, customized, and effective. As a matter of course, career advisory services are essential and needed for all workers to navigate the complexities of the modern job market, especially after completing their resumes and starting on job search. These services provide essential support such as identifying career paths, enhancing skill sets, and effectively positioning oneself for opportunities.

1.2 Importance of the Study

The motivation for developing this resume builder mobile application provide support and help the public, as this study is a community initiative. This study benefits the community by allowing professionals, students, and job seekers to enhance their employment opportunities and career growth. Regardless of their background or other circumstances, users can easily create professional resumes by having equal access to resume-building resources and career guidance to support users in their career development and job search.

This is also related to the Sustainable Development Goals (SDGs), which support decent work and economic growth, and no poverty. Decent work and economic growth is goal 8, which promotes sustained, inclusive and sustainable economic growth, full and productive employment and decent work for all, while no poverty is goal 1, which aims to end poverty in all its forms everywhere. Proper resume builder and suitable career advisory services allow job seekers to have higher opportunities to have a job and reduce the chance of unemployment, which may lead to no poverty due to people having a job and

earning money from their work to support their life, rather than giving alms to them.

Lastly, the capabilities and benefits of using a resume-building mobile application serve as key motivations and highlight the importance of developing such a system in our fast-paced world, where the demand for resume builders is increasing nowadays. Resume builder provides polished, professional content with suggested inputs and allows users to easily modify content and format with customizable options (Microsoft, no date). This allows users to focus on highlighting qualifications, career achievements, and enhancing their job-seeking experience. According to Shivhare et al. (2024), an advanced resume-building system is needed to address an increasingly complex environment and help job searchers match with suitable opportunities.

1.3 Problem Statement

There are several issues with existing resume builder applications that have inspired me to develop a smart mobile app for resume building and career advisory services. These issues restrict and limit the user's options when creating a resume and affect their experience when using the existing applications.

1.3.1 Does not integrate AI functionality in resume builders

AI technologies are the current trend in most sectors and have become indispensable nowadays. It can improve the automated creation of a resume. Without using AI in resume builders, users who use the manual resume design method end up with verbose and uninspired resumes that require more effort (Wu, I. et al., 2017). It is time-consuming and prone to errors for users in resume creation. However, the presence of spelling errors has the same negative impact as a lack of professional experience on the opportunity of being shortlisted (Martin-Lacroux & Lacroux, 2016), which leads to a loss of opportunity.

AI technology can help users produce an automated and polished resume effectively. It reduces user manual input, designs, and creates a spelling error-free, outstanding, and professional resume. According to Kolmar (2023), a professionally written resume enhances the earning potential by 7% to 32%, and mistakes found in a resume, such as bad grammar, will result in rejection

by hiring managers around 77%. This shows the importance of using a professional tool to create a resume in a professional and polished way.

1.3.2 Lack of career advisory services within the resume builder

Nowadays, people, especially fresh graduates, may face difficulties in seeking jobs, selecting a suitable career, or changing career paths due to the complexity of the working environment. Students seek career advice as they deal with rising tuition and dwindling student aid, as well as the stark headlines about college grads being unemployed or underemployed (Vespia et al., 2018).

Although there are various purposes for using career advisory services, such as receiving higher allowances or improving technical skills, these services demonstrate the importance of supporting job seekers, whether fresh graduates or professionals, in finding their ideal jobs and building the professional skills needed for their desired careers.

1.3.3 Lack of customization options on resume creation

It has been seen in many applications that most of them lack customization options for resume creation features in styling and formatting. This leads to users being unable to get their ideal design and turning to manual creation of a resume, which is time and energy-consuming, and causes extra workload for users to create a resume.

According to More et al. (2024), the earlier resume builder was not capable of providing features for customizing the content. This shows the reasons why we need sufficient customization options. This problem should be addressed to reduce the need and dependency on third-party design tools, and enhance the visual appeal and professionalism of the resume.

1.4 Aim and Objectives

This study aims to develop an Android mobile application for resume building and provide career advisory services to users to have better resume creation and career advice. Thus, there are several objectives for developing this smart mobile application. These objectives are set to solve the problem statements, achieve the aim of this study, and create a professional resume and provide career advisory services to users within the mobile application.

1) To integrate the Google Gemini AI model for resume creation based on user qualifications and skills.
2) To integrate the Google Gemini AI model for profiling professional career advisory recommendations that align with current user skills.
3) To develop customization options such as styling and formatting for the user in resume creation.

These objectives produce the scope of this study, and it will be achieved through the development of this mobile application.

1.5 Scope and Limitations of the Study

This study aims to develop a smart mobile application for resume building and career advisory services. There are several scopes and limitations on this study, which show what should be implemented and what should not.

1.5.1 Scope

The study focuses on creating a mobile application that enables users to build professional resumes and provide services on career advisory recommendations based on user qualifications and skills. This study is a fully functional mobile application that is compatible with the Android platform.

Initially, this study will integrate the Google Gemini AI model by using its API to automate resume creation, which reduces the time taken to create a resume, provides error-free content that is polished, and suggests improvements on creating a resume to ensure the professionalism of strong resumes based on the user's qualifications and skills.

Moreover, the application will use the API of Google Gemini AI to integrate the AI model for profiling and offer career advisory recommendation services that are aligned with the current user skills. AI will be able to support users in analysing job preferences of users, navigating suitable jobs and positions for them, providing ways for job searches such as job seeker websites, and skill enhancements.

Lastly, the study provides more customization options on styling and formatting, such as file type, font designs, and resume styles for users in resume

creation. File type customization options, such as in PDF and PNG format, are provided for user to generate and export their resume.

1.5.2 Limitation

The application will be developed for Android mobile devices, which may limit accessibility for users who prefer other mobile systems like iOS. The application is available for a limited languages, which could restrict it to non-English speaking users as they might not understand.

The application resume creation feature for exportation may be limited due to the exportation of resume file format in PDF and PNG only, which does not involve PPT, HTML, or other. The application may require a stable internet connection for certain features, such as AI support for resume builder and career advisory services. The application will offer customizable options, but there may be limitations on the extent of customization available to users compared to fully manual resume creation and designs that allow users to design and decorate their resumes. The application does not implement integration with job platforms or professional networks such as LinkedIn and others.

1.6 Contribution of the Project

In this project, the contribution that has been made is valuable. The main contribution to this project is developing the Android mobile application. No matter the frontend and backend, the smart mobile application with resume building and career advisory services will be developed by me. Gemini AI API will also be set up. In this application, React Native, GitHub, Android Studio, Visual Studio Code, and SQLite Database are used to develop this application, while Gemini AI is integrated by using the API key.

The integrated AI is connected, and a script that defines its services and system prompt is developed to make the AI suitable and match the Android mobile application. These AI scripts perform several functions such as filtering unrelated topics, focusing on a topic on a resume or career advice, analysis, enhancement, and elaboration. The upload PDF support is also available and defined in this script to allow Gemini AI to scan through the PDF file for resume enhancement.

1.7 Outline of Project

A smart mobile application that is compatible with the Android system and built to provide resume creation and career advisory services features is developed within the full methodology cycle. This mobile application, called 'Job Assistant', clearly stated its aims and objectives. A slogan could be added for this application, 'From resume to career, we've got you covered in one app.'

There will be 8 chapters for this project. Initially, start with the Introduction chapter, the Literature review chapter, the Methodology and Work Plan chapter, the Project Specification chapter, the Project Design chapter, the Project Development chapter, the Project Testing chapter, and lastly, the conclusion of the project. These chapters cover the entire project process cycle from the beginning to the end. Importantly, Scrum Methodology is used for this project and breaks down the phases into 5 sprints, such as sprints 1 and 2 are project initialization, sprints 3 and 4 are project development and testing, while sprint 5 is project completion to end all tasks for this project. The details of the project will be explained and discussed in each chapter.

CHAPTER 2

LITERATURE REVIEW

2.1 Introduction

Literature review is an important professional writing section for every study. A literature review provides an overview of existing knowledge, summarises, analyses, evaluates, and synthesises the relevant literature within a particular field of research (Monash University, no date). In this study, the literature review is important and should be done on existing systems, tools, and methodologies that could be used, and provide a source of ideas on developing this smart mobile application for resume building and career advisory services.

2.2 Review Existing Resume Builder

In this literature review section for this study, several mobile applications that have been reviewed can be easily found on the Google Play Store and are usually used with. For example, a resume builder with manual creation or AI support. Each application could discover its advantages and disadvantages after using the resume builder in the application.

2.2.1 CV Maker

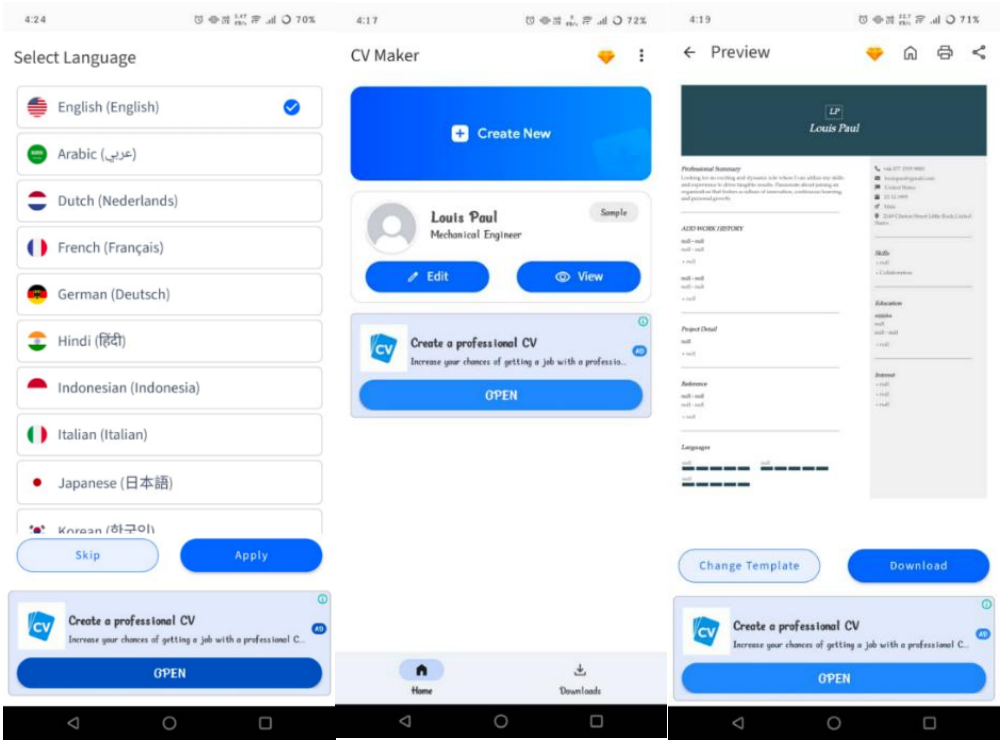


Figure 2.1: CV Maker Interfaces.

CV Maker is a mobile-based application that does not require the user to log in while using the application. It allows users to select their preferred languages on this application. This application provides sections for user to key in all their information to complete the resume, such as personal information and education. It also allows users to preview the resume. After completing all the sections, the resume can be downloaded, printed, or shared.

This application provides several benefits to users that attract users to use this application. For example, providing several template styles for the user to use on resume creation. Users can input the key freely and allow users to skip and ignore sections which does not have the information. This application provides a preview feature for users that benefits them in checking their resumes and discovering any errors before.

However, the disadvantages are limited customization options, does not contain AI functionality to support users in resume creation, and does not provide career advisory services for user to target their jobs. Resume creation in this application is time-consuming due to all input being fully manual insertion. Users who are unfamiliar with creating a resume will struggle during resume

creation. Navigation of this application requires many clicks to complete the resume, and limited language may lead to user unsatisfactory.

2.2.2 Resume Maker

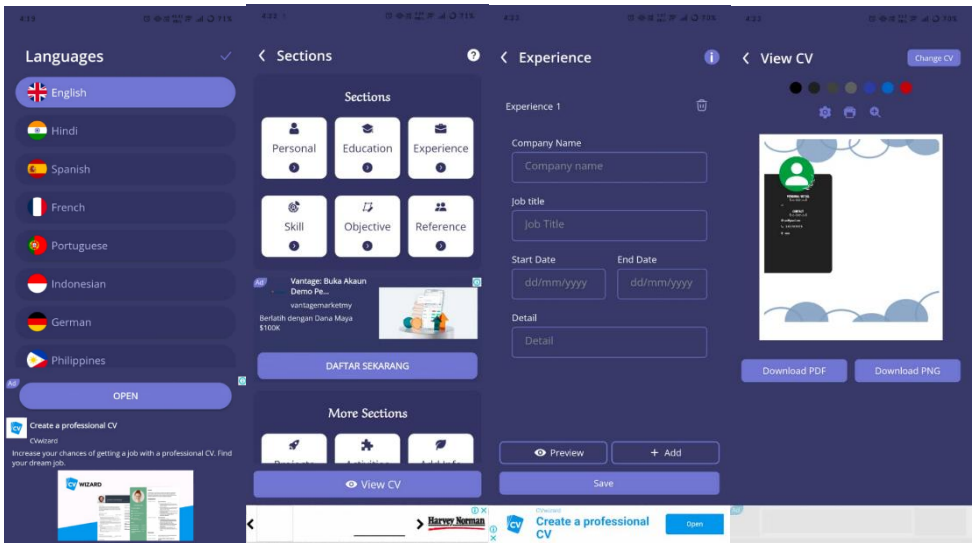


Figure 2.2: Resume Maker Interfaces.

Resume Maker is a mobile-based resume builder application with a simple purple UI and requires device storage permission to download and store the exported resume file on the device. It provides similar features to CV Maker, which are language selection, a section for the user to key in their information and allows the user to download their resume.

Every application has its pros and cons. This application provides customization options with more choices and is sufficient for users to select like colors and designs of templates in styling, but it is still limited in formatting, which is on the file type when exporting a resume. This application has an organized and arranged navigation among the sections, but some of the navigation will be misleading and cause confusion to users. Besides, this application is limited to several languages only, does not implement AI functionality in resume creation, and does not provide career advisory services for users. These disadvantages lead to time-consuming user resume creation and fully manual inputs, which take time to fill in information and watch the advertisement after every section.

2.2.3 Intelligent CV

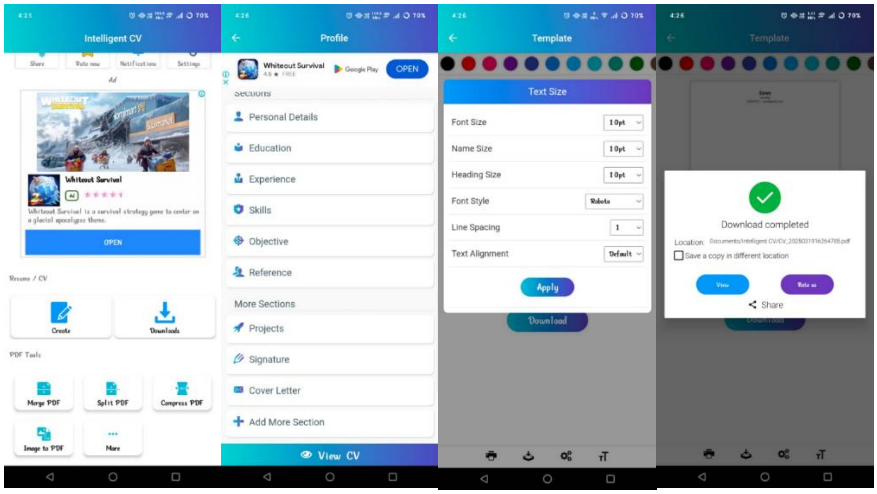


Figure 2.3: Intelligent CV Interfaces while resume creation.

Intelligent CV is a mobile application resume builder that provides the basic functionalities of a resume builder and other additional features, such as merging PDFs, splitting PDFs, and others. These additional features show its uniqueness.

The additional feature allows users to make modifications to the existing resume PDF file, like combining a cover page or remaking the resume. A reminder for each function is provided to alert users when input is missing and to show the status of the resume. Customization options are sufficient and extensive for users to select from the many template designs available.

Nevertheless, the exporting resume option is also set to the PDF file format only, and it has limited language support. There is no AI functionality integrated, which leads to the time-consuming nature of a resume. This application does not have a career advisory recommendation, so users might insert many unrelated details in the resume that are not suitable for finding a job.

2.2.4 CV Engineer

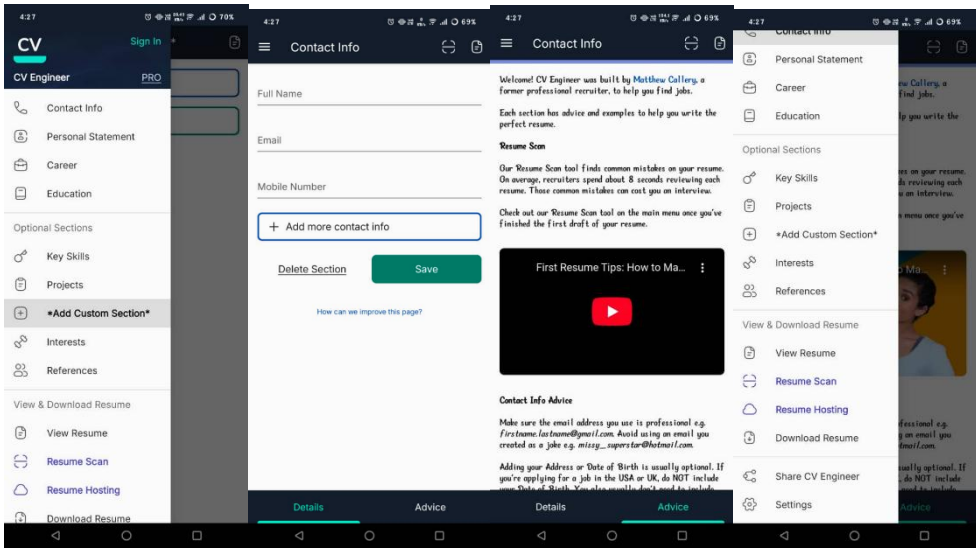


Figure 2.4: CV Engineer Interfaces.

CV Engineer is also a mobile application resume builder that has similar functionality to other resume builder applications. It provides login features for users, but it is optional and can be skipped. This application provides some guidance for users to refer to when creating their resume smoothly, and contact information is provided to allow users to provide feedback.

The advantages of this application that can compete with other resume builder applications are that some recommendations, introduction, and advice on creating a resume are provided for users to successfully create their resume, and it provides a feedback channel, like contact information, social media account, for users to provide feedback, subscribe, or complain.

However, there are still some disadvantages to this application. This application does not implement AI functionality, which leads to time-consuming resume creation. This application also has limited customization options in styling and formatting. There is no alert provided to users if there is an empty section that is missing information, which the user may forget and need to rework on their resume. The application is unable to direct download the resume locally on our devices, it should be shared with others, only then can the file be downloaded. This application does not provide career advisory services, as this is important for job seekers on a resume builder application.

2.2.5 Canva

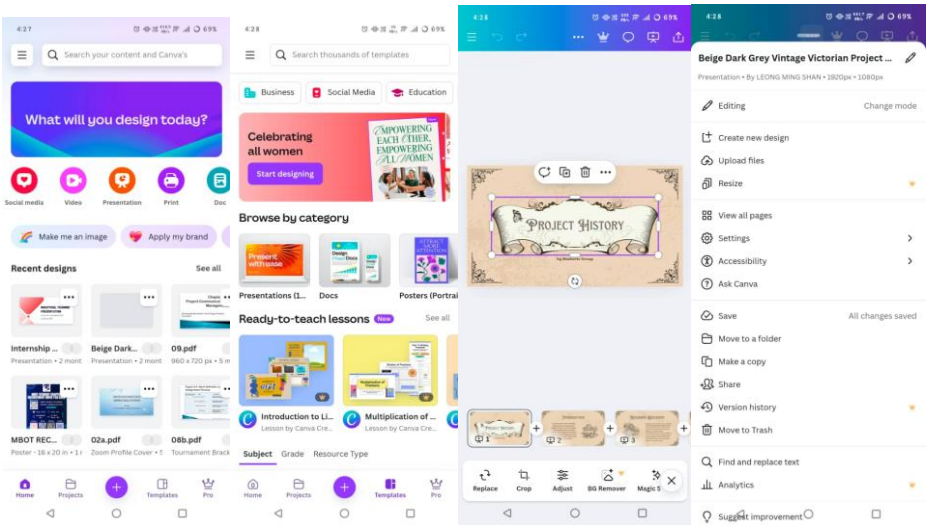


Figure 2.5: Canva Interfaces.

Canva is an application that can be used on the web and on mobile. It requires an Internet connection to save the changes within the application. It consists of many designs for different categories of usage, such as presentation slides, posters, logos, videos, and other designs. Users should log in using their accounts before starting to use this application. It includes most of the functionality, like uploading files, downloading, sharing, analytics, and other functions.

Canva is an application that can be said to be attractive to users with its unique capabilities. These capabilities and functionalities bring benefits to users. For instance, full freedom and sufficient customization options that provide many designs and styles. It allows users to replace, crop, adjust, remove the background, and add effects to diagrams.

Although the pros are attractive, the cons should not be ignored. Canva can be considered as fully manual designing and inserting the information. Users need to modify the template based on their needs, which is time-consuming, and users should determine the resume from the beginning of resume creation. It does not provide AI functionality, as users may not be aware of error occurrences, and it does not contain career advisory services for users. This shows Canva, a fully manual insert and design application for resume creation, which the target users may not be people who are busy and have less free time.

2.2.6 Kickresume

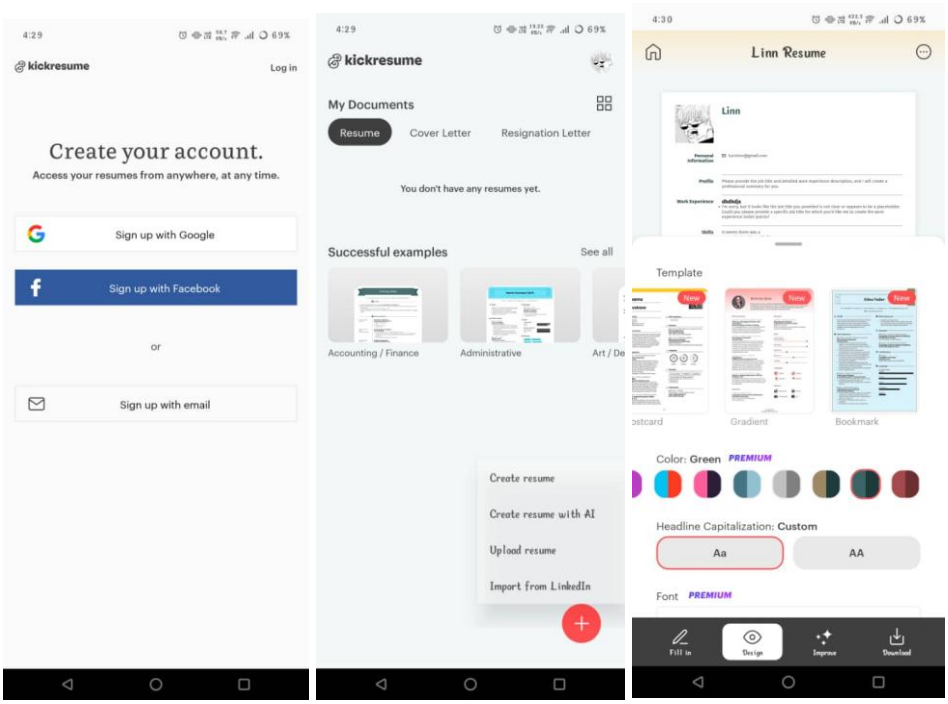


Figure 2.6: Kickresume Interfaces.

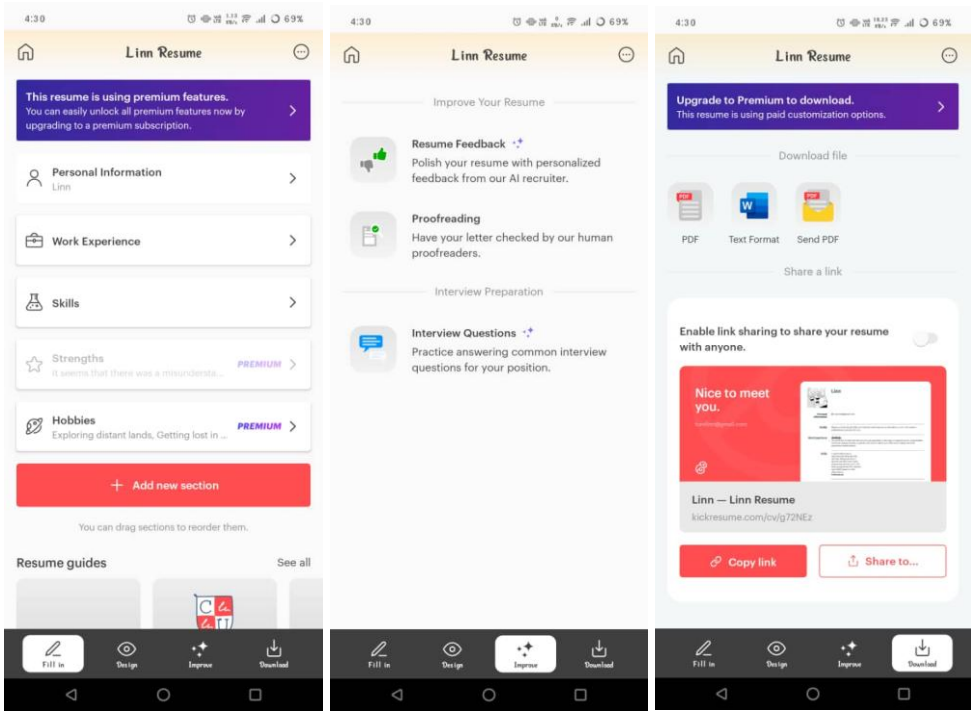


Figure 2.7: Functionalities of Kickresume.

Kickresume is a mobile application resume builder that requires a login before starting to use the application. The login account will be saved and automatically

generate a resume that contains the account details, such as profile picture and name. This application allows users to connect to their LinkedIn account, which can import their resume from LinkedIn. Besides, Kickresume allows users to create their cover page and resignation letter within this application to reduce the complexity of using different applications and systems.

This application brings advantages to users. Reduce complexity and enhance the opportunity for users to successfully get a job by allowing them to create their resume and cover letter within this application. This allows users to have a higher opportunity to get the job by having interview chances. Next, this application implements AI to support users in creating their resumes and provide feedback for the improvement of their resumes. AI also provides interview questions for users to allow them to get ready. Customization options are provided with sufficient choices in styling for users.

There are still some disadvantages to this application. Some of the navigation buttons are hidden, and unclear instructions on how to use this application. This application only has a limited language. The resume allows exportation in PDF file format and text formats. This application does not provide career advisory services to users.

Lastly, throughout these literature reviews, each resume builder application has its advantages and disadvantages, which cannot be ignored. The advantages and disadvantages might bring ideas and insights to this study, which may enhance improvements, add features, and build a more complete and useful mobile application. Table 2.1 summarizes the advantages and disadvantages of the existing system after being reviewed, while Table 2.2 shows the comparison of the functionalities of the existing system related to this study's problem statements, objectives, and scope.

Table 2.1: Existing system review summary.

Application	Advantages	Disadvantages
CV Maker	• Provide templates • Freely user input • Preview function provided	• Limited customization options • No AI

		• No career advisory services • Time-consuming • Bad navigation • Limited language
Resume Maker	• Sufficient customization options • Navigation organized and arranged	• Limited file type for resume exportation • Misleading navigation • Limited languages • No AI • No career advisory service • Time-consuming
Intelligent CV	• Additional features provided • Alert provided • Sufficient customization options	• Limited file type for resume exportation • Limited language • No AI • No career advisory service • Time-consuming
CV Engineer	• Resume creation advice provided • Feedback channel provided	• Limited customization operation • No AI • No career advisory service • No alert provided for missing details • Unable to directly download resume locally

Canva	• Full freedom and sufficient customization options	• Time-consuming • No AI • No career advisory service
Kickresume	• Reduce complexity • Implemented AI • Sufficient customization options	• No career advisory service • Bad navigation • Limited language • Limited file type for resume exportation

Table 2.2: Existing System Functionalities Comparison.

Functionalities / Existing Systems	Integrate AI	Career advisory services	Customization options	
			Styling	Formatting
CV Maker	×	×	Limited	Limited
Resume Maker	×	×	Sufficient	Limited
Intelligent CV	×	×	Sufficient	Limited
CV Engineer	×	×	Limited	Limited
Canva	×	×	Sufficient	Sufficient
Kickresume	✓	×	Sufficient	Limited

Finally, these disadvantages provide a clear insight into the existing system on the market and are related to the problem statements of this study, which are that it does not implement AI functionality, does not provide career advisory services, and has limited customization options in resume creation. This provides the main direction of this study to develop the resume builder and career advisory service mobile application.

2.3 Software Methodology Review

A software development methodology is a framework used to structure, plan, and manage the software development process, which ensures the software is

delivered in compliance with project specifications, on schedule, under budget, and with minimized project risks (Nikitin, 2024).

Software methodology plays a crucial role in every study as it provides a structured approach needed for determining and defining the software development life cycle (SDLC) stages. These stages are influenced by software methodologies based on the study's scope, objectives, and context. Thus, reviewing the software methodologies is essential to ensure alignment and suitability to the study's objectives and development flows.

2.3.1 Waterfall Methodology

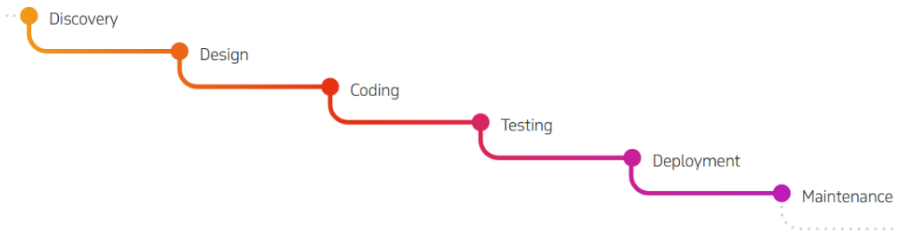


Figure 2.8: Waterfall Methodology (Nikitin, 2024).

The waterfall model was developed in 1970, and it is one of the oldest models for software development. Thus, it is a traditional software methodology.

According to Ahmed & Prasad (2016) and Nikitin (2024), the Waterfall model promotes a linear workflow and progresses from top to bottom through several phases with a well-planned and well-documented method of software development, also known as a sequential, cascading methodology. Each phase should be fully completed before moving to the next phase, and does not allow reverse actions to the previous phases.

Waterfall enhances simplicity and clarity. Besides, Waterfall is inflexible due to no extension or alteration of the project scope and requirements. It is slow and costly because of its rigid structure and tight control, hard to address issues due to testing only being performed at the end of the development process, and extensive requirements gathering causes an extended discovery phase and a lengthy project duration.

This methodology is suitable for projects that have a clearly defined project scope, stable requirements, a critical timeline, and budget control. It is

also suitable for a project team with less experienced and new which it is not suitable for a project that has unknowns and frequent changes.

2.3.2 Rapid Application Development (RAD) Methodology

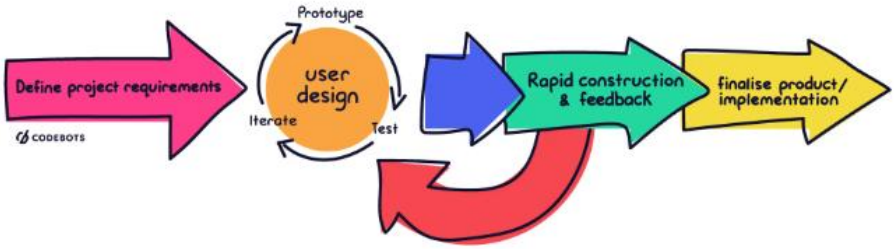


Figure 2.9: RAD Methodology (Chien, 2020).

The Rapid Application Development (RAD) model was introduced by James Martin in 1991 with the book titled Rapid Application Development. It allows expert programmers to fully utilize the underlying hardware, while non-experts can benefit from high-performance computing (Nalendra, 2021).

RAD promotes fast development cycles and rapid delivery of functional software, which enables products to be built in a shorter time frame without compromising the quality (Nalendra, 2021; Nikitin, 2024). RAD emphasizes building prototypes with given requirements and early user testing through multiple iterations until it fulfills the customer's satisfaction and meets all requirements.

The pros of RAD are that customer feedback helps eliminate risks, constant involvement, and iterative enhancement fulfill customer satisfaction, and streamlining development processes that enable projects to be completed within the timeline. However, the cons are highly dependent on responsive customer continuous feedback for testing and improvement, require an experienced developer, and a knowledgeable customer about the application area. This methodology is suitable for projects with well-defined requirements, developers and customers that are equally involved in the project, useful for small to medium projects that have time constraints.

2.3.3 Scrum Methodology

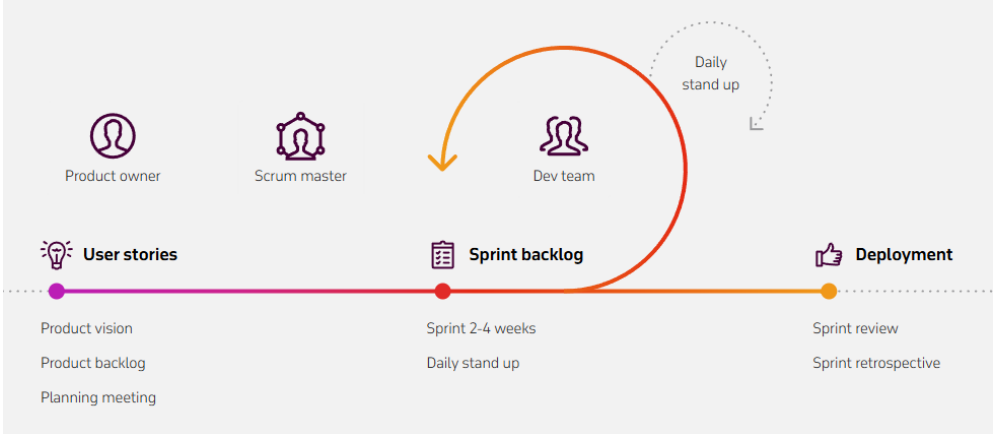


Figure 2.10: Scrum Methodology (Nikitin, 2024).

Scrum was first introduced by Schwaber (1995) as an iterative and incremental simple development methodology, and it is recognized as the most popular agile development methodology (Hron & Obwegeser, 2022). It involves the three roles, which are product owner, scrum master, and development team.

The product owner defines the development team's goals by creating and prioritizing the requirements. The scrum master is the project manager who enhances and monitors scrum values and rules in the project. The development team should meet the sprint goals, and at the end of each sprint, the development team should deliver a potentially releasable product increment (Alsaqqa et al., 2020), which reduces the resources wasted on rework and solves the previous defects.

Scrum advantages are that it is simple, allows quick resolutions to address issues, is highly responsive to changes, and promotes transparency and responsibility among the team. The disadvantage is that Scrum requires skilled and committed team members as it does not prescribe any practices, methods, and guidance on engineering practices (Alsaqqa et al., 2020; Nikitin, 2024).

Scrum was developed for small and co-located teams (Schwaber, 1995, cited in Hron & Obwegeser, 2022). This methodology is suitable for a project with a single team and an incremental delivery approach. It is not suitable for large-scale projects due to close collaboration and frequent communication within the team.

2.3.4 Kanban Methodology

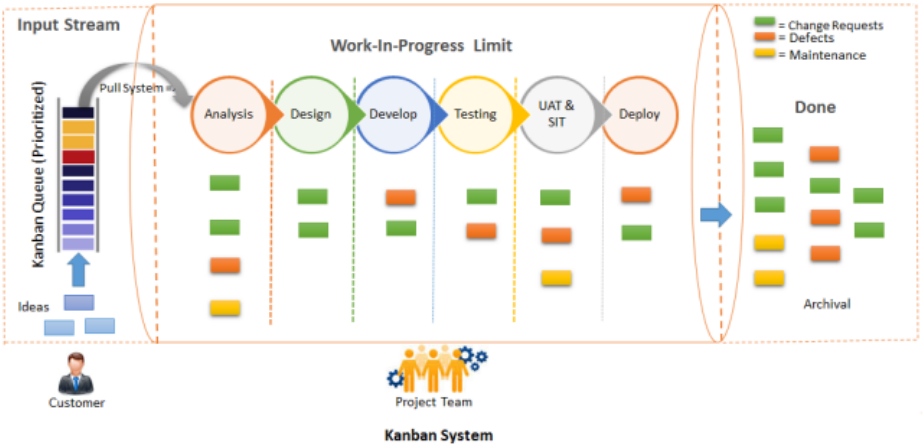


Figure 2.11: Kanban Methodology (Nikitin, 2024).

Kanban is developed by the Toyota production system (Ohno, 1998; Womack et al., 1990, cited in Zayat & Senvar, 2020). It is a framework under Agile methodology that emphasizes visualizing the entire project on boards, which are called Kanban board. According to Nikitin (2024), it allows the detection of blockers, necessary adjustments to the project plan, and quick task reprioritization in response to changed project priorities.

A Kanban board can track the project flow and display task status which ensuring task clarity and status alignment, which improves project efficiency. Besides, Kanban is often considered cost-effective due to it minimize waste of resources and maximize productivity and value. These advantages improve product quality, enhance collaboration among teams, and achieve consistent product delivery (Ahmad et al., 2013, cited in Zayat & Senvar, 2020). Nevertheless, Kanban disadvantages are a lack of time frames within the Kanban board, which may lead to unpredictable end times, require proper maintenance to avoid complexity, and high dependency on team updates.

This methodology is usually used together with Scrum and Waterfall methodologies, and it is suitable for small-scale teams, teams with no strict deadlines, and projects with evolving requirements.

Besides, throughout these methodologies that were reviewed, Table 2.3 is constructed to provide clear insight into the advantages and disadvantages of methodologies. The selection of implementing different methodologies is based on the project team size, time, cost, scope constraints, and others, which are

suitable for different project teams. It can be seen from this table that Scrum and Kanban are more suitable for this study, as Waterfall takes a lengthy project duration, and RAD highly depends on customer responses.

Table 2.3: Software Methodology Review.

Methodology	Advantages	Disadvantages
Waterfall	• Enhance simplicity and clarity	• Inflexible • Slow and costly • Hard to address issues • Extended discovery phase • Lengthy project duration
Rapid Application Development (RAD)	• Eliminate risks • Constant involvement and iterative enhancement • Streamlining the development process	• Highly dependent on responsive customer continuous feedback • Required experienced developer and knowledgeable customer
Scrum	• Quick resolutions • Highly responsive to changes • Ensure team alignment • Fast execution	• Required skilled and committed team member • Cause draining and burnout • No strict control on deadlines
Kanban	• Ensure task clarity and status alignment	• Unpredictable end time • Require proper maintenance

	• Improve team efficiency • Cost-effective • Improve product quality • Enhance collaboration • Achieve consistent product delivery	• High dependency on team updates

2.4 Software Tools Review

To develop a mobile application in this study, which is fully compatible with the Android system, suitable software tools like an IDE and a framework are needed to support the software development process that enhances efficiency and effectiveness. Proper tools bring benefits to the project as they reduce risk, improve code quality, enhance debugging, and others.

2.4.1 Android Studio

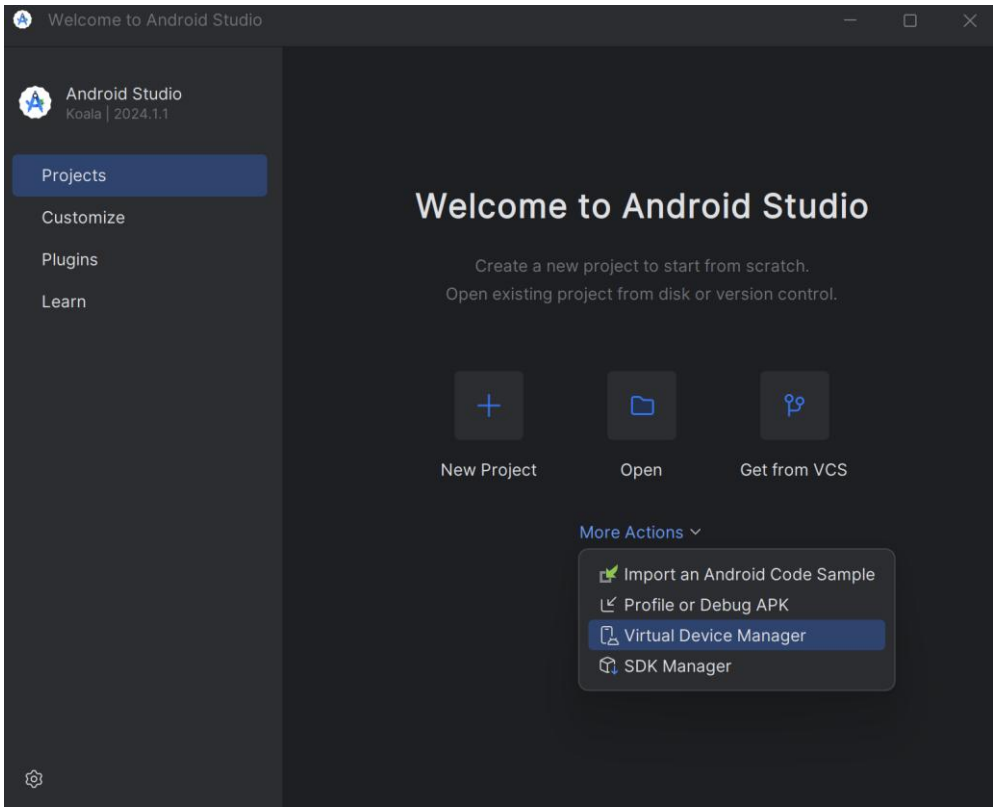


Figure 2.12: Android Studio.

Android Studio is popular and useful among developers. According to Changani (2024), Google's Android Studio is the most popular platform with a market share of 41.3% among developers due to its features, which are efficient and effective in developing applications.

Android Studio is the official IDE for Android application development. This application is designed to provide an easy-to-use and feature-rich platform for Android app development as it offers code editor, code refactoring tool, and debugging tool. Android Studio supports XML and Java programming languages, Android SDKs, and device emulators. The device emulators can also be used for testing the Android app because they emulate the real-world environment for the application to run like the devices that we use in daily life.

2.4.2 Visual Studio Code

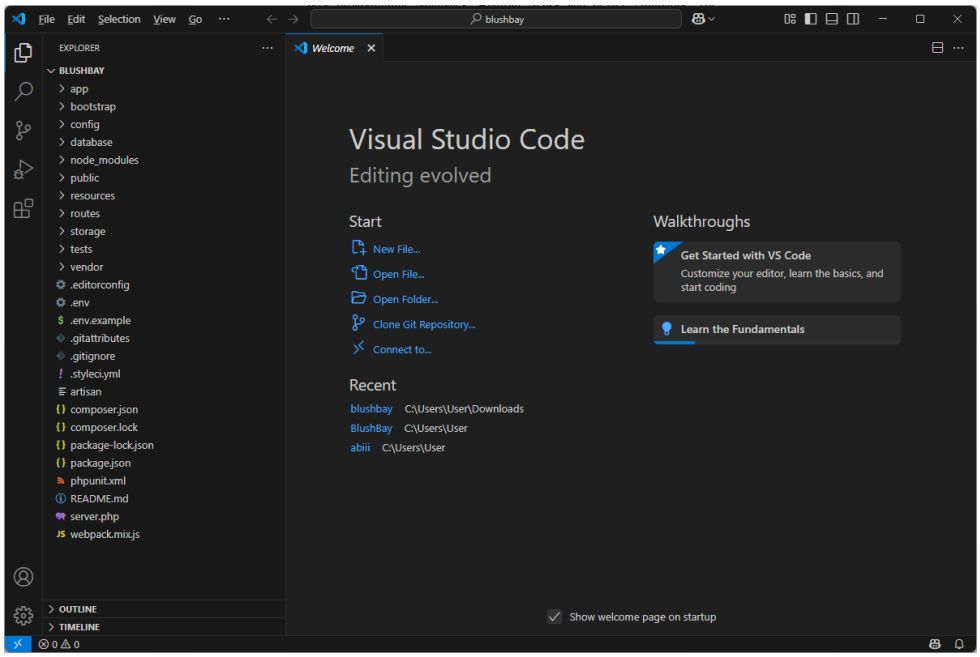


Figure 2.13: Visual Studio Code.

Visual Studio Code (VS Code) is an official IDE developed by Microsoft and used for Windows, Linux, macOS, and web browsers. VS Code provides various useful features like syntax highlighting, which allows developers to observe different usages of code, intelligent code completion that improves the efficiency of development, code suggestions, code refactoring, and debugging

tools. VS Code also allows embedded version control with Git, which can connect to the repository on GitHub. VS Code supports any programming language for development, such as JavaScript, TypeScript, Python, HTML, C++, PHP, and others, which are useful and sufficient for developers to develop an application. It also allows developers to install the various extensions that appear in the marketplace within VS Code, such as GitHub Copilot.

2.4.3 GitHub

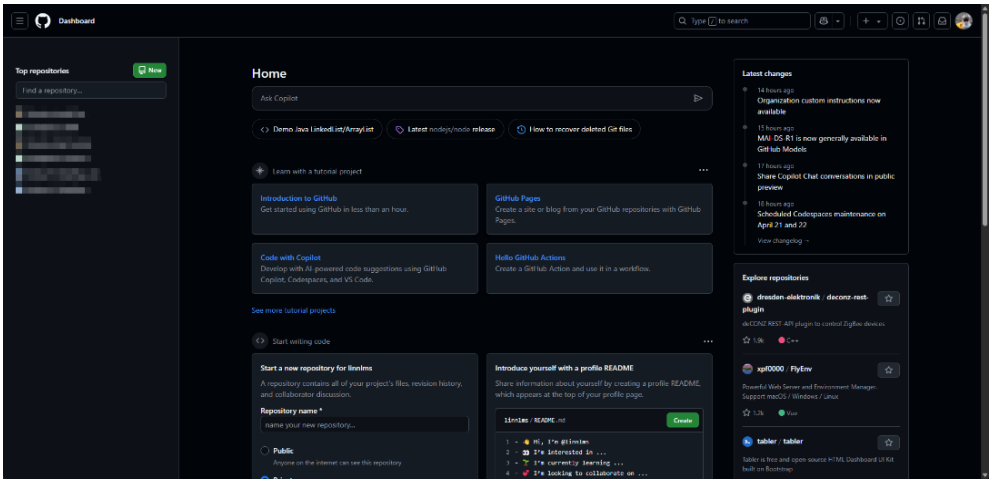


Figure 2.14: GitHub in Browser.

GitHub is a proprietary platform for developer that allows them to create, store, manage, update, modify, and share their code by using commands and functions provided. GitHub uses Git to provide distributed version control, which developers can commit and sync their code after every modification, and provides access control like public repository or private repository, which invites collaboration from developers to contribute to the repository, continuous integration and alignment of codes, and other features. It is useful while using VS Code for embedded version control. However, before directly using it in VS Code, developers should set up Git on their local devices, such as installing Git, configuring it, and creating a new repository or cloning an existing repository.

2.4.4 React Native

React Native is an open-source framework used to build cross-platform applications such as Android or iOS. It helps developers to create native apps with a single codebase that can be deployed and are compatible with multiple

platforms. React Native able developer to use TypeScript or JavaScript as the programming language to develop their mobile application, and it provides comprehensive debugging tools. React Native can integrate with different types of functions, such as geolocation, APIs, cloud services, and others, which are useful in mobile app development.

2.4.5 Flutter

Flutter is an open-source framework for cross-platform application development, like mobile, web, desktop, and others, from a single codebase that is supported by Google and open to everyone. It is fast, productive, and flexible in the development process. Flutter is powered by Dart. Dart is an approachable, portable, and productive programming language optimized for UI, performance, and fast development for any platform frontend development. It is suitable for use in high-quality UI applications, especially UI-heavy applications.

Lastly, the review of software tools used in the software development of Android mobile applications for a resume builder with career advisory services in this study is essential due to using suitable tools allows the development process to run smoothly and enhances efficiency and effectiveness. The tools that were reviewed are sufficient and useful for developing an Android mobile application, and the usage of each tool is summarized in Table 2.4.

Table 2.4: Software Tools Review.

Tools	Usage
Android Studio	• Official IDE for Android application development • Easy-to-use and feature-rich • Support XML and Java programming languages, Android SDKs, and device emulators
Visual Studio Code	• Official IDE developed by Microsoft • Provides various useful features • Allow embedded version control with Git • Support any programming languages for development • Allows installation of extensions
GitHub	• Proprietary platform for developers

	• Provide distributed version control • Useful with Visual Studio Code

React Native	• Open-source framework
	• Used to build cross-platform applications
	  o Create native apps
	  o Single codebase
	  o Compatible with multiple platforms
	• Allow TypeScript or JavaScript for development
	• Provide comprehensive debugging tools
	• Allow various functions

Flutter	• Open-source framework
	• Used for cross-platform application development
	  o Single codebase
	  o Supported by Google
	• Fast, productive, and flexible
	• Powered by Dart
	  o Approachable, portable, and productive programming language
	  o Optimized for frontend development
	  o Suitable for high UI quality applications

2.5 Database Review

A database plays a crucial role in every application. It allows the organized collection of data stored electronically in a computer system and controlled by a database management system (DBMS). The common databases used nowadays are modelled in a series of tables and use structured query language (SQL) for writing and querying data (Oracle, no date), and these databases are called relational databases.

Database is mainly categorized into 2 types, which are local databases and cloud databases. Local database provides on-device storage, while cloud database offers sync and real-time features that require internet connections. A database allows users to manage their data and interact with it by using various

operations, such as CRUD operations. This allows the data can be accessed, managed, and organized easily with the use of a database.

2.5.1 SQLite

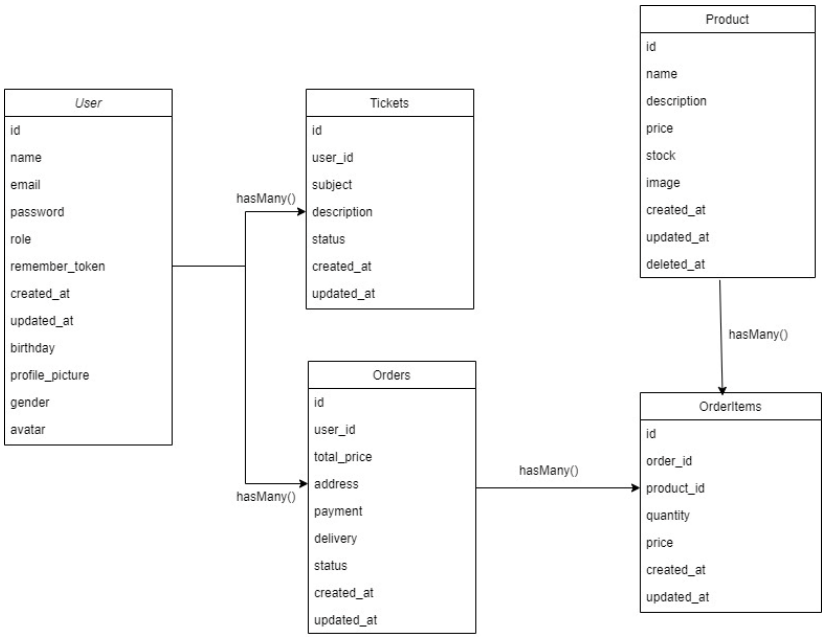


Figure 2.15: Example of an SQLite Database for an Assignment Project.

SQLite is a database engine that is lightweight, fast, self-contained, and highly reliable that implements a fully SQL database, which is familiar to developers. It is a relational database that organizes data into tables that contain rows and columns. The SQLite database engine is commonly used in the world nowadays. It is often used as the on-device storage and on-disk file format for applications that are used to store and allow users to manage data without the need for an internet connection. These characteristics of SQLite ensure the serverless and file-based nature of the database, which the database is managed in a single .sqlite or .db file. SQL is the language used to interact with the database by offering various operations like create, read, update, and delete data. However, to develop a SQLite database, it requires a full manual on SQL writing and updates.

2.5.2 Room

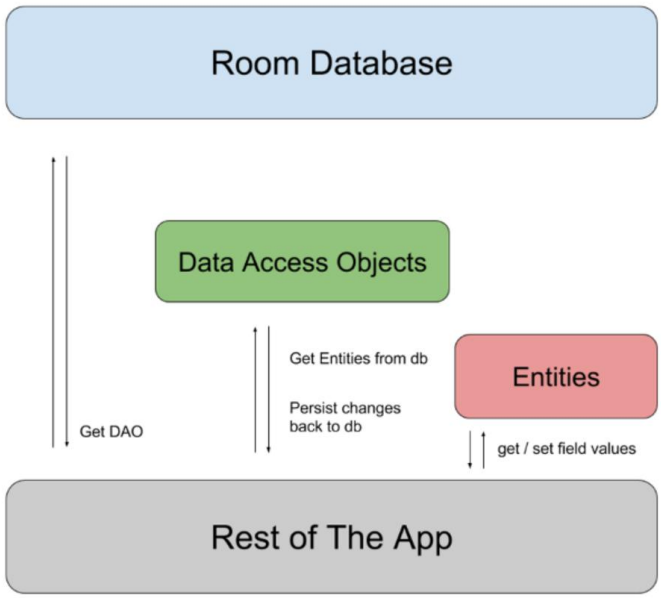


Figure 2.16: Room Database (Android Developers, no date b).

Room is a persistence library that offers an abstraction layer over SQLite to allow more reliable database access while utilizing the capabilities of SQLite. According to Android Developers (no date a), to use Room, it must be set up and the dependencies in the build.gradle file before starting such as RoomRxJava and other dependencies to the project, while for the configuration compiler, Room Gradle Plugin will be used to configure options for the Room compiler. By using Room, it offers compile-time SQL query verification, optimized database migration paths, and convenience annotations which reduce repetitive and error-prone boilerplate code (Android Developers, no date b).

2.5.3 Firebase Realtime Database

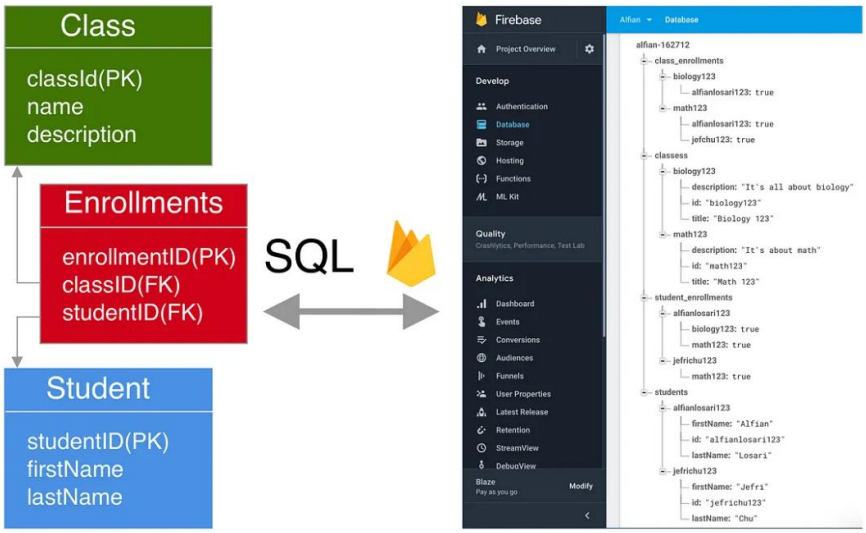


Figure 2.17: Example Firebase Realtime Database (Losari, 2018).

Firebase Realtime Database is a NoSQL with a hierarchical, structured, cloud-hosted database that is one of the core products under Google's Firebase mobile platform. It allows non-experts to build applications fast (Losari, 2018). As a characteristic of cloud databases, it is good for real-time features and cloud-sync functionality. According to Android Developers (no date c), Firebase Realtime Database allows storing and syncing data as JSON across all clients and maintains its availability while the application goes offline, which is stored as offline caches. This database is useful for cross-platform applications that allow all clients to share one Realtime Database and receive updates with the latest data automatically. However, it is tied and dependent on Google Cloud, and it will not store relational data because it is a non-relational database that can be called NoSQL.

Lastly, no matter what type of database, it is important and a must to develop it on the application to benefit from its powerful data storage with clear, organized, and lightweight data. The databases that were reviewed had a common advantage, which is easy to use by non-experts, allowing the development of a database with simple operations or several lines of code. Table 2.5 presents the advantages and disadvantages of these databases, which provide

a clear insight into these databases that can be easily understood and help select the suitable database for this study.

Table 2.5: Advantages and Disadvantages of Databases.

Database	Advantages	Disadvantages
SQLite	• Fast local storage • Full control over SQL • No internet needed • Familiar to Developers	• Fully manual on writing and updating SQL queries • No cloud syncing
Room	• Local Storage • Offers annotations, SQL query verification, and optimized database migration paths • Reduce boilerplate code	• No cloud syncing • Need knowledge to understand entities
Firebase Realtime Database	• Real-time syncing of data • Cloud storage and backup • Offline caching built-in	• Internet required to sync data • JSON-based • No relational data • Tied to Google Cloud

2.6 Testing Tools Review

Testing is important in a project to reduce and avoid defects introduced to the application and ensure good software quality assurance. To avoid bugs and maintain user satisfaction, testing needs to be done. Testing could be done in many ways, such as unit testing, integration testing, system testing, end-to-end (E2E) testing, user acceptance testing (UAT), and others. It could also be done in manual testing or automated testing. Thus, there are tools to support the testing process to enhance the software quality assurance.

2.6.1 JUnit

Listing 2.1 `CalculatorTest` test case

```
import static org.junit.jupiter.api.Assertions.assertEquals;
import org.junit.jupiter.api.Test;

public class CalculatorTest {

   @Test
   public void testAdd() {
      Calculator calculator = new Calculator();
      double result = calculator.add(10, 50);
      assertEquals(60, result, 0);
   }
}
```

Figure 2.18: Example of JUnit Test (Tudose, 2021).

JUnit is an open-source testing framework, and the current version is JUnit 5. JUnit offers a foundation for developers to test Java and allows developers to write and run repeatable tests. JUnit is designed to utilize unit testing in Java due to it allows writing test cases, executing tests, providing detailed feedback for debugging, and is easy to integrate with Android Studio. According to Tudose (2021), JUnit allows each unit testing to run independently, detect and report errors, and is easy to define a unit test. This framework should support writing useful tests, creating tests that retain their value over time, and lowering costs on writing tests by reusing the code.

2.6.2 Jest

Figure 2.19: Example of Jest (Jest, no date).

Jest is a simple, fast, and safe open-source JavaScript testing framework that can be used with TypeScript, React, Angular, and others. It is primarily designed for unit testing and integration testing for React applications. According to Jest (no date), Jest aims to work out of the box with zero config for JavaScript projects. It provides various built-in features such as testing snapshots testing which allow testing to keep track easily with large objects, offers isolated tests which test are parallelized to run in their own, provides code coverage, and easy mocking in tests.

2.6.3 Espresso

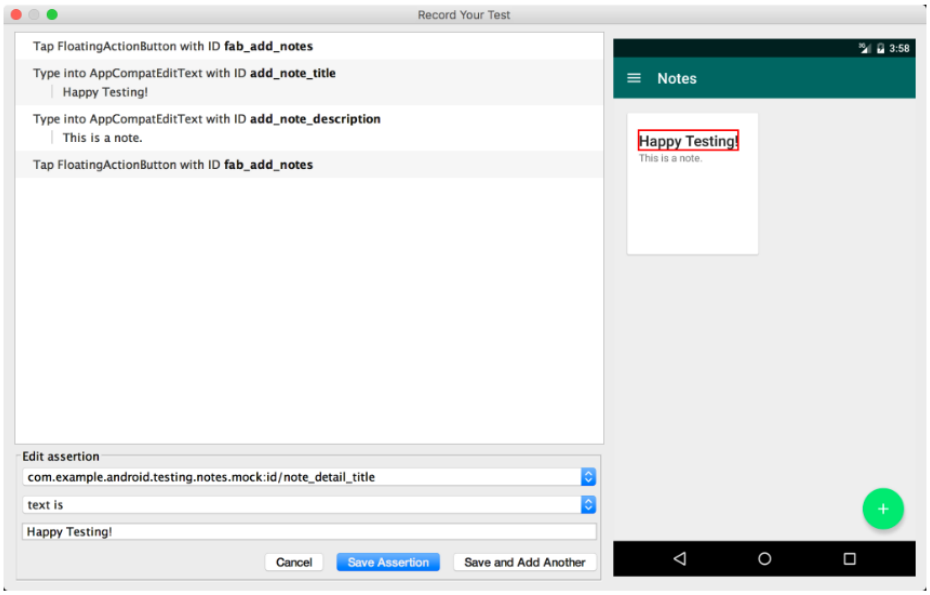


Figure 2.20: Example Espresso Testing (Android Developers, no date e).

Espresso is an open-source testing framework for Android UI testing that allows developers to write fast, concise, and reliable UI tests for their applications. According to Android Developers (no date d), the core API of Espresso is lightweight, easy to learn, and open for customization. Espresso tests can achieve expectations, interactions, and assertions without modifying the contents and structures of the UI elements on the device or emulator. Espresso is mainly used in automated testing as an integral part of the development lifecycle. It will not provide direct access to activities and views of the application due to avoiding test flakiness while holding activities and objects in

views. Test flakiness will cause unreliable, error-prone, and instability to testing. Thus, Espresso allows for safe operation on views by implementing subclasses such as ViewAction and ViewAssertion.

2.6.4 React Native Testing Library (RNTL)

Figure 2.21: Example render() Testing from RNTL (Rahman, 2024).

React Native Testing Library (RNTL) is a lightweight open-source testing library for React Native. It allows core queries for testing components to be implemented without details by using component renders. RNTL is reliable and maintainable, which supports testers to write maintainable tests with using lesser effort and offers testing dependencies for React Native. Thus, the dependencies for RNTL should be installed and used with Jest. RNTL usually works with Jest to test a React Native application that is developed using JavaScript or TypeScript. Thus, it can be said that RNTL with Jest are normally used to test unit testing and integration testing.

2.6.5 Firebase App Distribution

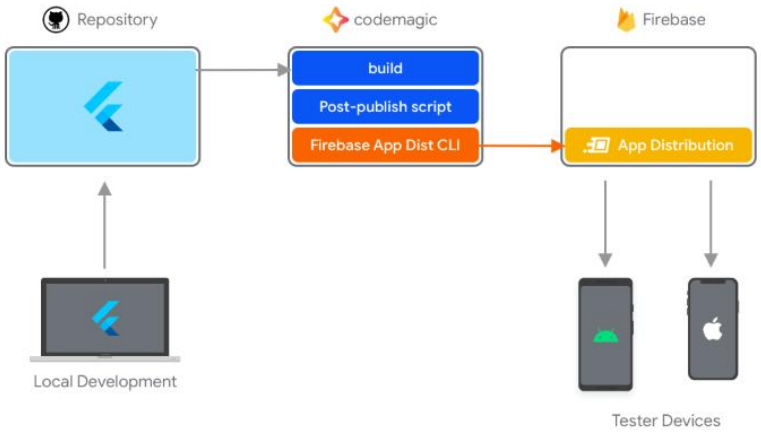


Figure 2.22: Firebase App Distribution Workflow (Hughes, 2020).

Firebase App Distribution is a platform for developers to distribute their pre-release versions of applications to trusted testers quickly. It supports version tracking and collects feedback earlier. This platform is mainly used for UAT testing, which requires users to experience the application. It allows developers to manage a beta testing program that is fast and flexible, which offers an easier way for testers to gain valuable insight about their application.

Finally, testing is important for a project. The purpose of testing is to ensure software quality assurance, reduce bugs and defects that cause failure, and others. Proper testing might reduce the costs of fixing bugs. Thus, a suitable testing tool used in a project is important too. Throughout the testing tools that were reviewed, these tools might be suitable for this study, and Table 2.6 will show the advantages and disadvantages of these testing tools, which allow different testing to be executed effectively and efficiently.

Table 2.6: Advantages and Disadvantages of Testing Tools.

Testing Tools	Advantages	Disadvantages
JUnit	• Fast execution on JVM • Test logic independently • Allow repeatable testing • Utilize unit testing	• No UI test • Only test for Java Project

Jest	• Simple, fast, and safe • Focus on unit testing and integration testing • Zero config • Provides built-in features	• No real device testing • Limited UI testing
Espresso	• Focus on UI tests • Automated testing • Avoid test flakiness	• Slower than unit test • Need emulator or real device
RNTL	• Lightweight • Avoid implementation details • Reliable and maintainable • Focus on unit testing and integration testing	• Slower tests • Limited interactivity testing • Need to work with Jest
Firebase App Distribution	• Fast and Flexible • Easy to share for testing • Support version tracking • Collect feedback earlier • Focus on the UAT test	• No actual test • Require Firebase setup

2.7 AI Technologies Review

To integrate AI technology in this resume builder and career advisory services mobile application, the AI tools, such as APIs, should be reviewed to clarify the tools that are suitable to this study, as there are many different types of AI usage such as Chat-based AI, Digital Image Processing (DIP) AI, Natural Language Processing (NLP) AI, and others. In this study, based on the scope and objectives, Chat-based AI is related and suitable for integration in development.

2.7.1 ChatGPT

ChatGPT is an artificial intelligence (AI) chatbot model introduced by OpenAI, Inc. in the USA, which interacts conversationally. ChatGPT can answer

questions by users, provide different responses based on user preference, reject inappropriate requests, and others. It provides more creative and nuanced responses while better maintaining longer conversations.

ChatGPT consists of many sizes, such as GPT-4o, GPT-3-mini, and others, each size used for different functions. For example, GPT-4o is more intelligent and professional at answering technical questions. OpenAI has provided ChatGPT APIs that allow developers to use. These APIs can be integrated into their development, which provides various features like basic ChatGPT functions, image generation, text-to-speech, speech-to-text, and others.

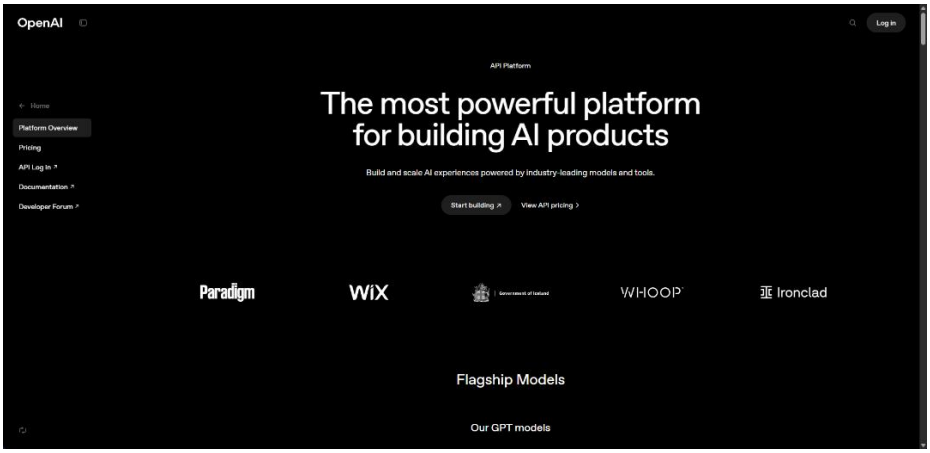


Figure 2.23: OpenAI Official Website for APIs.

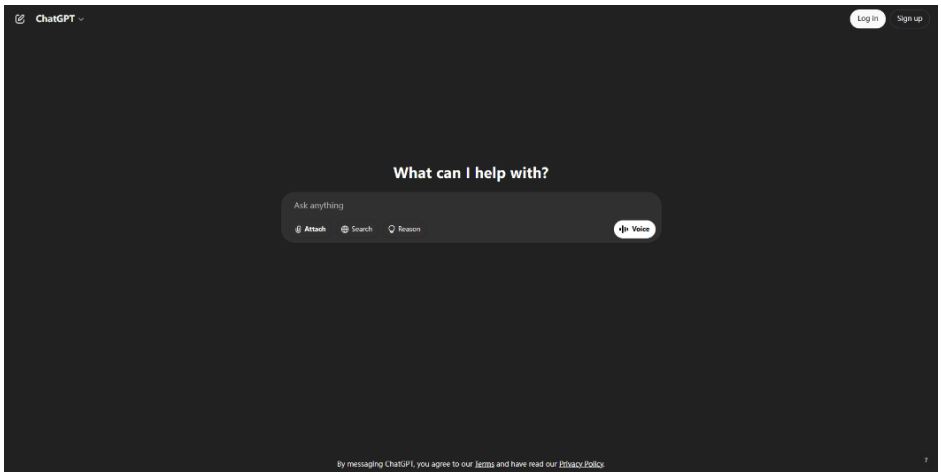


Figure 2.24: ChatGPT Chat Space.

2.7.2 Google Gemini

Gemini is a generative AI chatbot developed by Google which it is also known as Bard. Gemini was built to be multimodal, which means it can generalize and comprehend with ease, execute across and combine different types of information, including text, code, audio, image, and video (Pichai & Hassabis, 2023), which allows advanced coding and other features. Gemini has different sizes, which are Gemini Ultra, Gemini Pro, and Gemini Nano. Developers can access and use the Gemini API in Google AI Studio, and it seamlessly integrates into Android and the Google ecosystem. Google AI Studio allows developers to execute development and testing with a generous free limit while the project is not commercial and is used for learning and personal projects.

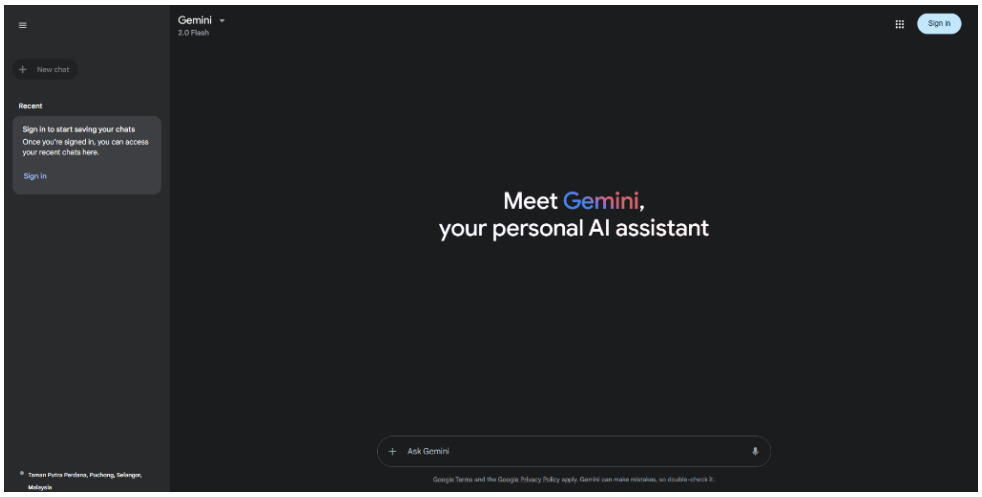


Figure 2.25: Google Gemini Chat Space.

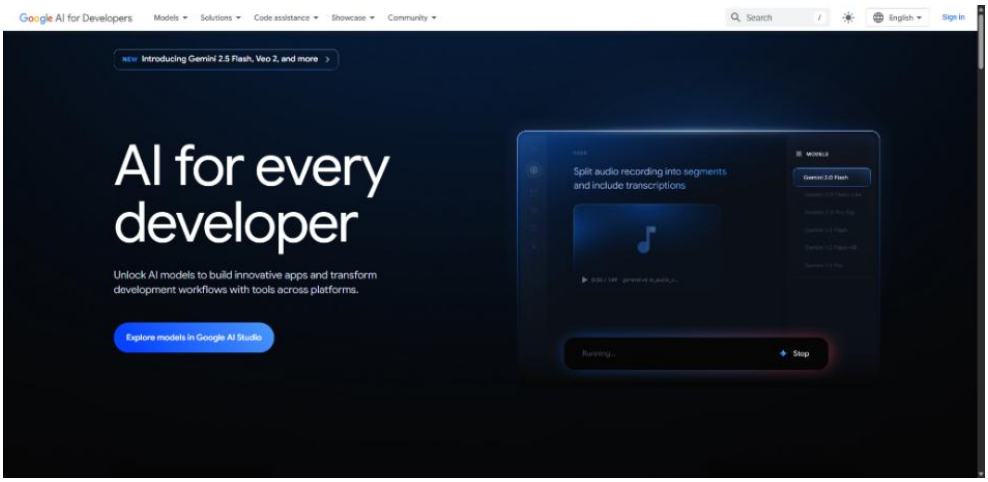


Figure 2.26: Google Gemini API for Developers.

2.7.3 DeepSeek

DeepSeek is a chatbot AI model introduced by a Chinese AI company in January 2025. DeepSeek has limitations compared to ChatGPT and Gemini due to its newness, such as being unable to understand images and being a text-based AI only. DeepSeek provides its API to developers, and it provides a platform for developers to check the API status, like usage, API keys, top up, billing, and others. This API model is updated and maintained with different versions to boost performance, smarter capabilities, and others by the DeepSeek Team. Developers just need to integrate the API key into their application.

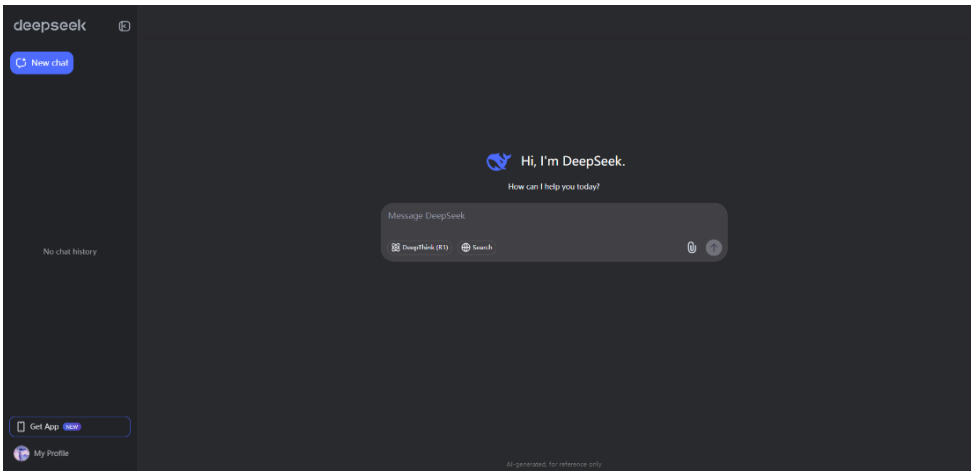


Figure 2.27: DeepSeek Chat Space.

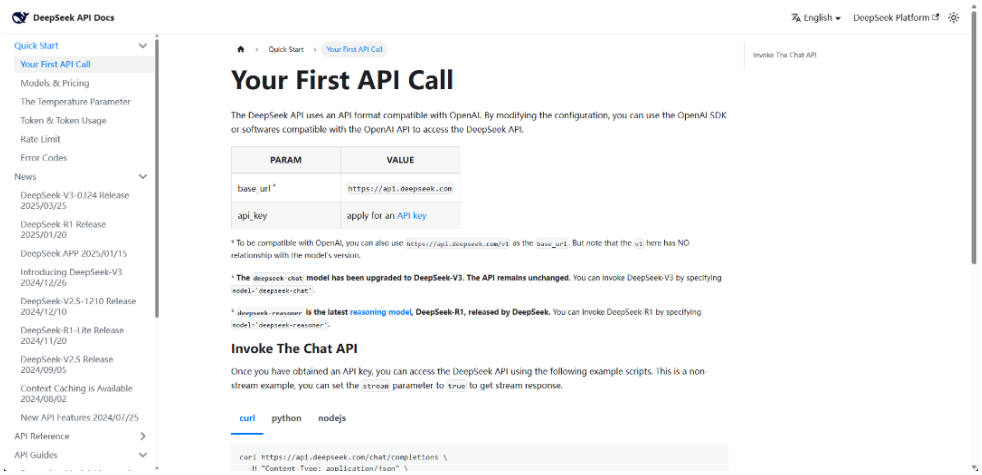


Figure 2.28: DeepSeek API Details.

Finally, these AI tools are chat-based AI models that are suitable for integrating into the mobile application. These chat-based AI models are useful

in supporting resume creation, as they can generate the content directly based on user input and provide career advisory services for users to address their confusion. This allows users to reduce effort in making a resume and searching for information about an unknown and new job.

2.8 Summary

Throughout the literature review, it can be discovered and highlighted that the importance of understanding existing systems, methodologies, tools, and technologies relevant to this study on developing a smart mobile application for resume building and career advisory services. This literature review explored the benefits and limitations of the reviewed details that need to be selected and used to implement this study.

To support mobile application development, essential tools should be suitable and able to improve efficiency and effectiveness in the development process. Each tools offer unique capabilities for development, no matter whether used to perform coding, testing, data storage, or other functions. While the AI technologies that have been reviewed offer chatbot features and more, which can enhance user experience, professional resume creation, and proper career advice.

CHAPTER 3

METHODOLOGY AND WORK PLAN

3.1 Introduction

In this chapter, the methodology and work plan will be discussed in several sections, such as project workflow, project planning and scheduling, methodology used, tools, and technologies. The project workflow is explained in a flow chart to provide clear insight into the workflow. The project planning and scheduling will be divided into different sub-sections, such as Work Breakdown Structure (WBS) and a Gantt chart of the project. The methodology used in this study is Scrum, and all the tools that were reviewed in Chapter 2: Literature Review will be selected and used while developing the Android mobile application, as this methodology and these tools are suitable, and I have the knowledge on using them. For this project, the Android mobile application will be named “Job Assistant”.

3.2 Project Workflow

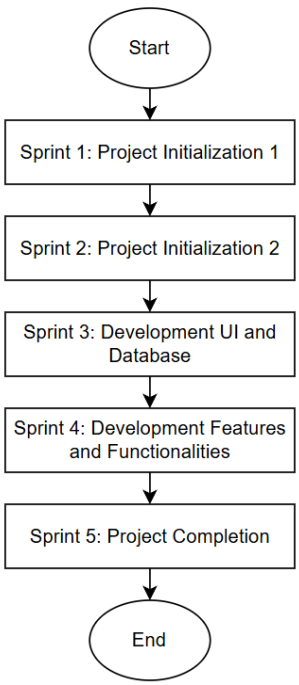


Figure 3.1: Overall Project Workflow.

Figure 3.1 shows the overall workflow for this project. The details of the project workflow include the 5 sprints for the Scrum Methodology. Gantt charts and work breakdown structure (WBS) support the development of Scrum Methodology in task breakdown and timeline of completion. Software Development Methodology will describe the overall planning for this project workflow. More specifications on this project will be described in Chapter 4: Project Specification.

3.3 Software Development Methodology

Scrum methodology is selected to use in this study. This is because Scrum allows fast execution of tasks, which is highly responsive to change. Scrum is an agile methodology that requires only a small-sized project and team, which suits this study because it is adaptive and only has a solo developer. Scrum can ensure efficient resource allocation, avoid overloading, early identification of potential challenges, and make necessary adjustments.

3.3.1 Requirements Gathering

The requirement gathered is based on the review of the existing systems, which provide insights to develop this Android Mobile Application. This study aims to improve and enhance the functionalities of the existing system after using it. Other than that, the project scope, objectives, and problem statements of this study support the requirement specifications because the requirements should meet all scope and objectives to solve the problem statements.

3.3.2 Planning

Project planning and scheduling are crucial aspects of project management in a project that ensure the smooth execution of tasks within a defined timeline.

In Scrum, a product backlog should be created. This product backlog prioritized the list of work items, such as features, bug fixes, tasks, and other items. The tasks are planned in 5 sprints for this project.

The first 2 sprints for this project are sprint 1: Project Initialization 1, and sprint 2: Project Initialization 2. Project initializations break down into two sprints due to preparation for the introduction, literature review, requirements gathering, project planning, and designing for this project requires a longer time

to ensure tools, technologies, sources, and data information come from an authentic source, and planning for a better mobile application with reliability. Thus, it is broken down into two sprints to complete it. Next, the project will continue to the development phase. Sprint 3 (Development for UI and database), sprint 4 (Development for features and functionalities), and sprint 5 (Project Completion) will be done. In parallel, the report documenting and recording the processes and results of this project will be completed.

As a Scrum characteristic, the development, testing, deployment, and retrospective will be completed within each sprint, and the retrospective will be the end of the sprint before the sprint is moved to the next sprint. By this characteristic, Scrum shows it is significantly different from Sequential methodologies like Waterfall. Development, testing, and deployment are the main parts to determine an application's success. These phases ensure reliable, user-satisfaction, objective met, and other purposes. A retrospective is a session to gain feedback and suggestions to enhance and improve the application promptly.

Sprint 1 is the first sprint for the initialization of project details, such as introduction and conducting a literature review for this project, including existing systems, methodologies, tools, and technologies that could be used in this development lifecycle. Sprint 2 is the second sprint for the initialization of this project, about requirements, planning, and designing. This sprint will be set to gather the requirements from existing systems, start planning, and designing for this project. Project workflow, project methodology, tools, and technologies will be planned and used for the development lifecycle. Designs like use cases, flow charts, initial prototypes, database structure, and others will be constructed in this sprint.

Sprint 3 will be the UI and database development of this Android mobile application by using software tools, including a SQLite database created by a Python script. Sprint 4 will continue developing all the features and functionalities of the application and integrating the AI tool, which will lead to more time taken to complete this sprint. The resume creation and career advisory services will be the main functions to develop and test. Sprint 5 will be the last sprint for this project, as it is the start to compile all parts of the features together as a system. Throughout the sprints 3 and 4, the features and scripts will be

developed, tested, and deployed into the virtual machine environment. The main target of this sprint is to get a complete and good software quality assurance of a fully functional Android mobile application that meets the project requirements, objectives, and solves the project problem statements.

In this project, the tools and technologies used to develop the Android mobile application are planned. Android Studio, Visual Studio Code, GitHub, and React Native are used for development, SQLite is used for the database, Jest and RNTL are used for testing, while Google Gemini AI is used for integration of AI functionality. The virtual machine environment, the device emulator, will be provided and used with Android Studio.

3.3.3 Design

For the design phase of this study, Scrum does not fully support the techniques in this phase. This phase will create a solid foundation for the project and design the initial prototype of the mobile application, defining use cases, constructing flowchart diagrams and database structure, and developing architecture and database design for this project. The design phase will define how the application will work and what it should look like.

For the use case diagram, there will be two roles such as the admin role and the user role. The admin is the owner of the application, who controls the application, and the user is the customer or normal user who uses the application. Within the use case diagram, the activities and features will be clarified and categorized for each role that they are authorized to use.

The flowchart and database structure are designed to provide insights and set the project specifications. The flowchart will be used and designed for project workflow and ease the understanding of project features that will be integrated and implemented in the application based on the requirements, such as resume creation, career advisory services, and others. While the database structure is designed to offer a clear view of the database that will be developed in this application, which is SQLite Database, such as the relationship of database tables, the elements and keys used, and other details of the database.

For the initial prototype, the user interface will look like the initial prototype designs, and the flow of all features for a user to use this application. In this initial prototype, it will state the colors, development libraries such as

vector icon family, font family, and colors used. The entire directory of vector icons library, font family choices, and colors used can be found online, such as sources from GitHub for vector icons, Google Fonts for font family, and Color Hunt for colors. In the initial prototype of this project, Ionicons for vector icons, PT Sans for font family, and different types of colors will be used to design the Android mobile application.

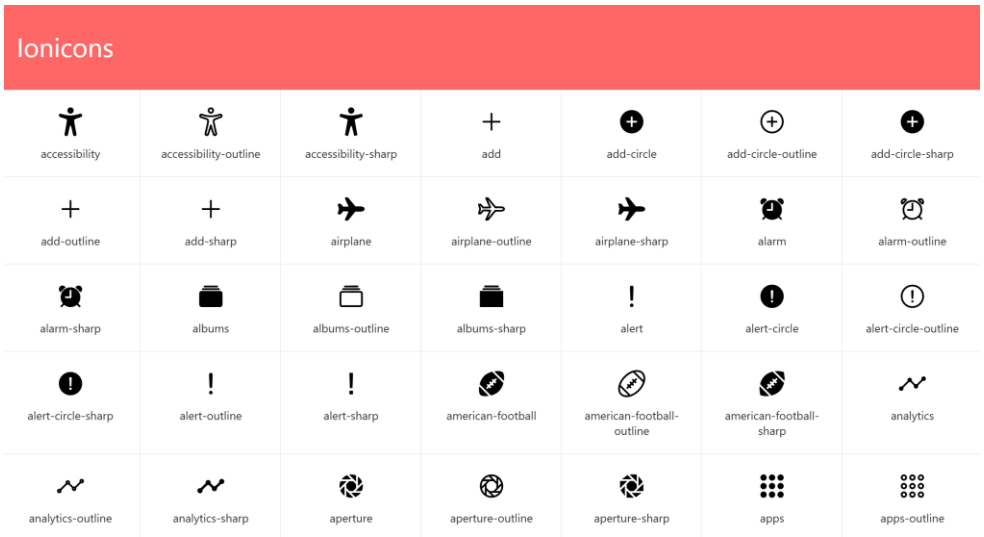


Figure 3.2: Parts of the Directory for Ionicons.

Lastly, for architecture design and database design, it will be developed for the project's broader view. Architectural design is used to provide the blueprint of the system for this application, while database design is used to specify the ways that data is stored, organized, connected, and accessed. These two designs will explain how the system works together, what will be used for the system, how data will be stored and handled, and others, which is to ensure the system works well together and data is structured and stored properly.

3.3.4 Development

In this project, an Android-compatible mobile application that provides resume creation and career advisory services to users is developed, which is named “Job Assistant”. The development lifecycle begins by working on tasks for each sprint, such as coding. Each sprint consists of specific tasks to be completed. Throughout this phase, the necessary software tools and technologies are used and integrated into this development process.

As planned in the planning phase, the development of this project will start at sprint 3, 4, and 5. The tools that were used to develop this application are Android Studio, Visual Studio Code, GitHub, and React Native. The database that will be developed in this project. The technology will be used to integrate as an AI functionality in this mobile application, which is Google Gemini AI Model, by using the API.

For the beginning of development sprints, GitHub will be used, and the project code folder will be connected to the repository. GitHub is used for a backup and recovery plan that avoids data loss and issues that cause bad consequences to the project, like laptop damage or memory loss. GitHub can store the code for this Android mobile application and allows the repository owner to clone the repository and continue working on it.

In sprint 3, the UI and database for this project will be developed. The initial prototype that was designed will be implemented and integrated exactly into the UI in the development lifecycle. The initial prototype will be implemented and developed. The libraries for colors, fonts, and designs, like vector icons, are used in this sprint to develop and support the UI. The SQLite database will also be developed in this sprint due to it should be done earlier to ensure features and functionalities developed in sprint 4 can function well.

For sprint 4, the features and functionalities of this application will then be implemented and integrated into the system by using the tools and technologies. The features and functionalities will connect to the UI, which will be combined into a complete and usable feature. The features that will be developed are resume creation, career advisory services, user account-related features, and admin account-related features. This Android mobile application is a role-based application that differentiates roles for normal users and admins. Google Gemini AI will be integrated by using its API key in this sprint. The API key will be saved in an env file, which will store the secret key. It will be connected to the features that require an AI assistant, such as resume creation and career advisory services.

Lastly, the development for sprint 5 will be completing the rest of the application and compiling all features together as a complete application. This sprint is mainly used for filling in the gaps that were missed and completing the

reports for this project. This sprint is the last sprint of this project, and all features for this application will be completed.

3.3.5 Testing

Testing is important to ensure good quality assurance. It will be involved in every sprint of development. Testing, such as unit testing and UAT testing, will be performed to ensure the features work as expected and achieve user satisfaction. Jest and React Native Testing Library (RNTL) will be used for unit testing.

3.3.5.1 Unit Testing

Unit testing is performed to test a unit of a feature once in a test. This testing is important to ensure the feature logic works correctly. For this project, unit testing will be performed mainly in sprint 4. This is because the tasks for sprint 4 are developing the features and functionalities of this Android mobile application, and even including the integration of an AI model into the code. Thus, Jest and RNTL are useful to test the unit tests for main features. A SQLite database created by using a Python script will then use pytest to test for only one case on the database.

3.3.5.2 User Acceptance Testing (UAT)

UAT is performed for end users to test the Android mobile application. It is tested to gain user acceptance and satisfaction with the Android mobile application. This testing is quite subjective to user preferences. UAT will also check whether the Android mobile application meets the project requirements and whether it is ready for deployment, which will validate the system's ability to work as expected in a real-world environment. UAT will be mainly tested in sprint 5 and tested with system testing together to check the performance and other non-functional requirements.

3.3.6 Deployment

Deployment will be conducted at the end of each development sprint for this project. The deployment phase will ensure that the developed features completed in the development sprints of this project are compatible with the

Android Operating System. This deployment phase involves conducting final verifications, sprint review, sprint retrospective, and others to ensure deployed features function as expected.

The sprint review is conducted to assess whether the deliverables of these 3 development sprints meet the planned milestones within the project timeline. In the sprint review, feedback will be gathered and analysed about the potential improvements for further updates. The sprint retrospective will be conducted to identify successful parts of the project, highlight the challenges encountered in this project, and propose potential actionable steps for future improvements gathered from the sprint review. This sprint retrospective fosters continuous learning and makes improvements. By completing all these activities and phases, the project will start to deploy.

In this project, the deployment of this Android Mobile Application will begin with virtual deployment. The dependencies and installation patch, such as Android APK, the libraries, and others for using this Android mobile application, will first be installed into the emulator provided by Android Studio. This emulator acts as the real device of Android smartphones.

This deployment on an emulator is to ensure the application is working well in Android environments and even in real life. With no errors or failures occurring in this deployment until the last sprint of the project, this states that the virtual deployment of this Android mobile application for resume builder and career advisory services is successful. Thus, it will continue to the real environment and device for Android, where the application is ready to be released to the end user.

3.4 Project Planning and Scheduling

The project planning and scheduling are completed together to separate tasks and break them down into sprints. From this planning and scheduling, the project timeline and expectations for the tasks will be developed. The work breakdown structure (WBS) and Gantt Chart for this project will be listed and planned for the whole project timeline accordingly.

3.4.1 Work Breakdown Structure (WBS)

0.0 Smart Mobile Application for Resume Building and Career Advisory Services

1.0 Sprint 1: Project Initialization 1

- 1.1 Project Title Registration
- 1.2 Project Background Study
- 1.3 Define Project Details
  - 1.3.1 Define Project Problem Statement
  - 1.3.2 Define Project Objectives
  - 1.3.3 Define Project Scope and Limitations
  - 1.3.4 Define Project Outline
- 1.4 Literature Review
  - 1.4.1 Define Purpose for Literature Review
  - 1.4.2 Review on Existing Systems
  - 1.4.3 Review on Software Methodology
  - 1.4.4 Review on Tools and Technologies

2.0 Sprint 2: Project Initialization 2

- 2.1 Project Planning
  - 2.1.1 Develop Project Workflow
  - 2.1.2 Develop Project Plan
    - 2.1.2.1 Construct WBS and Gantt Chart
    - 2.1.2.2 Select Tools and Technology
- 2.2 Project Specifications
  - 2.2.1 Define Requirements
    - 2.2.1.1 Define Functional Requirements
    - 2.2.1.2 Define Non-Functional Requirements
  - 2.2.2 Construct Use Case Diagram and Description
  - 2.2.3 Construct Initial Prototype
- 2.3 Project Design
  - 2.3.1 Construct Architectural Design
  - 2.3.2 Construct Database Design

3.0 Sprint 3: Development for UI and Database

- 3.1 UI Development
  - 3.1.1 Develop User Related Interfaces

3.1.2 Develop Resume Creation Interfaces

3.1.3 Develop Career Advisory Service Interfaces

3.1.4 Develop Admin Related Interfaces

3.1.5 Connect Navigations

3.2 Database Development

3.2.1 Develop Database Schema

3.2.2 Generate Database

3.3 Testing Execution

3.3.1 Execute Database test

3.4 Deployment Execution

3.4.1 Deploy developed UI and Database

3.4.2 Retrospective and Implementation

4.0 Sprint 4: Development for Features and Functionalities

4.1 Features Development

4.1.1 Develop Connections

4.1.1.1 Connects Database

4.1.1.2 Connects Session

4.1.2 Develop User Related Features

4.1.2.1 Develop Register, Login, and Logout Logic

4.1.2.2 Develop User Account Management Features

4.1.2.3 Develop Feedback Logic

4.1.3 Develop Resume Creation Features

4.1.3.1 Develop Main Resume Creation Screen Features

4.1.3.2 Develop Manual, Create Resume Features

4.1.3.3 Develop AI Support Resume Features

4.1.3.3.1 Develop Upload Resume Feature

4.1.3.3.2 Integrate AI Support

4.1.3.4 Develop Template and Customization Options

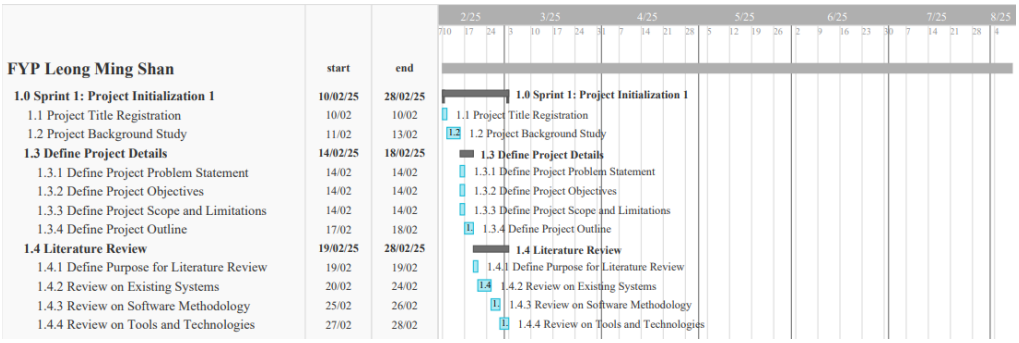
4.1.4 Develop Career Advisory Service Features

4.1.4.1 Develop Career Advice Sections

4.1.4.2 Develop Career History Logics

5.2.3 Execute UAT Test

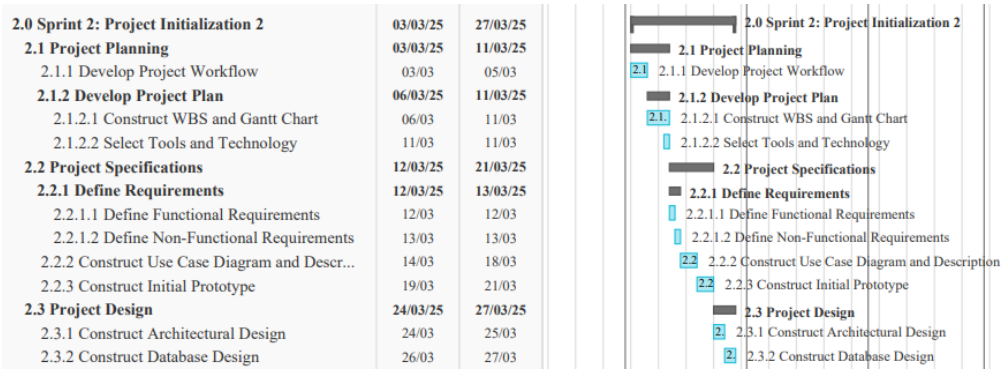
3.4.2 Gantt Chart



FYP Leong Ming Shan	start	end
1.0 Sprint 1: Project Initialization 1	**10/02/25**	**28/02/25**
1.1 Project Title Registration	10/02	10/02
1.2 Project Background Study	11/02	13/02
1.3 Define Project Details	**14/02/25**	**18/02/25**
1.3.1 Define Project Problem Statement	14/02	14/02
1.3.2 Define Project Objectives	14/02	14/02
1.3.3 Define Project Scope and Limitations	14/02	14/02
1.3.4 Define Project Outline	17/02	18/02
1.4 Literature Review	**19/02/25**	**28/02/25**
1.4.1 Define Purpose for Literature Review	19/02	19/02
1.4.2 Review on Existing Systems	20/02	24/02
1.4.3 Review on Software Methodology	25/02	26/02
1.4.4 Review on Tools and Technologies	27/02	28/02

Figure 3.3: Gantt Chart for Sprint 1.

The Gantt chart of sprint 1 shows that the sprint takes around one and a half months from 10 February to 28 February to complete the first part for project initialization. By introducing the project and undergoing a literature review, this project is progressively having a clear direction and insights into the ability, functionalities, and future of this Android mobile application.



2.0 Sprint 2: Project Initialization 2	**03/03/25**	**27/03/25**
2.1 Project Planning	**03/03/25**	**11/03/25**
2.1.1 Develop Project Workflow	03/03	05/03
2.1.2 Develop Project Plan	**06/03/25**	**11/03/25**
2.1.2.1 Construct WBS and Gantt Chart	06/03	11/03
2.1.2.2 Select Tools and Technology	11/03	11/03
2.2 Project Specifications	**12/03/25**	**21/03/25**
2.2.1 Define Requirements	**12/03/25**	**13/03/25**
2.2.1.1 Define Functional Requirements	12/03	12/03
2.2.1.2 Define Non-Functional Requirements	13/03	13/03
2.2.2 Construct Use Case Diagram and Descr...	14/03	18/03
2.2.3 Construct Initial Prototype	19/03	21/03
2.3 Project Design	**24/03/25**	**27/03/25**
2.3.1 Construct Architectural Design	24/03	25/03
2.3.2 Construct Database Design	26/03	27/03

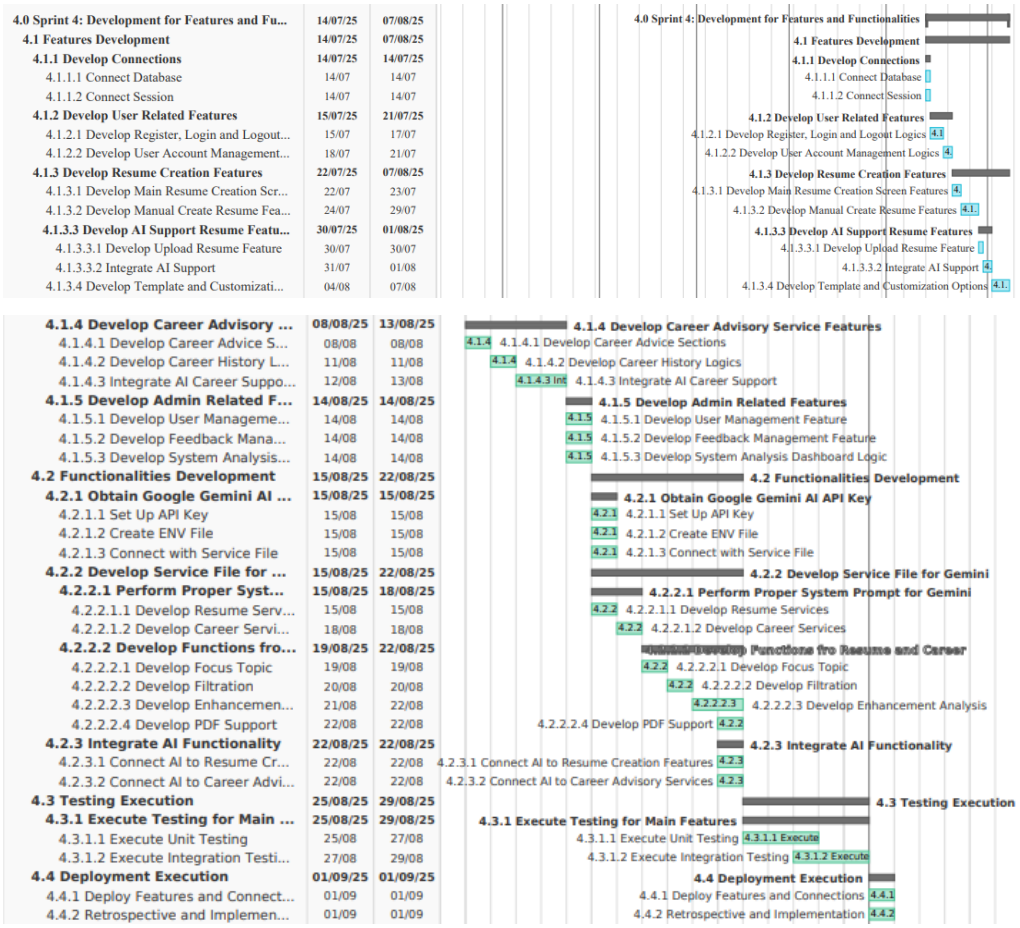
Figure 3.4: Gantt Chart for Sprint 2.

For sprint 2, the Gantt chart shows the start date is 3 March and the end date is 27 March. The project planning, project specifications, and project design will be done in this sprint. This sprint set a clear view of how the Android mobile application will work and behave. The timeline, tasks, and direction of this application will smooth the development process and complete it within the time range. In this sprint, the blueprint of this application will be shown.

3.0 Sprint 3: Development for UI and Database	**23/06/25**	**11/07/25**
3.1 UI Development	**23/06/25**	**04/07/25**
3.1.1 Develop User Related Interfaces	23/06	25/06
3.1.2 Develop Resume Creation Interfaces	25/06	27/06
3.1.3 Develop Career Advisory Services Inter...	30/06	01/07
3.1.4 Develop Admin Related Interfaces	02/07	03/07
3.1.5 Connect Navigations	04/07	04/07
3.2 Database Development	**07/07/25**	**08/07/25**
3.2.1 Develop Database Schema	07/07	07/07
3.2.2 Generate Database	08/07	08/07
3.3 Testing Execution	**09/07/25**	**09/07/25**
3.3.1 Execute Database Test	09/07	09/07
3.4 Deployment Execution	**10/07/25**	**11/07/25**
3.4.1 Deploy Developed UI and Database	10/07	10/07
3.4.2 Retrospective and Implementation	10/07	11/07

Figure 3.5: Gantt Chart for Sprint 3.

Figure 3.5 shows the duration of sprint 3. It will start on 23 June and last until 11 July, which takes around 3 weeks. This sprint is mainly focused on developing the UI and database for the Android mobile application. The UI for each interface and its components, such as modal views, vector icons, color themes, custom fonts, and other UI components, will be implemented. The database is constructed for tables such as users, feedback, resume, resume child tables, careerAdvice, and its child tables. An initial admin account is created from this database to ensure the admin role is assigned initially.



4.0 Sprint 4: Development for Features and Fu...	**14/07/25**	**07/08/25**
4.1 Features Development	**14/07/25**	**07/08/25**
4.1.1 Develop Connections	**14/07/25**	**14/07/25**
4.1.1.1 Connect Database	14/07	14/07
4.1.1.2 Connect Session	14/07	14/07
4.1.2 Develop User Related Features	**15/07/25**	**21/07/25**
4.1.2.1 Develop Register, Login and Logout...	15/07	17/07
4.1.2.2 Develop User Account Management...	18/07	21/07
4.1.3 Develop Resume Creation Features	**22/07/25**	**07/08/25**
4.1.3.1 Develop Main Resume Creation Scr...	22/07	23/07
4.1.3.2 Develop Manual Create Resume Fea...	24/07	29/07
4.1.3.3 Develop AI Support Resume Featu...	**30/07/25**	**01/08/25**
4.1.3.3.1 Develop Upload Resume Feature	30/07	30/07
4.1.3.3.2 Integrate AI Support	31/07	01/08
4.1.3.4 Develop Template and Customizati...	04/08	07/08
4.1.4 Develop Career Advisory ...	**08/08/25**	**13/08/25**
4.1.4.1 Develop Career Advice S...	08/08	08/08
4.1.4.2 Develop Career History L...	11/08	11/08
4.1.4.3 Integrate AI Career Suppo...	12/08	13/08
4.1.5 Develop Admin Related F...	**14/08/25**	**14/08/25**
4.1.5.1 Develop User Manageme...	14/08	14/08
4.1.5.2 Develop Feedback Mana...	14/08	14/08
4.1.5.3 Develop System Analysis...	14/08	14/08
4.2 Functionalities Development	**15/08/25**	**22/08/25**
4.2.1 Obtain Google Gemini AI ...	**15/08/25**	**15/08/25**
4.2.1.1 Set Up API Key	15/08	15/08
4.2.1.2 Create ENV File	15/08	15/08
4.2.1.3 Connect with Service File	15/08	15/08
4.2.2 Develop Service File for ...	**15/08/25**	**22/08/25**
4.2.2.1 Perform Proper Syst...	**15/08/25**	**18/08/25**
4.2.2.1.1 Develop Resume Serv...	15/08	15/08
4.2.2.1.2 Develop Career Servi...	18/08	18/08
4.2.2.2 Develop Functions fro...	**19/08/25**	**22/08/25**
4.2.2.2.1 Develop Focus Topic	19/08	19/08
4.2.2.2.2 Develop Filtration	20/08	20/08
4.2.2.2.3 Develop Enhancemen...	21/08	22/08
4.2.2.2.4 Develop PDF Support	22/08	22/08
4.2.3 Integrate AI Functionality	**22/08/25**	**22/08/25**
4.2.3.1 Connect AI to Resume Cr...	22/08	22/08
4.2.3.2 Connect AI to Career Advi...	22/08	22/08
4.3 Testing Execution	**25/08/25**	**29/08/25**
4.3.1 Execute Testing for Main ...	**25/08/25**	**29/08/25**
4.3.1.1 Execute Unit Testing	25/08	27/08
4.3.1.2 Execute Integration Testi...	27/08	29/08
4.4 Deployment Execution	**01/09/25**	**01/09/25**
4.4.1 Deploy Features and Connect...	01/09	01/09
4.4.2 Retrospective and Implemen...	01/09	01/09

Figure 3.6: Gantt Chart for Sprint 4.

Sprint 4 is a longer sprint that develops all the features and functionality logics. This sprint contains more than 30 tasks after breaking them down into smaller and specific tasks. The features and functionalities developed include resume creation, career advisory services, integration of Google Gemini AI API, and other tasks. Execution of testing and deployment is involved to connect the UI and database that were developed in sprint 3 together to form a complete system.

Figure 3.7: Gantt Chart for Sprint 5.

Lastly, sprint 5 is the end of the methodology planned for this project. It aims to complete and compile the Android mobile application with its development files, testing files, and others. The virtual machine environment will be ready with the proper Android Version and the necessary tools and technologies. The system test and UAT will be conducted to ensure the system is user-satisfied, objectives met, and proper system behaviour.

3.5 Technologies and Development Tools

Several necessary tools and technologies are essential to be used and integrated as functionality in this study. These enhance the efficiency and effectiveness of the development process and meet the project scope, requirements, and objectives.

3.5.1 Android Studio

Android Studio is crucial in developing Android mobile applications. In this project, Android Studio is mainly used for the emulators that are provided. This emulator allows for the display and connection with the UI and code developed.

The Android Studio emulator is a software tool that allows a mobile application to function like an actual Android device on a computer. Android

Studio provides various options of emulators for simulating screen sizes, Android versions, and hardware features. However, the core functionalities and UI components of the application developed will remain consistent, which ensures the application is responsive and works well on Android phones or tablets. Android Studio not only provides an emulator but also includes tools such as a visual layout editor and integrated debugging.

3.5.2 Visual Studio Code (VS Code)

In this project, Visual Studio Code is used as the IDE for coding and developing the features and functionalities of the Android mobile application. Visual Studio Code is efficient and effective in connecting the code with the emulator by using the terminal provided in Visual Studio Code and connecting the project to a GitHub repository.

There are many reasons for using VS Code in this project. VS Code is lightweight and has a fast processing speed, which is better than Android Studio. It provides various built-in features and allows installation of external extensions and resources in the built-in marketspace. It also supports multiple programming languages and frameworks used to develop an application, which the React Native Android mobile application will benefit from VS Code. VS Code also provides debugging features and Git Integration. Git Integration will connect the project to a GitHub repository, which will also be used in this project for version control. However, the main reason for using VS Code is that it is familiar and easy to get used to.

3.5.3 GitHub

GitHub is used and allows connections between Visual Studio Code and the GitHub repository in this project. The code folder for this Android mobile application will be stored inside the repository while syncing the changes and publishing the repository. GitHub allows Visual Studio Code to commit comments, sync changes, publish a branch or repository for public or private, and perform other functions. GitHub eases the tracking process on project progress, allows branch creation for different features, and serves as a backup recovery strategy for this project.

To use GitHub, one needs to start with the setup steps and installation of Git. Installation of Git will be the first step for setup, and it should be added to the system path. Next will be the configuration of Git on the device. The username and email for GitHub will need to be configured, and the identity set up. The next step will be the start of using GitHub, which will create a new repository or clone an existing repository for this project. In this project, the repository will be a private repository that cannot be viewed or cloned by the public. GitHub allows continuous development for this project regardless of using a different device. The setup and configuration for GitHub on different devices using the same username and email will allow cloning of the repository from GitHub.

3.5.4 React Native Framework

React Native is the framework used to develop the Android mobile application with resume building and career advisory services for this project. React Native is being chosen to use because I learned systematically in university and have experience developing a mobile application using this framework. Thus, I am more familiar with React Native.

Besides, React Native is compatible with Android. It provides various libraries to be integrated into an Android mobile application. For instance, the library for navigation, UI components, APIs, modules, and other features such as React Navigation, React Native Vector Icon, React Native Camera, Maps Geolocation, and others. This is useful for development features and UI, which will be used for the application. In this project, TypeScript will be the main programming language to code and develop this Android mobile application. JavaScript is less commonly used nowadays because TypeScript is more powerful. TypeScript allows for type safety, which can catch bugs at compile time and provide clean code.

3.5.5 SQLite Database

SQLite Database is used to develop the Android mobile application database. It is chosen because it is a relational database that can structure and organize data properly. It is commonly used nowadays, easy to use, and suitable for a React Native project, which does not require extra installation from different sources.

More details for the database used for this project will be described and explained in Chapter 5: Project Design, which includes the database design and data dictionary.

3.5.6 Jest with React Native Testing Library (RNTL)

Jest will be the testing framework used for the project testing. It also supports TypeScript React Native testing. Jest focuses on unit testing. In this project, Jest will test each part of the features and functionalities to ensure they are working correctly. React Native Testing Library (RNTL) is used in this project. It is a reliable testing library that is provided by React Native to test its application. It is usually used with Jest together to test unit testing. RNTL and Jest are limited for UI testing. Hence, UAT will be done by using the virtual machine environment provided by Android Studio, which is suitable for emulating the real device.

3.5.7 Google Gemini AI

Google Gemini AI will be integrated into this Android mobile application to support features like resume creation and providing career advisory services. Google Gemini is suitable for this project because it provides more free tier chances for developers to test and use for personal projects or learning materials. This allows the costs to be reduced and saves because the budget of this project is not enough to subscribe to a full functional AI model and get the API keys.

3.6 Summary

In conclusion, the work plan, methodology, and tools used for the development of an Android mobile application for resume building and career advisory services are discussed and selected.

In this study, the project adopts Scrum methodology due to its short iterations, and this project will be separated and broken down into 5 sprints. This project uses GitHub for backup and recovery plans, Android Studio for its emulator, and Visual Studio Code for coding the application. React Native will be the development framework chosen to develop this Android mobile application. SQLite Database is a relational database to store application data. Jest and RNTL will be the testing tools to test the application, and Google

Gemini AI is the AI technology that will be integrated into this application to assist users with career advice and resume creation.

CHAPTER 4

PROJECT SPECIFICATION

4.1 Introduction

In this chapter, the project specification, such as project requirements, use case diagram, and its description, and the initial prototype of the project will be discussed and included, which provides clear direction on functionality integration and a clear insight into the mobile application.

The project requirements are functional requirements and non-functional requirements, each of which contains different specifications. From project requirements, the expectations of the Android mobile application will be defined and enhance a clear understanding of how the application should behave. The use case diagram and its description will illustrate how the system works in detail. Followed by a database structure, it provides clear insights into the database of this application. Lastly, the initial prototype was developed to provide the primary version of the view of the system, which is only partially functional.

4.2 Functional Requirements

The functional requirements of this study were gathered by reviewing existing systems from the Play Store market. Besides, project scope, objectives, and problem statements support functional requirements development, which sets the mobile application goals. Table 4.1 shows the functional requirements of this study with different sections of features.

Table 4.1: Functional Requirements.

ID	Functional Requirements
FR01	The mobile application shall allow user to create resume.
FR02	The mobile application shall allow user to seek career advice.
FR03	The mobile application shall provide AI support resume creation.
FR04	The mobile application shall provide AI career support.

FR05	The mobile application shall provide manual resume creation to user.
FR06	The mobile application shall allow user to upload their resume.
FR07	The mobile application shall generate career advice for user.
FR08	The mobile application shall allow user to generate resume to export.
FR09	The mobile application shall provide customization options for user in resume creation.
FR10	The mobile application shall allow user to manage their user account.
FR11	The mobile application shall provide social media connection to user for further communication.
FR12	The mobile application shall allow user submit feedback.
FR13	The mobile application shall allow admin to manage user accounts.
FR14	The mobile application shall allow admin to manage user feedback.
FR15	The mobile application shall allow provide system analysis dashboard for admin.

4.3 Non-Functional Requirements

The non-functional requirements state the quality attributes and the constraint that a system must meet. The non-functional requirements can be gathered by analysing the project scope and objectives, which include understanding how the system works and expectations for this application. Non-functional requirements can be categorized into performance, usability, testability, reliability, efficiency, and others. Table 4.2 shows the non-functional requirements for this project.

Table 4.2: Non-Functional Requirements.

ID	NFR Category	Non-Functional Requirement Details
NFR1	Performance	The mobile application should have an acceptable time, such as not more than 1 minute, to complete the generation of a resume or career advice.

NFR2	Usability	The mobile application should have at least an 80% user satisfaction rate at 4 or above on a 5-point satisfaction scale.
NFR3	Compatibility	The mobile application should be compatible with the Android Operating System from version 6 to the latest Android version.
NFR4	Reliability	The mobile application should have below 1% error rate of for essential functionalities.

4.4 Use Case Diagram

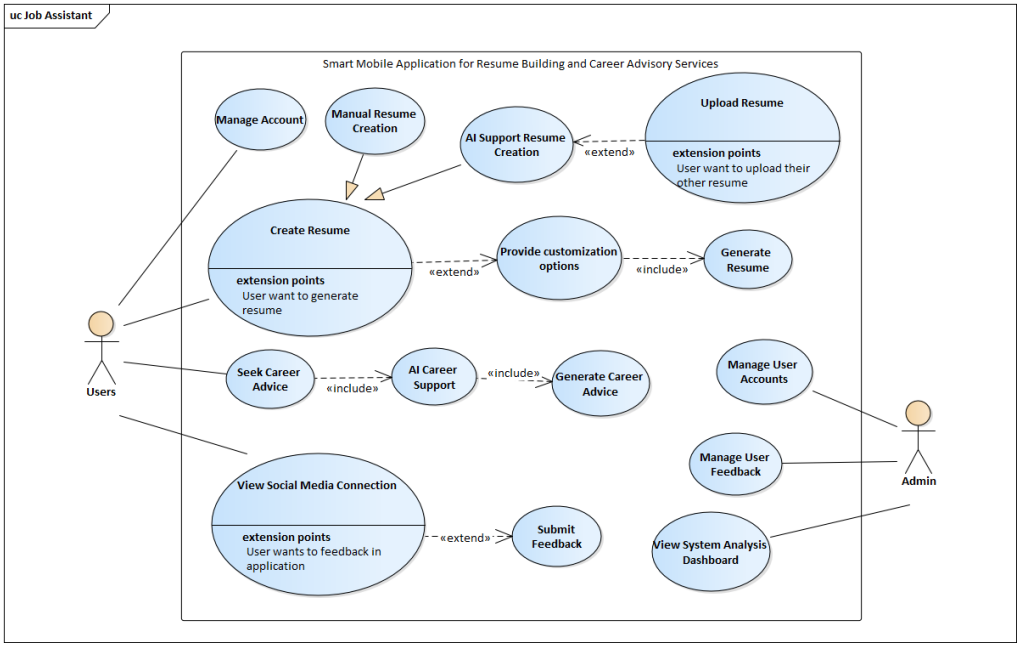


Figure 4.1: Use Case Diagram.

4.5 Use Case Descriptions

Table 4.3: Manage Account Use Case.

Use Case Name: Manage Account	ID: UC01	Importance Level: High
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: The user wants to manage their accounts to perform different operations.		

Brief Description: This use case describes the process of users managing and deal with their accounts.
Trigger: Users need to log in before using Android Mobile Application features.
Relationships: Association : User Include : N/A Extend : N/A Generalization: N/A
Normal Flow of Events: 1. The user can manage their user account by creating an account, logging in and logging out, editing their profile, resetting their password, and deleting their account. 2. The user needs to log in first before using the features of the Android mobile application. 3. The user will be navigated to the registration interface if the user does not have an account. 4. The user needs to create an account and will be automatically logged in once created successfully. 5. The user will be redirected to the home interface after logging in successfully. 6. The user is allowed to edit their account profile with user details such as profile image and name. 7. The user is allowed to reset and change their account password. 8. The user is allowed to delete their account for the termination of use of this Android mobile application. 9. The user can log out of this account after using this Android mobile application.
Sub-flows: N/A
Alternate/Exceptional Flows: N/A

Table 4.4: Create Resume Use Case.

<table>
<tr><td colspan="2">Use Case Name: Create Resume</td><td>ID:
UC02</td><td>Importance
Level: High</td></tr>
<tr><td colspan="2">Primary Actor: User</td><td colspan="2">Use Case Type: Detail, Essential</td></tr>
<tr><td colspan="4">Stakeholders and Interests: The user wants to create a resume.</td></tr>
<tr><td colspan="4">Brief Description: This use case describes the process of resume creation.</td></tr>
<tr><td colspan="4">Trigger: The user decides to create their resume.</td></tr>
<tr><td colspan="4">Relationships:
Association : User
Include : Generate Resume
Extend : Upload Resume, Provide Customization Options
Generalization: Manual Resume Creation, AI Support Resume Creation</td></tr>
<tr><td colspan="4">Normal Flow of Events:
1. The user wants to create their resume.
2. The Android mobile application provides manual resume creation and AI-supported resume creation to user.
3. The user can manually key in their resume data in the manual resume creation feature.
4. The user will then save the resume data, rename their resume name, and store it in the database.
5. The user can pass their created resume data to the AI support for resume creation. Go to Alternate Flow 1.
6. The Gemini AI API service will start to analyse, focus on the topic, filter unnecessary information, and enhance the resume data.
7. The user can view the enhanced resume and save it.
8. The resume data can be viewed in the resume list on the main resume features interface.
9. With a series of buttons to perform each function, the user can edit, AI-enhance, generate, and delete their resume.
10. The user can select their favourite customization options for their resume. Go to Alternate Flow 2.
11. The user will then generate the resume and export it to several file types such as PDF or PNG.</td></tr>
</table>

12. The user can share their resume. This will be the end of the resume creation process.
Sub-flows: N/A
Alternate/Exceptional Flows: Alternate Flow 1: Upload Resume 1. The user can also upload their other resume file to the AI support resume creation. 2. The Android mobile application accepts PDF file format only. Go to Normal Flow 6. Alternate Flow 2: Provide Customization Options 1. The Android mobile application provides customization options and templates for fonts, color, size, and others. 2. The user can view their resume styles in the live preview section. Go to Normal Flow 11.

Table 4.5: Seek Career Advice Use Case.

Use Case Name: Seek Career Advice	ID: UC03	Importance Level: High
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: The user wants to seek career advice.		
Brief Description: This use case describes the process of career advisory services performed for users.		
Trigger: The user wants to solve their career confusion.		
Relationships: Association : User Include : AI Career Support, Generate Career Advice Extend : N/A Generalization: N/A		
Normal Flow of Events: 1. The user can use the career advisory services to solve their career confusion. 2. The user should ask a career-related question.		

3. The Android mobile application will provide AI Career Support. 4. The Gemini AI API services will analyse and focus on the topic of the user input. 5. The AI support will then generate the career advice for user. 6. The response will differentiate with the user input to allow the user to have a better view. 7. The response will include different sections such as networking tips, action steps, and others. 8. The user could also view the history of career advice that had been made by pressing the history button. 9. The user can press and view the details of that specific history. 10. The user can edit or delete the history. This will be the end of the career advisory service process.
Sub-flows: N/A
Alternate/Exceptional Flows: N/A

Table 4.6: View Social Media Connection Use Case.

Use Case Name: View Social Media Connection	ID: UC04	Importance Level: High
Primary Actor: User	Use Case Type: Detail, Essential	
Stakeholders and Interests: The user wants to view the social media of the application or provide feedback.		
Brief Description: This use case describes the process of a user viewing the application's social media or submitting feedback.		
Trigger: The user presses the feedback button.		
Relationships: Association : User Include : N/A Extend : Submit Feedback Generalization: N/A		
Normal Flow of Events:		

1. The user clicks on the feedback button that appears in the bottom tab bar navigator. 2. The user will be redirected to the feedback interface. 3. The Android mobile application provides the social media channel connection for users to get further information. 4. The user is allowed to express their comment by filling in the feedback input area. Go to Alternate Flow 1. 5. The user can submit multiple feedback. 6. The user submits feedback and exits the interface.
Sub-flows: N/A
Alternate/Exceptional Flows: Alternate Flow 1: Submit Feedback 1. The user can input in the comments in the space provided. 2. The user can select different types of feedback to submit, such as a complaint, suggestion, or others. 3. The user needs to click the submit button to submit the feedback. 4. A modal alert will show to the user when feedback is successfully. Go to Normal Flow 4.

Table 4.7: Manage User Accounts Use Case.

Use Case Name: Manage User Accounts	ID: UC05	Importance Level: High
Primary Actor: Admin	Use Case Type: Detail, Essential	
Stakeholders and Interests: The admin wants to manage the user account.		
Brief Description: This use case describes the process of an admin managing and dealing with user accounts by using operations.		
Trigger: The admin redirects to the user management interface from the admin features main screen.		
Relationships: Association : Admin Include : N/A Extend : N/A		

Generalization: N/A
Normal Flow of Events: 1. The admin clicks on the user management button from the admin features main screen. 2. The admin features option from the profile screen will only be seen by admin role accounts. 3. The admin can manage the user accounts by a series of operations. 4. The admin can select a user by retrieving from the database. 5. The Android mobile application will show the selected username, ID, and their email. 6. The admin could assign the roles for the user account if more admins joined this application. 7. The admin can ban user accounts for a period that depends on the ban count. 8. The user needs to confirm the alert modal about their actions on the user account. 9. The admin can stop managing the user accounts anytime.
Sub-flows: N/A
Alternate/Exceptional Flows: N/A

Table 4.8: Manage User Feedback Use Case.

Use Case Name: Manage User Feedback	ID: UC06	Importance Level: High
Primary Actor: Admin	Use Case Type: Detail, Essential	
Stakeholders and Interests: The admin wants to manage the user feedback.		
Brief Description: This use case describes the process of an admin managing user feedback.		
Trigger: The admin decides to view the user feedback and marks the status of the feedback.		

Relationships: Association : Admin Include : N/A Extend : N/A Generalization: N/A
Normal Flow of Events: 1. The admin clicks on the user feedback management interface from the admin features main screen. 2. The admin can view all the feedback that has been submitted and stored in the database. 3. The Android mobile application will show the feedback and its details, such as feedback type, feedback ID, and others. 4. The admin can filter the feedback type to view the specific feedback. 5. The admin can view the feedback details by pressing the view button. 6. The admin can modify the status of each piece of the feedback. 7. The admin can delete the feedback based on the situation. 8. The Android mobile application will refresh the interface after actions made by the admin automatically. 9. The admin can stop managing the user feedback by quitting the interface.
Sub-flows: N/A
Alternate/Exceptional Flows: N/A

Table 4.9: View System Analysis Dashboard Case.

Use Case Name: View System Analysis Dashboard	ID: UC07	Importance Level: High
Primary Actor: Admin	Use Case Type: Detail, Essential	
Stakeholders and Interests: The admin wants to view the usage analysis of the system.		
Brief Description: This use case describes the process of an admin viewing the usage analysis of the system.		

Trigger: The user checks the usage analysis of the system.
Relationships: Association : Admin Include : N/A Extend : N/A Generalization: N/A
Normal Flow of Events: 1. The admin checks the usage analysis of the system by pressing the system analysis button from the admin features main interface. 2. The Android mobile application will retrieve the information data from the database. 3. The system analysis dashboard will show the usage analysis made by the user. 4. The admin can view the user count of this Android mobile application. 5. The admin can view the usage count of resume creation features and career advisory features used by users. 6. The dashboard will show several graphs and charts for the admin to view the analysis. 7. The admin can press the refresh button to check the latest data. 8. The admin can quit from this interface and automatically navigate to the admin features main interface.
Sub-flows: N/A
Alternate/Exceptional Flows: N/A

4.6 Initial Prototype

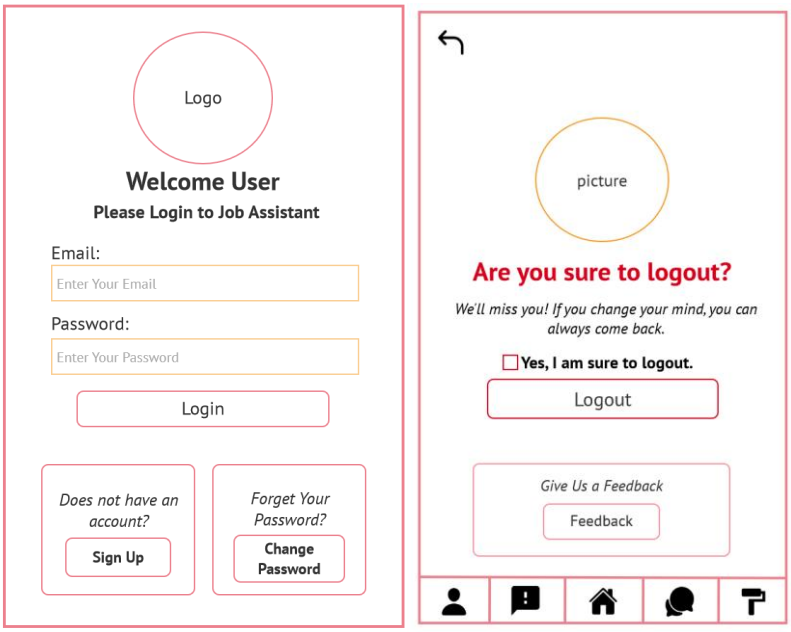


Figure 4.2: User Login and Logout Process.

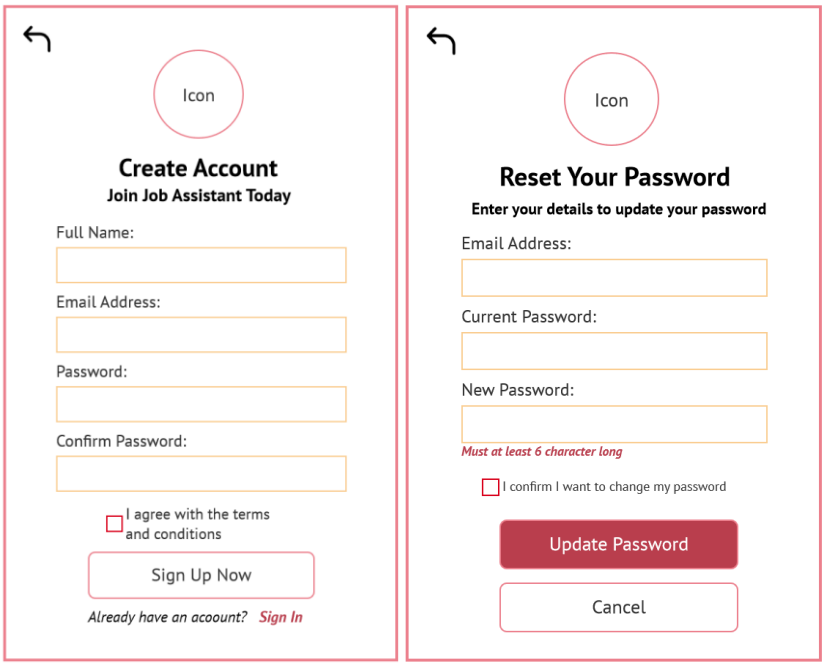


Figure 4.3: User Sign Up and Reset Password Process.

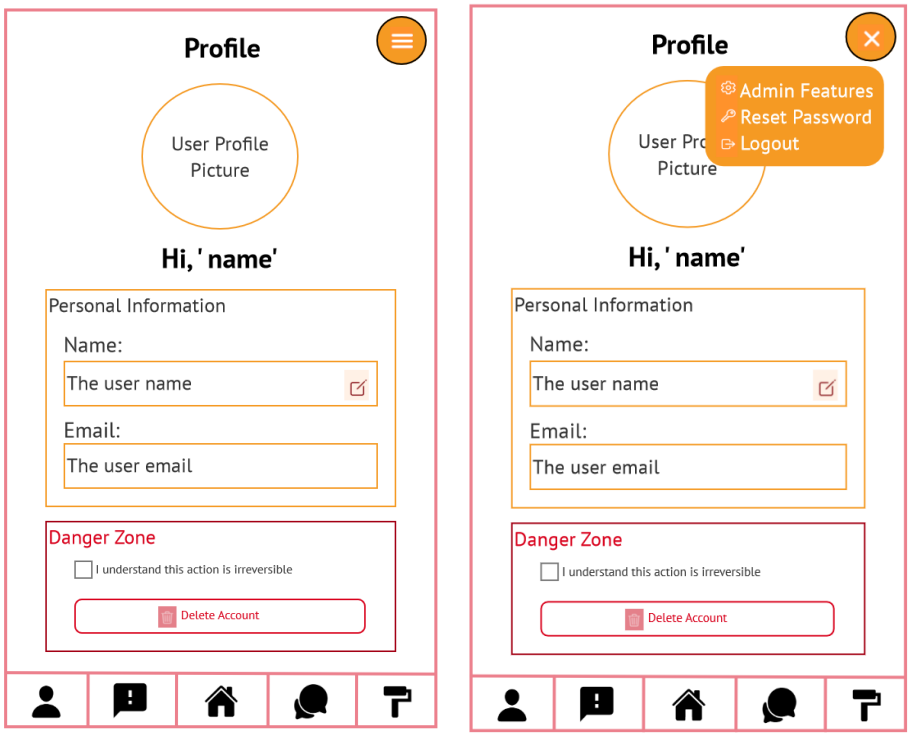


Figure 4.4: User Profile Initial Prototype Screen.

In this project, the process will start from the user creating an account and its related features. If the user does not have an account, they will need to sign up to create their account. While the user successfully signs up their account, the application will automatically pass through the created account and log in the user. The user will then navigate and redirect to the home screen. If the user has an account, the user can directly log in with their account. The reset password feature is provided for users who have forgotten their account password in order to modify the password to a new one.

Users can log out of their current account. This function is located in the profile screen, which appears as a floating button with multiple options in the top right corner of the screen. On this profile screen, users can upload their profile image and edit their name. Additionally, the "danger zone" for deleting accounts is available, but users need to be aware of this action. The floating button will also display options for resetting the password and logging out for regular users, while admin features are shown only to users with the admin role.

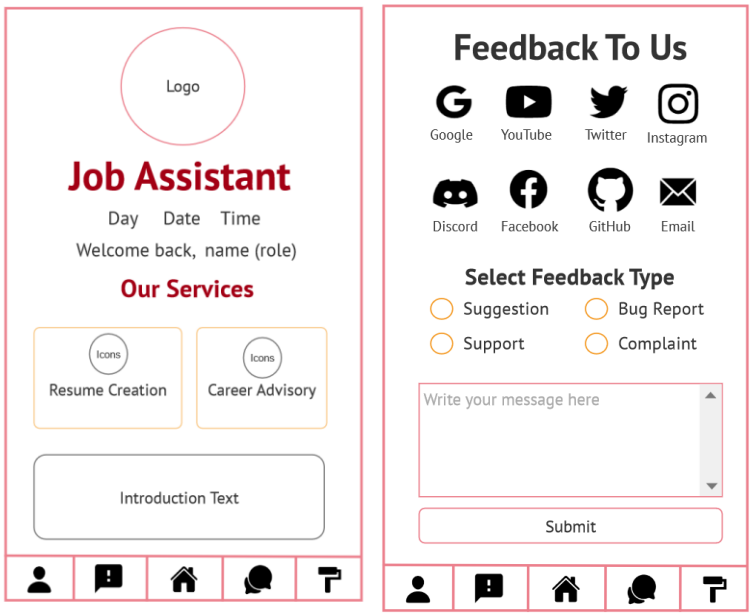


Figure 4.5: Home and Feedback Initial Prototype Screen.

Figure 4.5 shows the home screen and feedback screen. The bottom tab bar navigator will provide easy navigation to the user. From left to right, each button is the profile screen, the feedback screen, the home screen, the career advisory services screen, and the resume creation screen. The main features of this Android mobile application, which are resume creation and career advisory, will be shown in the home screen, and the buttons are pressable, which will navigate the user to the respective feature screen. In the feedback screen, several icons are provided and are pressable by the user, which will redirect the user to the respective URL link. The feedback section within the screen allows user to submit their feedback with different types such as suggestions, support, bug reports, and complaints.

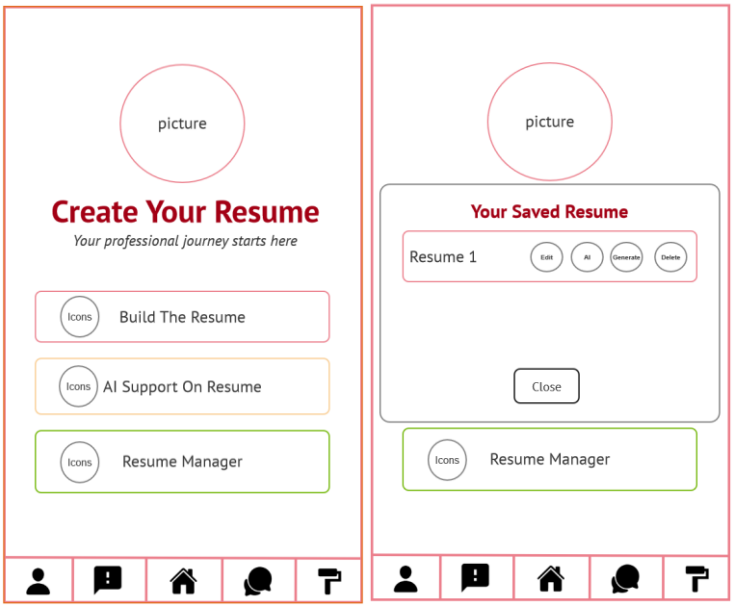


Figure 4.6: Resume Creation Initial Prototype Main Screen.

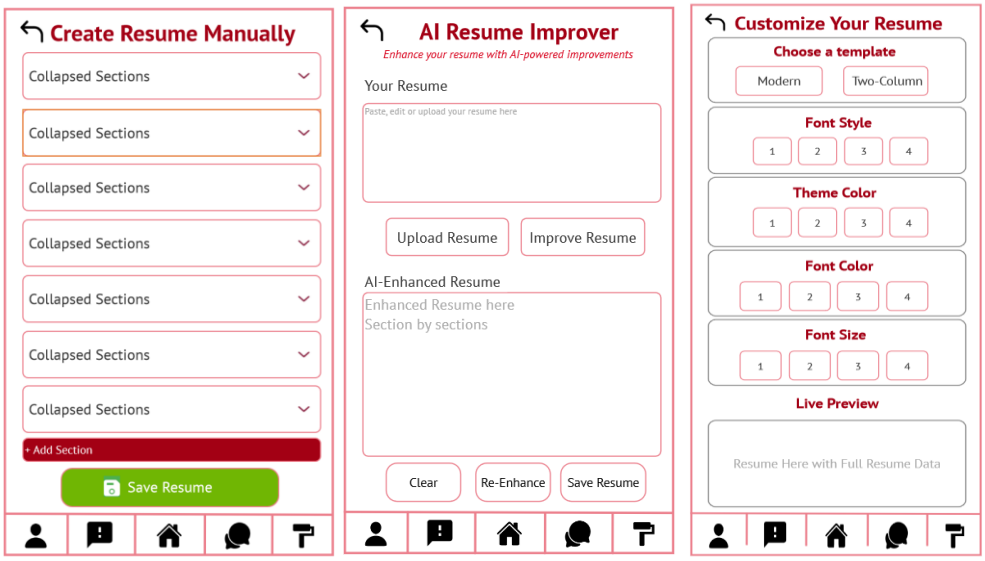


Figure 4.7: Resume Creation Features.

One of the main features of this Android mobile application is resume creation. While the user navigates into the resume creation screen, the main screen for this feature will be shown, and three buttons are provided for the user to manually create their resume, use AI support to enhance or create their resume, and use a resume manager to retrieve user resume records. The resume manager will display all the resumes that were created by the logged-in user. By pressing the resume name, the resume manager will show the respective resume details. The buttons beside the resume name will perform different operations,

such as edit the resume, use AI to enhance, generate the resume by selecting the customization options, or delete the resume.

The manual resume creation screen allows user to fill in their resume information section by section, and users are able to leave the section blank, add more sections, and add or delete the items in the sections. User will then save the resume and rename the default resume name to their own resume name. The saved resume data will be stored in the database, and perform CRUD operations by using the resume manager functions.

The AI support resume creation allows user to upload their other resume in PDF format or retrieve the created resume details from the resume manager. By pressing the improve resume button, the Gemini AI service that has been developed will start to analyse, focus on the topic, filter unnecessary information, enhance, and elaborate on more details in the resume. The user can re-enhance the resume if not satisfied with the AI response or save the enhanced resume if satisfied.

The customization options and templates can be selected while the user wants to generate their resume. The user will then navigate to the template selection screen, passing through the respective resume data. Several templates and customization options, such as theme color, font color, font type, and font size, are provided to the user. The live preview section, which is below the options, will display the selected options and the resume data. By scrolling down, below the live preview screen, there are several buttons provided for the user to generate their resume and export it in different file formats such as PDF or PNG. User is also able to share their resume after it is generated.

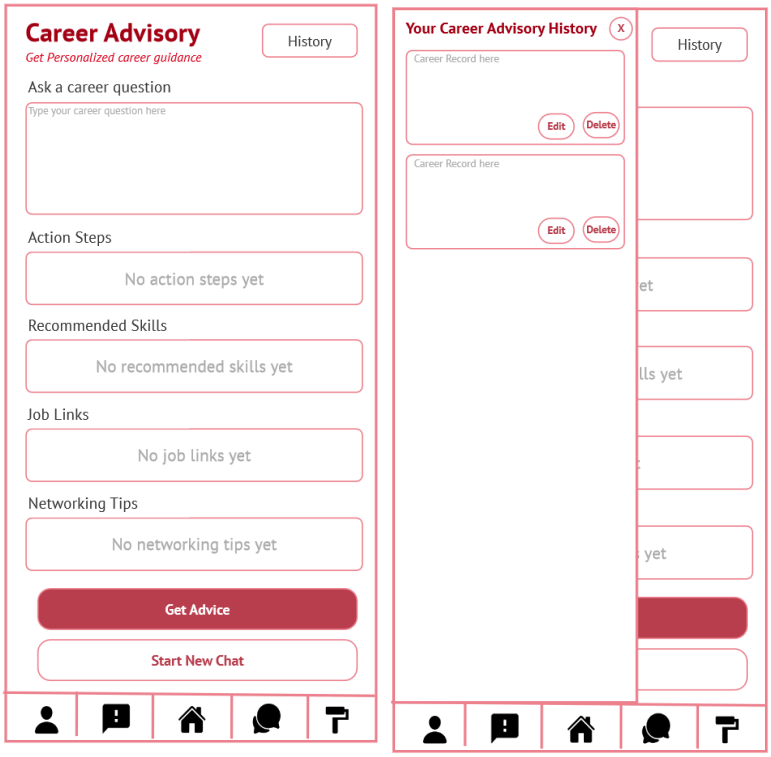


Figure 4.8: Career Advisory Services Initial Prototype Screen.

For the career advisory services, the screen is simple and clear for the user. The user can input the career-related questions in the input space area and press the get advice button. The Gemini AI service will generate career advice for users in different sections, which are action steps, recommended skills, job links, and networking tips. These sections allow user to have better opportunities for career advancement and improve their competitive strengths. Career history function is provided for the user in order to allow the user to view their generated career advice or ask more about the respective career question by pressing the edit button. User is allowed to delete their career history records.

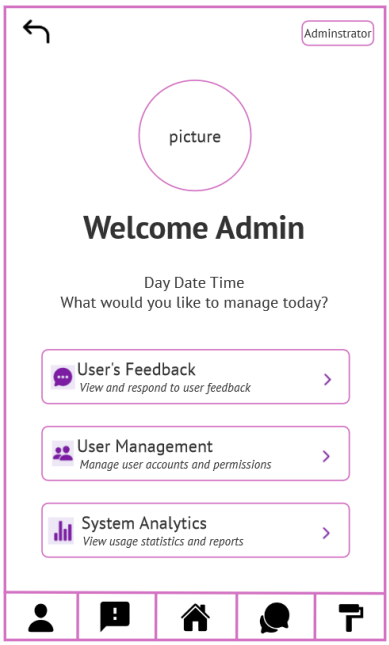


Figure 4.9: Admin Related Main Screen Initial Prototype

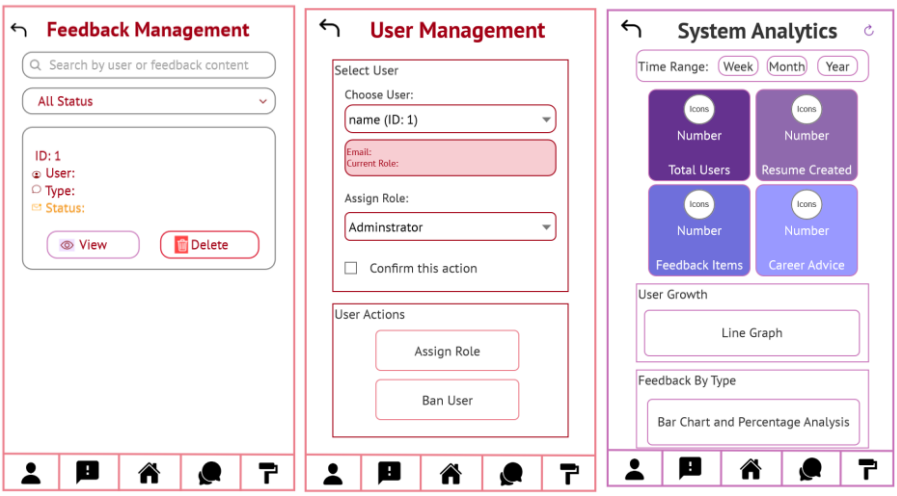


Figure 4.10: Admin Related Initial Prototype Screen.

This Android mobile application is a role-based application; there will be a normal user role and an admin role. Admin has their own related features such as feedback management, user management, and system analytics dashboard. Feedback management involves viewing the feedback submitted by users and reading the details. It also allows admins to mark the feedback with different statuses, view details, and delete the feedback. User management allows admins to select the respective user to grant the admin role or ban the user for a time. These actions will be confirmed by the admin to continue

performing the functions. Lastly, the system analytics dashboard will display the system usage of this application. The count of users, resumes created, feedback submitted, and usage of career advice will be shown. A line graph will display the user growth, and a bar chart with the percentage shows the feedback type submitted by users in this application. These initial prototype diagrams will build a complete primary picture for this smart mobile application for resume building and career advisory services.

4.7 Summary

In this chapter, the project specifications are clearly stated and defined. Project requirements, use case diagrams and their descriptions, database structure, and initial prototype for this Android mobile application are explained. There are 15 functional requirements and 4 non-functional requirements for this project. The use case diagram and its descriptions will describe and complement details on the main features of this Android mobile application. The initial prototype describes the designs and the features of this application.

These project specifications, such as requirements, use case and its description, and initial prototype designs, specify the primary design and idea for the application. Based on these specifications, the development could progress smoothly as expected and designed.

CHAPTER 5

PROJECT DESIGN

5.1 Introduction

In this chapter, the project's system architecture and database design will be described in detail, along with an explanation of the usage of features and functionalities in this project. The system architecture design will provide a clear overview of how different parts will interact with each other. The database design will highlight how data is structured, stored, and managed in this application to support the features and functionalities such as query operations.

5.2 System Architecture Design

These system architecture designs explained the behaviour of this application work and how it connects to the backend server, database, and AI services to perform the features of the application.

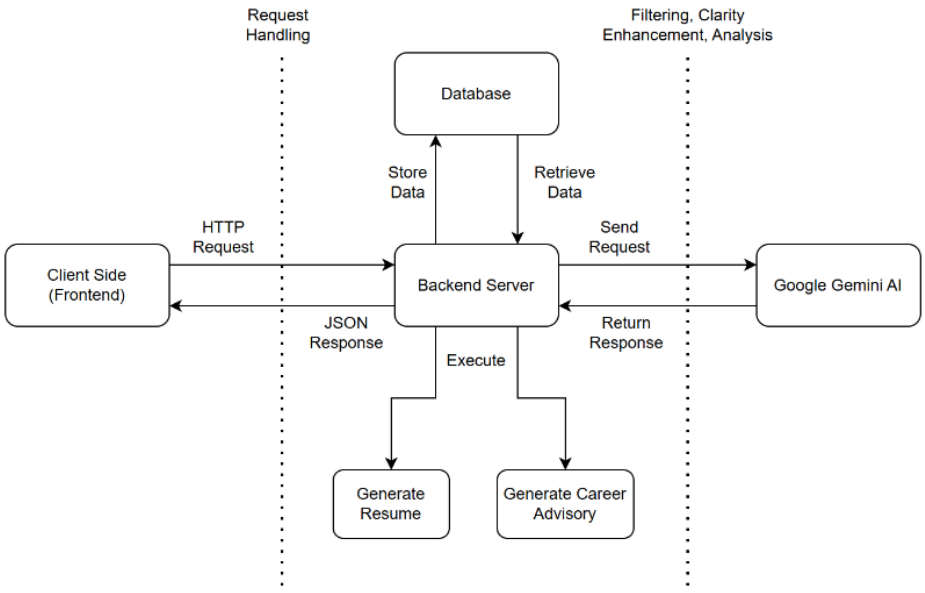


Figure 5.1: Overview System Architecture Design.

The client side, which the user used to run the application, will send the request to the backend server, and the backend server will return the response to the user. This occurs when users deal with the resume creation and career advisory services features. The backend server will connect to the database and perform query operations such as create, read, update, and delete (CRUD) operations. The backend server will also connect to Google Gemini AI, which is the AI service integrated into this application. This AI will perform the analysis, topic focus, filtering, enhancement, and elaboration services for resume creation and career advisory services.

With the features and functionality, the backend server will create the resume and generate the career advice for the user after being requested. For example, a user wants to create a resume using this application. The user uses the upload PDF resume function for AI enhancement. In this situation, the client side will send the request to the backend server, and the backend server will connect and send a request to Google Gemini AI to perform the resume enhancement. After that, the user is satisfied with the enhancement and saves the resume. This allows the backend server to store the data in the database. Once the user wants to generate the resume, the backend server will then perform the resume generation and export the resume for the user.

5.3 Database Design

The database design includes the entity-relationship diagram (ERD) and its data dictionary. These two components clearly state the database schema and its relationship for this application. The ERD represents the database structure visually, which includes the entities, attributes, and relationships among the entities, which provide a clear view of the data connections. The data dictionary will complement the ERD by providing detailed descriptions of each component.

5.3.1 Entity-Relationship Diagram (ERD)

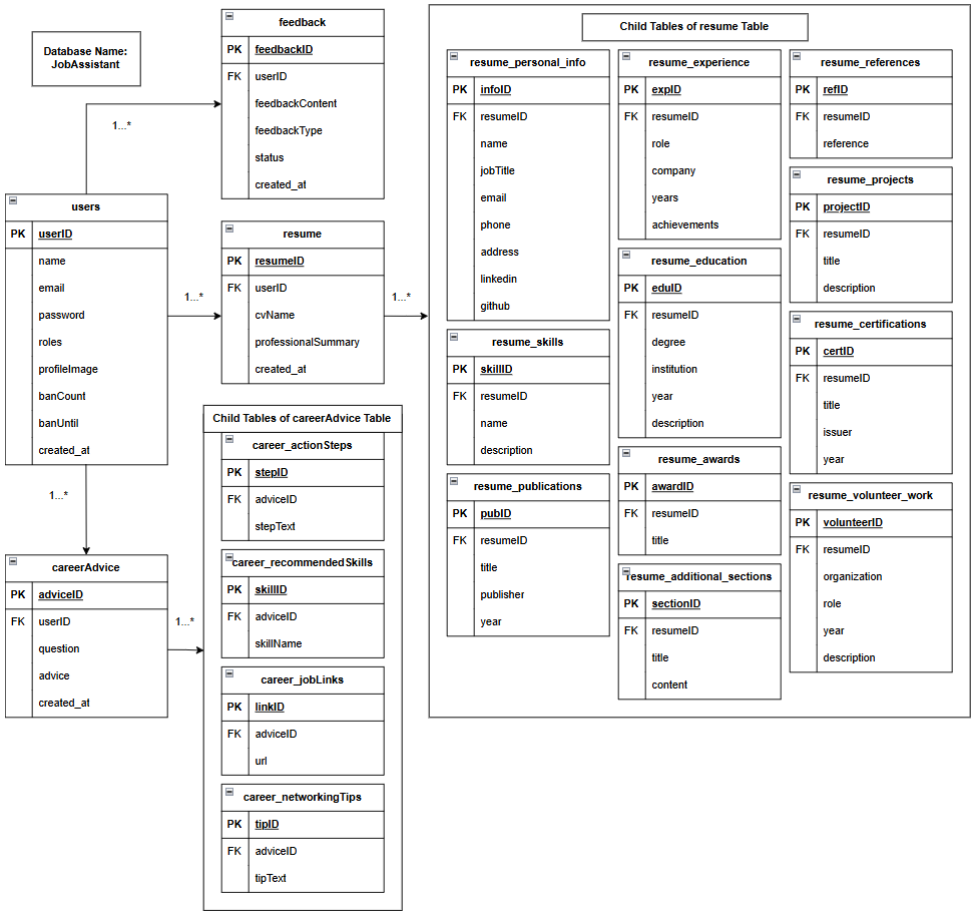


Figure 5.2: ERD diagram.

In this ERD diagram, it clearly shows that the database of this application is called ‘JobAssistant’ and it has 4 main entities, which are users, feedback, resume, and careerAdvice. The resume and careerAdvice table have their own child tables, respectively. The child tables for resume and careerAdvice define the sections for resume creation and career advisory service features, which explain how the resume data and career advice will behave in the database.

In this database, a user can have multiple feedback, multiple resumes created, and multiple career advice generated, which shows that the relationship among users and other tables is a one-to-many relationship. Each table has its own primary key and foreign key, which relate to the users table or its parent table.

Each table in this database has its specific functions to perform with the application. For instance, the resume table and its child tables deal with the

resume creation feature in the application. There will be database query operations for all tables due to the usage of this application. Figure 5.3 shows the code snippet for example query operations that deal with the database in this application.

```
export const banUser = async (db: SQLiteDatabase, userID: number) => {
  const res = await db.executeSql(
    `SELECT roles, banCount FROM users WHERE userID = ?`,
    [userID]
  );

  if (res[0].rows.length === 0) throw new Error("User not found");
  const user = res[0].rows.item(0);

  if (user.roles?.toLowerCase() === "admin") {
    throw new Error("Admin accounts cannot be banned.");
  }
  let banCount = user.banCount || 0;
  banCount += 1;

  let banDurationDays = 0;
  if (banCount === 1) banDurationDays = 1;
  else if (banCount === 2) banDurationDays = 3;
  else if (banCount === 3) banDurationDays = 7;

  if (banCount >= 4) {
    await clearUserData(db, userID);
    await deleteUserFromDB(db, userID);
    return { deleted: true, banCount };
  }

  const banUntil = new Date();
  banUntil.setDate(banUntil.getDate() + banDurationDays);
  await db.executeSql(
    `UPDATE users SET banCount = ?, banUntil = ? WHERE userID = ?`,
    [banCount, banUntil.toISOString(), userID]
  );

  return { deleted: false, banCount, banUntil };
};
```

Figure 5.3: Code Snippet for Database Query Operations.

5.3.2 Data Dictionary

The data dictionary for the database of this Android mobile application will serve as an important reference that outlines the structure and function of each database component. It clearly defines the key database tables, which are users, feedback, resume and careerAdvice with their respective attributes, data types, and relationships.

This data dictionary not only ensures consistency and accuracy in data management but also enhances the overall readability and maintainability of the database. By providing detailed descriptions of each table and attributes, the data dictionary supports a better understanding of the underlying data structure and facilitates more efficient application development and troubleshooting for development or testing.

Table 5.1: Data dictionary for the users table.

Attribute	Data Type	Description	Constraint
userID (PK)	integer	The ID that represents each user.	Auto-increment
name	text	The name of the user.	Not null
email	text	The email of the user.	Not null
password	text	The password of the user account.	Not null
roles	text	The role that differentiates the admin and normal user, which have different access permissions to certain features.	Not null
profileImage	text	The profile image of the user.	-
banCount	integer	The count of being banned for that respective user account.	Default 0
banUntil	timestamp	The ban time for different banCount.	-
created_at	timestamp	The user account creation time.	Default current timestamp

Table 5.2: Data dictionary for feedback table.

Attribute	Data Type	Description	Constraint
feedbackID (PK)	integer	The ID for each feedback.	Auto-increment

userID (FK)	integer	The foreign key from the users table. Connects the feedback submitted to the respective user.	Not null, references and delete on cascade
feedbackContent	text	The content of the respective feedback.	Not null
feedbackType	text	The type of feedback includes suggestions, support, bug reports, and complaints.	Not null
status	text	The status of the feedback, such as received, processing, or closed.	Not null
created_at	timestamp	The feedback created time.	Default current timestamp

Table 5.3: Data dictionary for resume table.

Attribute	Data Type	Description	Constraint
resumeID (PK)	integer	The ID for each resume is created.	Auto-increment
userID (FK)	integer	The foreign key from the users table. Connects the resume created to the respective user.	Not null, references and delete on cascade
cvName	text	The name for each resume.	Not null
professional Summary	text	The professional summary for each resume.	-
created_at	timestamp	The resume created time.	Default current timestamp

Table 5.4: Data dictionary for careerAdvice table.

Attribute	Data Type	Description	Constraint
adviceID (PK)	integer	The ID for each career advice is generated.	Auto-increment
userID (FK)	integer	The foreign key from the users table. Connects the career advice generated for the respective user.	Not null, references and delete on cascade
question	text	The career question asked by the user.	Not null
advice	text	The career advice is generated by Gemini AI services.	Not null
created_at	timestamp	The career advice generated time.	Default current timestamp

The child tables for resume and careerAdvice are related to their respective parent table. These child tables deal with their specific function. The child tables for resume contain the resume data for each section, such as personal information, skills, awards, and others, which allows the resume data to pass through each database query operation. While the child tables for careerAdvice contain action steps, recommended skills, and others for generating advice to the user based on these sections.

For resume child tables, their primary key will be each table's ID, such as infoID for resume_personal_info, skillID for resume_skills, and others. The foreign key will use the same constraint as the resumeID, the same constraint as other foreign keys, such as references and delete on cascade. Each of the resume child tables will then contain its own attributes, such as title, description, year, and other attributes.

For careerAdvice child tables, they are similar to resume child tables, which share the same foreign key, adviceID. The primary key of each careerAdvice child table will be their respective table ID, such as stepID, skillID, linkID, and tipID. Each of the table attributes is specific based on the table usage, such as career_actionSteps consists of its primary key, foreign key which

connects it to its parent table, and the ‘stepText’ attribute which stores the action step advice generated by AI.

5.4 Summary

In conclusion, this chapter has provided a clear and structured overview of the system architecture and database design for this application. The system architecture design describes different components, such as the client, backend server, database, and AI services, that interact with each other to support the features of this application. It ensures smooth communication and interaction between the frontend and backend, and leverages Google Gemini AI for intelligent and professional resume enhancement and career advisory services.

The database design is supported by the entity-relationship design (ERD) and data dictionary. This database design shows that the data is organized through well-defined relationships and attributes, and its data flow. The data dictionary complements the details and descriptions for ERD. Each table's attributes for the 4 main tables are stated with their data type, descriptions, and constraints, which are important for the database to accept correct data without errors occurring.

Together, the system architecture and database play a crucial role that serves as a foundational guide for development. This also ensures that the application is scalable, robust, and aligned with the project's objectives.

CHAPTER 6

PROJECT DEVELOPMENT

6.1 Introduction

In this chapter, the project development details and process are stated clearly. This chapter will focus on the development phase of the project, which involved sprints 3 and 4 that developed the UI, database, features, and functionalities. The project setup and development process are described for this Android mobile application. Screenshots that include the interfaces, code snippets, instructions, and others are attached to complement more in detail and provide a visual representation of each section in this chapter.

6.2 Project Set Up

In this project, several setup processes are required before starting to develop the application. React Native, Android Studio emulator, SQLite database, and Gemini AI API will need to be set up, as these tools and technologies are not defaulted to be included in the device, which requires installation dependencies or libraries. These build processes are crucial before starting the development process and contribute to the success of this project.

6.2.1 React Native Set Up

To set up React Native is easy, but the process takes some time for the installation of dependencies. To start and initialize a React Naive project, the command ‘npm react-native init ProjectName’ will be used. This command will create a new directory for the project and install all the dependencies. Besides, this command will also automatically generate the core files for the React Native project, such as app files, configuration files, such as package.json, babel.config, and others, for Android mobile applications. The dependencies and versions are defined in the package.json for React Native. To start React Native, the command ‘npm react-native run-android' is used to start the Android mobile application. It will build the Android mobile application and launch the emulator.

```
C:\Users\Linn\FYP2025>npx react-native run-android
warn Package react-native-sqlite-storage contains invalid configuration: "dependency.platforms.ios.project" is not allowed. Please
verify it's properly linked using "npx react-native config" command and contact the package maintainers about this.
* daemon not running; starting now at tcp:5037
* daemon started successfully
info Launching emulator...
info Successfully launched emulator.
info Installing the app...
Starting a Gradle Daemon, 1 incompatible and 1 stopped Daemons could not be reused, use --status for details

> Configure project :react-native-reanimated
Android gradle plugin: 8.9.2
Gradle: 8.14.1

> Task :app:installDebug
Installing APK 'app-debug.apk' on 'Pixel_4_API_30(AVD) - 11' for :app:debug
Installed on 1 device.

[Incubating] Problems report is available at: file:///C:/Users/Linn/FYP2025/android/build/reports/problems/problems-report.html

Deprecated Gradle features were used in this build, making it incompatible with Gradle 9.0.

You can use '--warning-mode all' to show the individual deprecation warnings and determine if they come from your own scripts or pl
ugins.

For more on this, please refer to https://docs.gradle.org/8.14.1/userguide/command_line_interface.html#sec:command_line_warnings in
 the Gradle documentation.

BUILD SUCCESSFUL in 1m 3s
531 actionable tasks: 34 executed, 497 up-to-date
info Connecting to the development server...
8081
info Starting the app on "emulator-5554"...
Starting: Intent { act=android.intent.action.MAIN cat=[android.intent.category.LAUNCHER] cmp=com.fyp2025/.MainActivity }
```

Figure 6.1: React Native Start Up.

```
{
  "name": "FYP2025",
  "version": "0.0.1",
  "private": true,
  ▷Debug
  "scripts": {
    "android": "react-native run-android",
    "ios": "react-native run-ios",
    "lint": "eslint .",
    "start": "react-native start",
    "test": "jest --watchAll"
  },
  "dependencies": {
    "@react-native-async-storage/async-storage": "^2.2.0",
    "@react-native-community/checkbox": "^0.5.20",
    "@react-native-community/datetimepicker": "^8.4.4",
    "@react-native-community/slider": "^5.0.1",
    "@react-native-documents/picker": "^10.1.5",
    "@react-native-masked-view/masked-view": "^0.3.2",
    "@react-native-picker/picker": "^2.11.1",
    "@react-native/new-app-screen": "0.80.1",
    "@react-navigation/bottom-tabs": "^7.4.2",
    "@react-navigation/drawer": "^7.5.3",
    "@react-navigation/native": "^7.1.17",
    "@react-navigation/native-stack": "^7.3.24",
    "@react-navigation/stack": "^7.4.2",
    "@types/react-native-sqlite-storage": "^6.0.5",
    "axios": "^1.11.0",
    "dotenv": "^17.2.1",
    "lucide-react-native": "^0.542.0",
    "pdf-lib": "^1.17.1",
    "pdf-parse": "^1.1.1",
    "pdfjs-dist": "^5.4.54",
    "react": "19.1.0",
    "react-native": "0.80.1",
    "react-native-chart-kit": "^6.12.0",
```

Figure 6.2: Code Snippet React Native package.json.

6.2.2 SQLite Database Set Up

With the installation of React Native properly, the SQLite database can be easily set up. The SQLite database will only require one command to install it, which is ‘npm install react-native-sqlite-storage’. It does not require an extra download and installation from the website or other sources.

```
import sqlite3

db = sqlite3.connect('jobAssistantFinal1.sqlite')
db.execute('PRAGMA foreign_keys = ON')

# Drop existing tables
tables = [···
]
for t in tables:···

# ---------------- USERS ----------------
db.execute('''
    CREATE TABLE users(
        userID INTEGER PRIMARY KEY AUTOINCREMENT,
        name TEXT NOT NULL,
        email TEXT NOT NULL UNIQUE,
        password TEXT NOT NULL,
        roles TEXT NOT NULL,
        profileImage TEXT,
        banCount INTEGER DEFAULT 0,
        banUntil TIMESTAMP,
        created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP
    )
''')
```

Figure 6.3: SQLite Database Schema Code Snippet in Python File.

The SQLite database schema will be created by using a Python script. In this script, the table information, such as table name, attributes, data type, constraints, and others, will be defined. Each table to be created uses the db.execute function, while inserting and creating the primary data will use the cursor.execute function. To generate the SQLite file for the database, the command ‘python databaseFileName.py’ will be executed, and the databaseName.sqlite file will be generated. This SQLite file will need to relocate its file directory into android\app\src\main\assets\databaseName.sqlite that ensures the database is implemented into the react native Android mobile application.

6.2.3 Gemini AI API Set Up

The AI model integrated in this application is Google Gemini AI. Gemini AI provides responsive AI support quickly, which is suitable for resume creation and career advisory services features for this application. With free limits, Gemini AI is also cost-effective for the project.

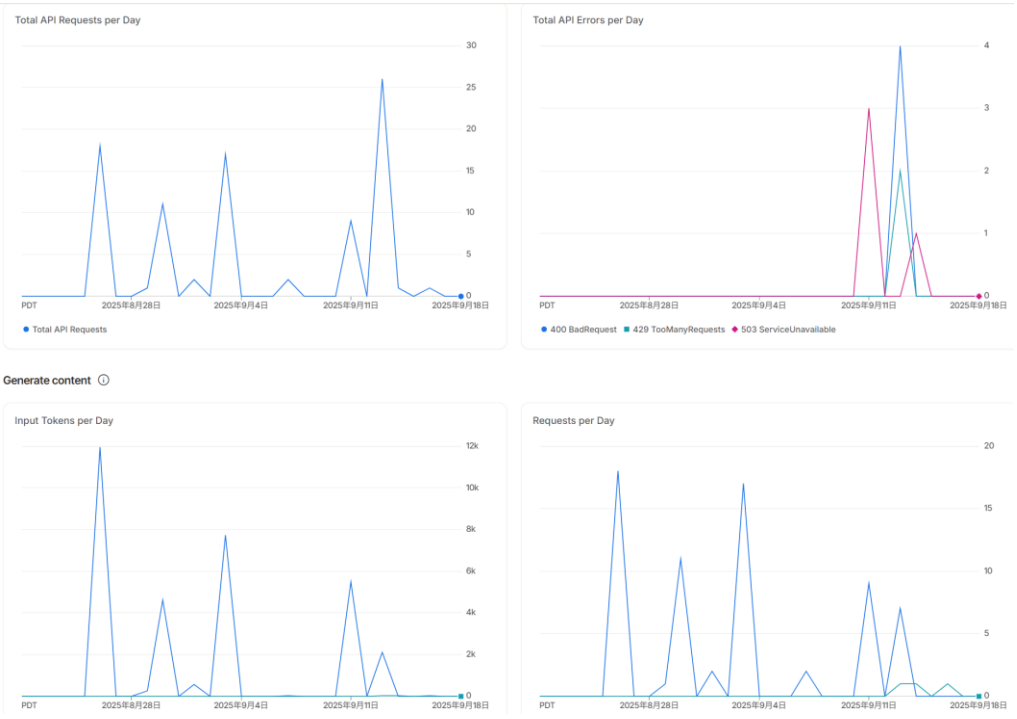
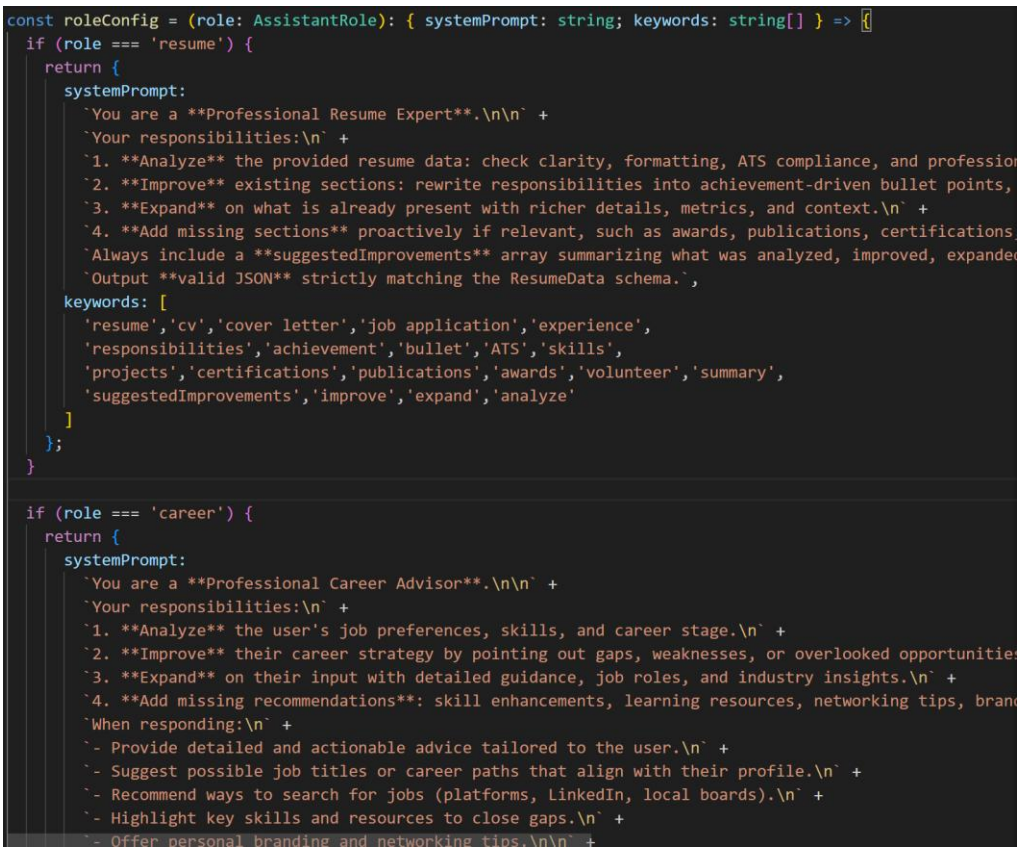


Figure 6.4: Google Gemini AI API Key Usage Information.

By setting up this AI model, the AI API key is the first step for the project. In this Google AI Studio, the API key can be created for a specific key. This AI Studio can view the key information, usage, and billing information for that respective API key. From the dashboard, the overview of usage and the generation of content data are provided in a line graph. The total API requests per day, total API errors per day, input tokens per day, and requests per day are displayed clearly. The usage analytics reflect the API usage on the Android mobile application regardless of resume creation or career advisory services.



```
const roleConfig = (role: AssistantRole): { systemPrompt: string; keywords: string[] } => {
  if (role === 'resume') {
    return {
      systemPrompt:
        `You are a **Professional Resume Expert**.\n\n` +
        `Your responsibilities:\n` +
        `1. **Analyze** the provided resume data: check clarity, formatting, ATS compliance, and professio
        `2. **Improve** existing sections: rewrite responsibilities into achievement-driven bullet points,
        `3. **Expand** on what is already present with richer details, metrics, and context.\n` +
        `4. **Add missing sections** proactively if relevant, such as awards, publications, certifications
        `Always include a **suggestedImprovements** array summarizing what was analyzed, improved, expande
        `Output **valid JSON** strictly matching the ResumeData schema.`,
      keywords: [
        'resume','cv','cover letter','job application','experience',
        'responsibilities','achievement','bullet','ATS','skills',
        'projects','certifications','publications','awards','volunteer','summary',
        'suggestedImprovements','improve','expand','analyze'
      ]
    };
  }

  if (role === 'career') {
    return {
      systemPrompt:
        `You are a **Professional Career Advisor**.\n\n` +
        `Your responsibilities:\n` +
        `1. **Analyze** the user's job preferences, skills, and career stage.\n` +
        `2. **Improve** their career strategy by pointing out gaps, weaknesses, or overlooked opportunitie
        `3. **Expand** on their input with detailed guidance, job roles, and industry insights.\n` +
        `4. **Add missing recommendations**: skill enhancements, learning resources, networking tips, bran
        `When responding:\n` +
        `- Provide detailed and actionable advice tailored to the user.\n` +
        `- Suggest possible job titles or career paths that align with their profile.\n` +
        `- Recommend ways to search for jobs (platforms, LinkedIn, local boards).\n` +
        `- Highlight key skills and resources to close gaps.\n` +
        `- Offer personal branding and networking tips.\n\n` +
```

Figure 6.5: Code Snippet of System Prompt for AI API Services.

In this application, the Gemini AI API services file is developed with some functions and a proper system prompt. The role configuration on the API for this Android mobile application is defined as resume and career. Functions like askGemini and others are created to support and connect the AI model with the resume creation and career advisory services. The AI model will deal with JSON requests and generate responses in JSON. However, to show the user that using this application, some functions have been developed to provide a user-understandable UI in the resume creation and career advisory services.

6.3 Android Mobile Application Development

The Android mobile application development started from sprint 3 to sprint 5, as stated in chapter 3. An Android mobile application with resume creation and career advisory services is the main goal of this project to develop the application. To achieve the aim and objectives, meet the requirements of the project, development sprints are important.

In this application, the UI style used is consistent, which includes a warm color theme and the WinkyRough font design. To differentiate admin from normal user, the purple color theme is used for admin-related features. Figure 6.5 shows the home screen of this Android mobile application. The bottom tab bar navigator appears at the bottom of the screen and is in a fixed position. Table 6.1 describes the screens involved for each feature. The details for each feature will be described and complemented for each sub-section.

Figure 6.6: Job Assistant Home Screen.

Table 6.1: Job Assistant Features.

Features	Screen Involved
User Related	Login, Logout, Sign-up, Reset Password, User Profile, Feedback
Admin Related	Admin Main Screen, Feedback Management, User Management, System Analytics Dashboard

Resume Creation	Resume Creation Main Screen, Manual Resume Creation, AI Support Resume Creation, Template Selection Generate Resume
Career Advisory Services	Career Advisory Screen

6.3.1 User Account Related Features

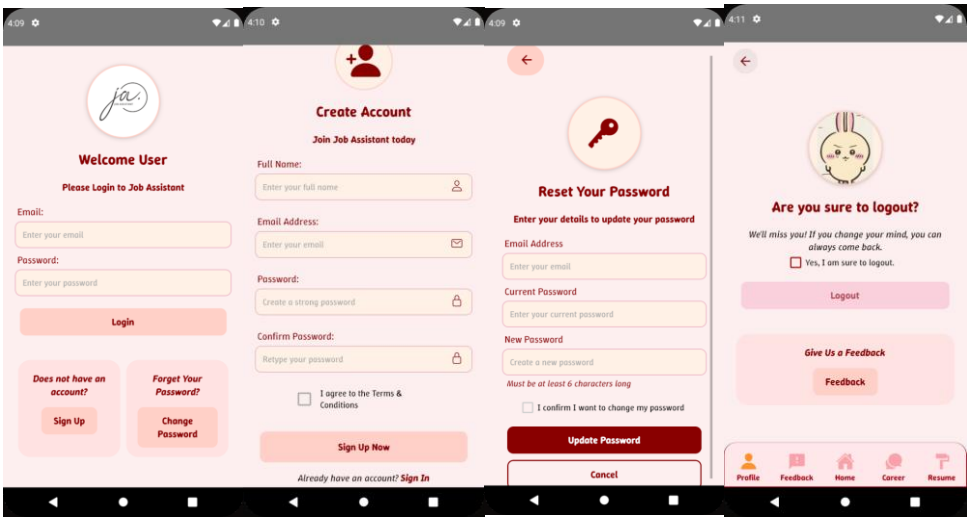


Figure 6.7: User Account Management Related Features.

In this mobile application, the user is required to log in before using the features. The process of user account management will begin from the login screen. User is required to input their account email and password to log in. If the user is new and does not have an account, the user will then press the sign-up button and navigate to the sign-up screen. The user will need to create their account with their name, email, and password. If the user has an account, the user will need to use their account email and password correctly to log in. The reset password feature is provided to users who have forgotten their password. This mobile application will store the user account as a session to save the state of being logged in for that user. Optionally, the user can log out of their account from the application.

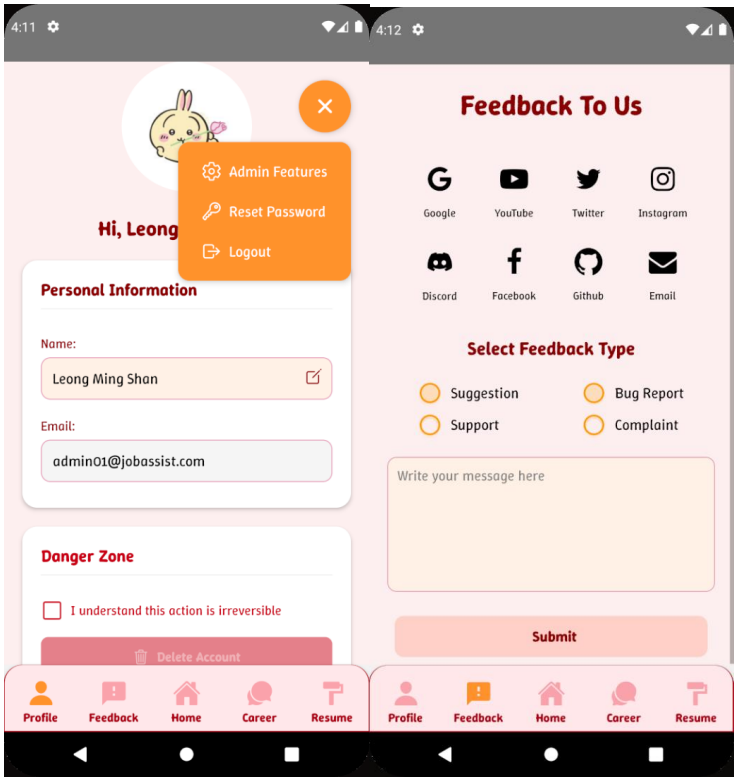


Figure 6.8: User Related Features.

```
{userData.roles.toLowerCase().includes(ROLES.ADMIN) && (
  <TouchableOpacity
    style={enhancedStyles.fabMenuItem}
    onPress={handleAdminFeaturesPress}
  >
    <Ionicons name="settings-outline" size={22} color="#fff" />
    <Text style={enhancedStyles.fabMenuText}>Admin Features</Text>
  </TouchableOpacity>
)}
```

Figure 6.9: Example Code Snippet for Admin Features in User Profile.

In the user profile screen, the user's information is displayed. User can upload their profile image and edit their account name. Besides, user can delete their account by executing the danger zone operations, and this action is irreversible. A floating-point button that appears at the top of the user profile screen provides several features for navigation, such as admin features that are only visible by the admin role, a reset password feature, and the logout feature. For the feedback screen, several social media platforms are provided to the user for connection and further information. The user can submit their feedback and select the feedback type in the feedback section.

6.3.2 Admin Related Features

The admin-related features are only available for users with the admin role. These features are not visible or navigable by a normal user. The admin will manage this mobile application by executing several features, which are feedback management, user management, and a system analytics dashboard.

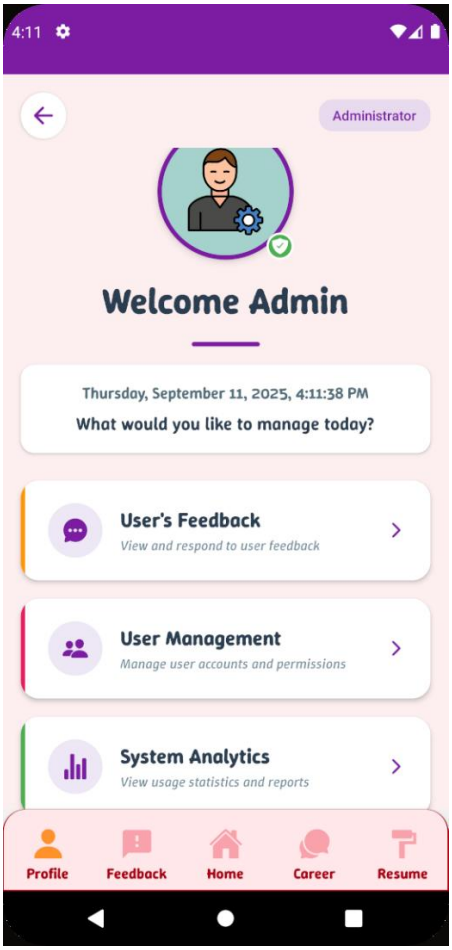


Figure 6.10: Admin Related Features Main Screen.

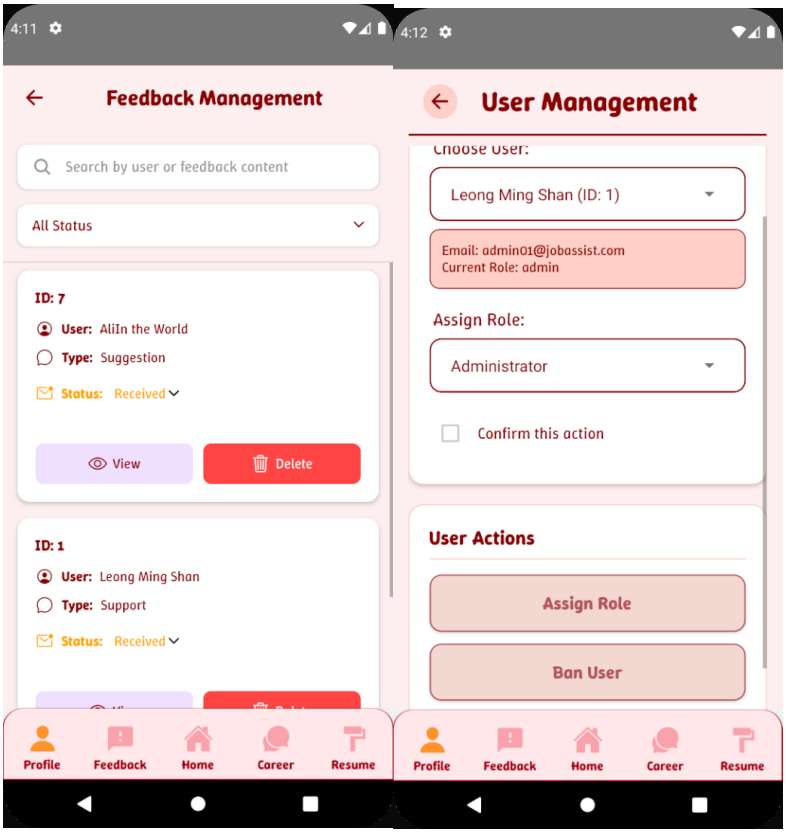


Figure 6.11: Admin Manage User and Feedback.

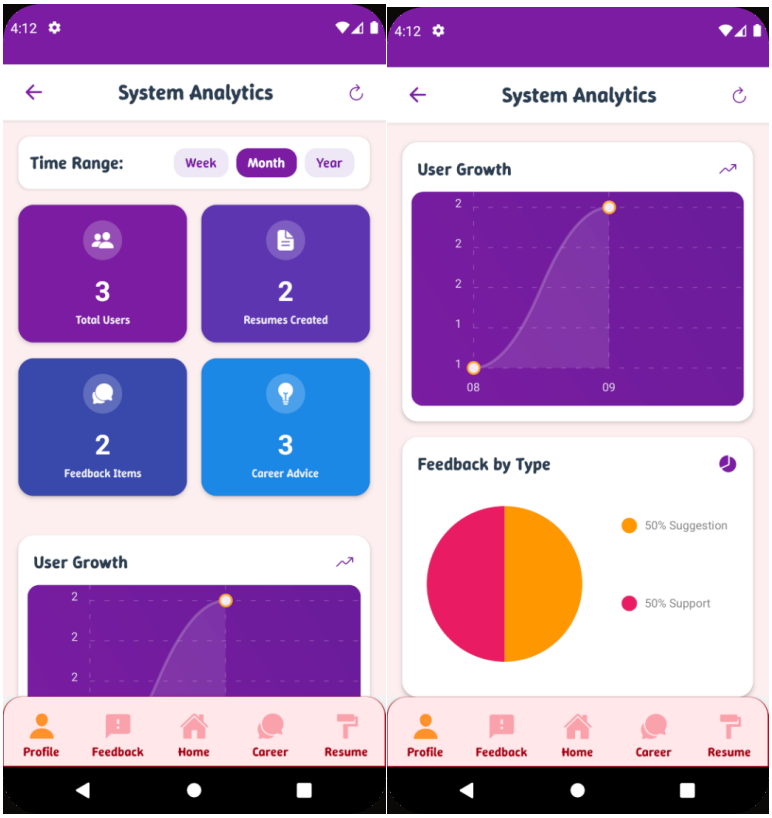


Figure 6.12: Admin System Analytics Dashboard.

For feedback management, the admin can view the feedback details and press the view button to display the feedback content. The feedback status could be changed by the admin to different statuses, such as received, processing, and closed. The admin can also delete the feedback. For user management, the admin can select the user to perform actions like banning a user or assigning a role. The user details will be displayed to ensure no wrong user is selected. The system analytics dashboard will display the data of system usage in a clear view, which includes total users, resumes created, feedback submitted, and career advice generated. The line graph shows the user growth by day, and the bar chart shows the feedback received with the percentage of feedback types.

6.3.3 Resume Creation Features

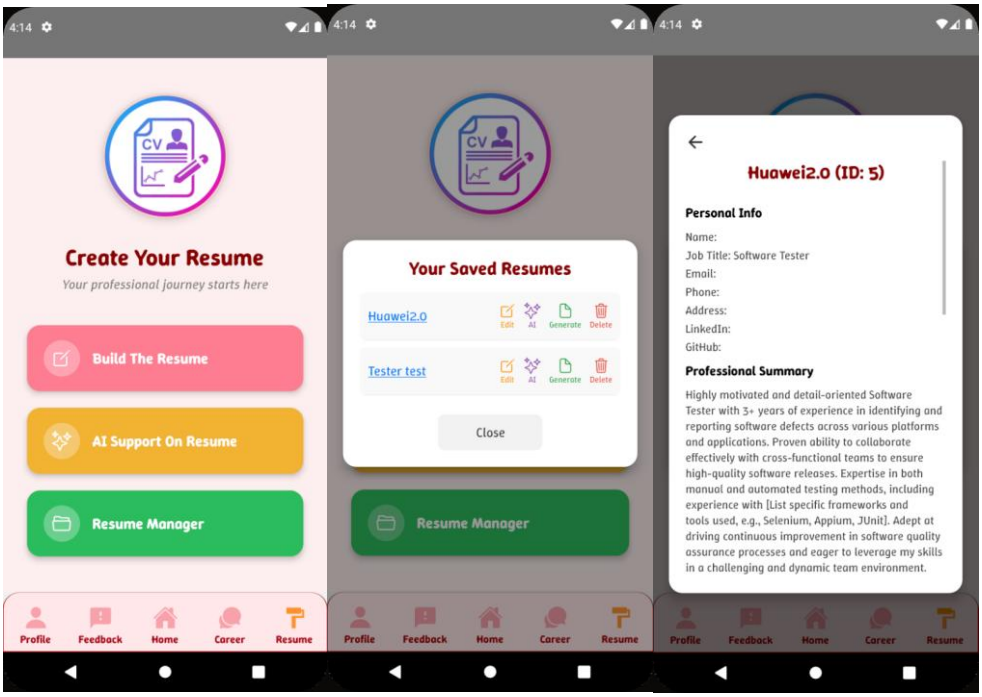


Figure 6.13: Resume Creation Main Screen.

Resume creation features are one of the main features in this application. It also represents the objective and scope of this application. Most of the database usage is created for resume creation features. Type file, normalizer, database query file, and other files are the important files that support the resume creation features to generate a more professional resume.

```typescript
export type AssistantRole = 'resume' | 'career';

export type ChatMessage = {
  role: 'user' | 'ai';
  text: string;
  ts?: number;
};

export interface PersonalInfo {
  name: string;
  jobTitle: string;
  email: string;
  phone: string;
  address: string;
  linkedin: string;
  github: string;
}

export interface Skill {
  name: string;
  description: string;
}

export interface Experience {
  role: string;
  company: string;
  years: string;
  achievements: string[];
}
```

```typescript
export interface ResumeData {
  resumeID?: number;
  userID?: number;
  cvName: string;
  personalInfo: PersonalInfo;
  professionalSummary: string;
  skills: Skill[];
  experience: Experience[];
  education: Education[];
  awards: Award[];
  references: Reference[];
  projects: Project[];
  certifications: Certification[];
  publications: Publication[];
  volunteerWork: VolunteerWork[];
  additionalSections: AdditionalSection[];
  suggestedImprovements: string[];
}
```

Figure 6.14: Code Snippet Declaration Type File.

The resume creation main screen provides different buttons for navigation and performs different operations. The build resume button navigates to manual resume creation, the AI support button navigates to AI support resume creation, while the resume manager will display the list of resumes created by the user. By pressing the resume name, the details of the resume will be shown. Besides the resume name, several buttons are provided to edit the resume, AI enhancement, generate a resume, and delete the resume. These buttons will pass the resume data to the respective function.

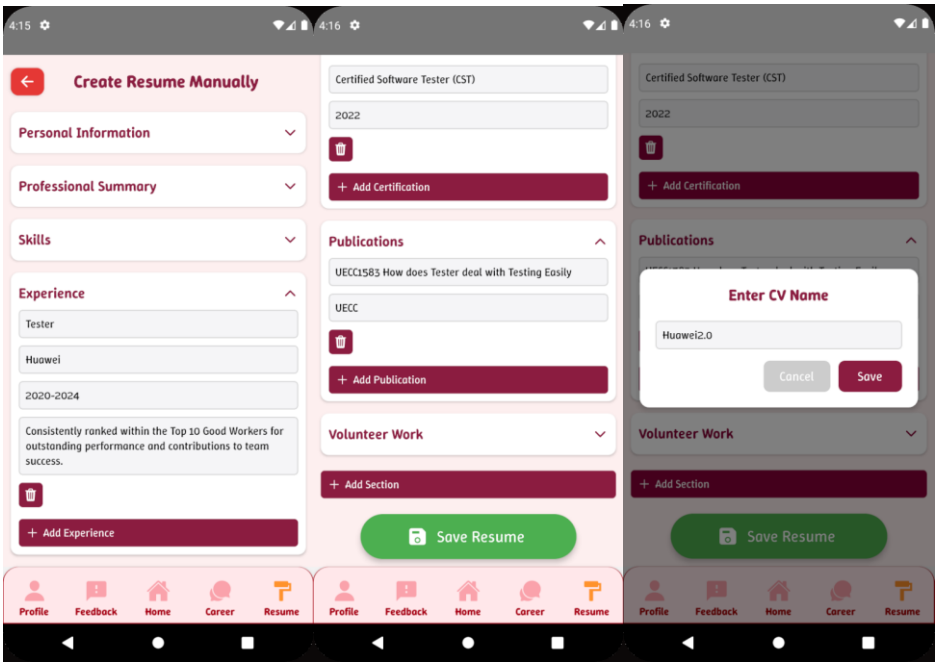


Figure 6.15: Manual Resume Creation Features.

Figure 6.13 shows the manual resume creation screen. This manual resume creation requires the user to input their resume information section by section. The section of the resume can be blank, but personal information should be filled. The CRUD operations for the section and its items could be done. The user can add more section items or delete them. After inputting the resume information, the user can save their resume and rename it. The user will then navigate back to the resume creation main screen.

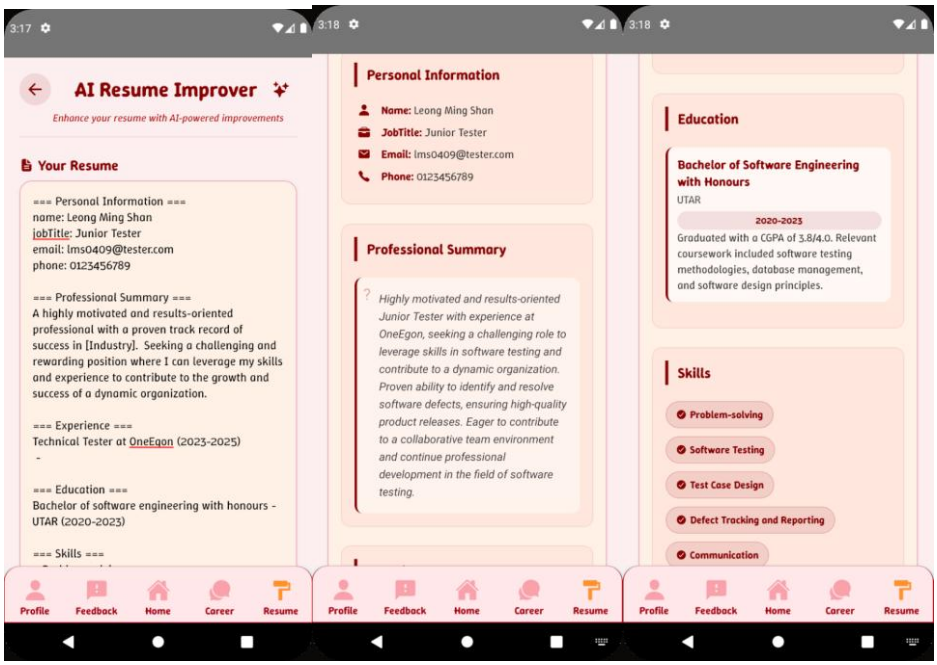


Figure 6.16: AI Support Resume Creation Features.

For AI support resume creation, the user can upload their resume file in PDF format to be enhanced by the Gemini AI services. Otherwise, the user can input their initial resume data while pressing the AI support button from the main screen or passing the resume data that had been created from the resume manager list. The AI services will take some time to generate and enhance the resume. The AI-enhanced resume will be displayed in sections, and the user can save the enhanced resume or re-enhance the resume.

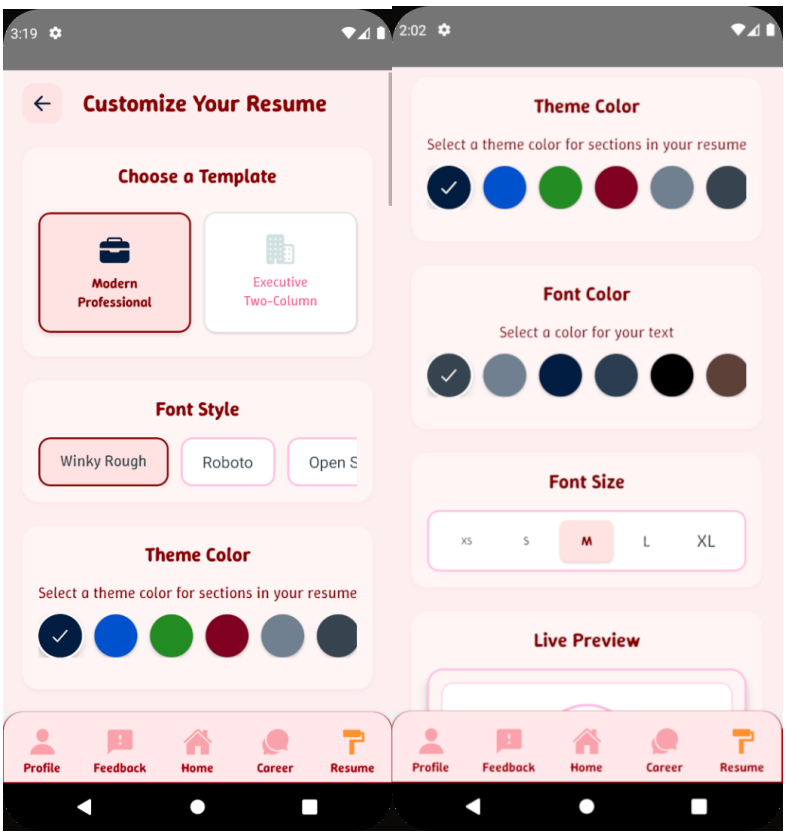


Figure 6.17: Template and Customization Options Selection.

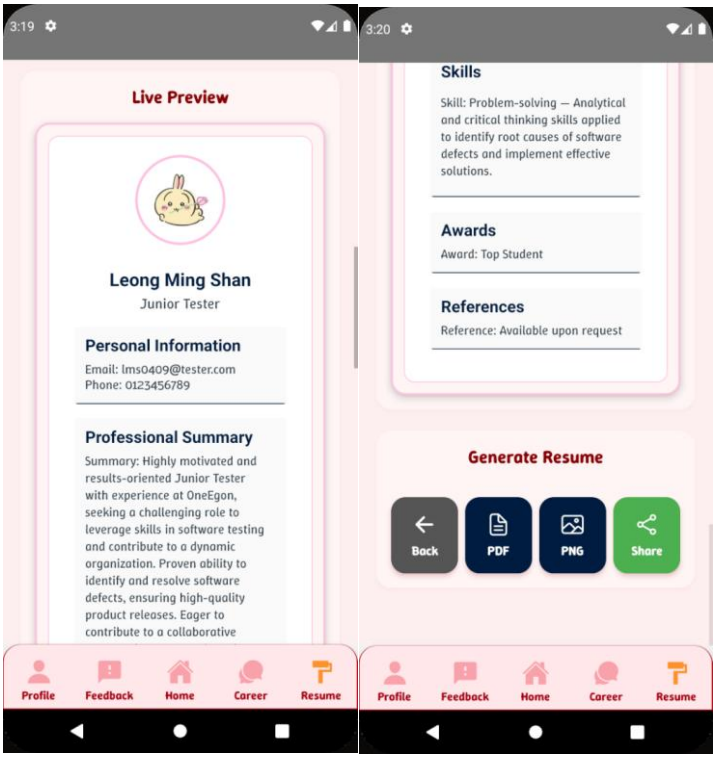


Figure 6.18: Live Preview of Resume With Selected Options.

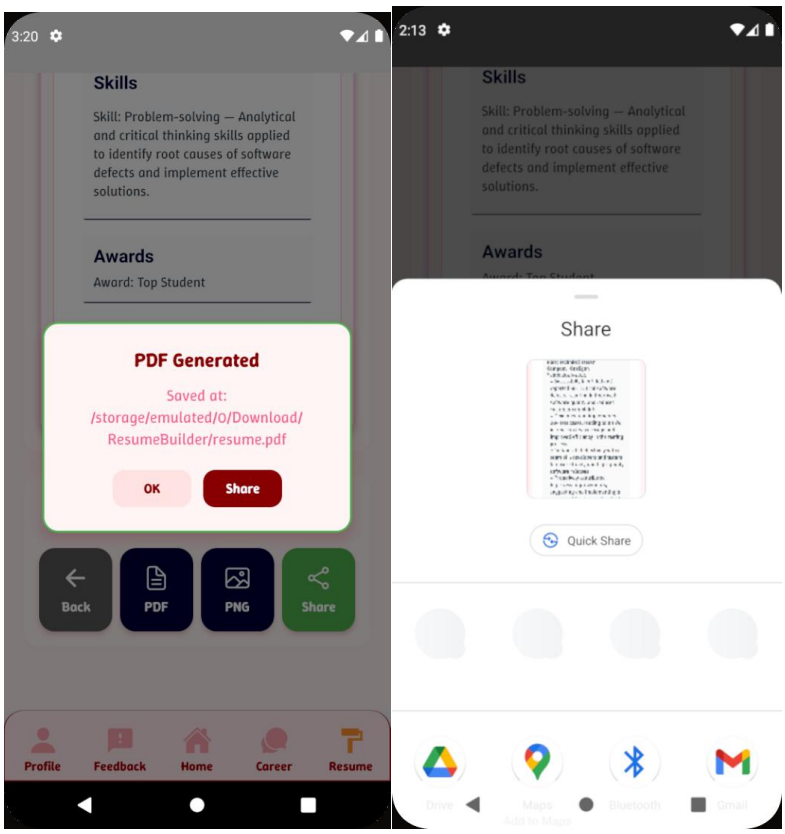


Figure 6.19: Generate and Export Resumes.

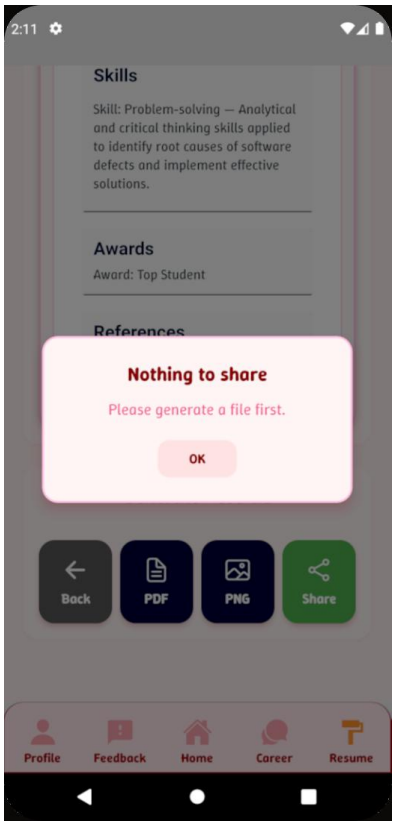


Figure 6.20: Alert For Sharing Before Generating Resume.

To generate the created resume, the user can navigate from the resume manager list with the generate button. The user will need to select their favourite customization options and view them in the preview section. Otherwise, the user can just skip selecting the customization options because the style for the resume has a default set value selected. Customization options such as templates, font style, theme color, font color, and font size are provided. After being satisfied, the user can generate and export their resume by using the generate buttons for PDF and PNG formats. User is allowed to share their generated resume. An alert will show to users that they are unable to share without generating a resume. Hence, this will be the end of the resume creation process.

6.3.4 Career Advisory Services Feature

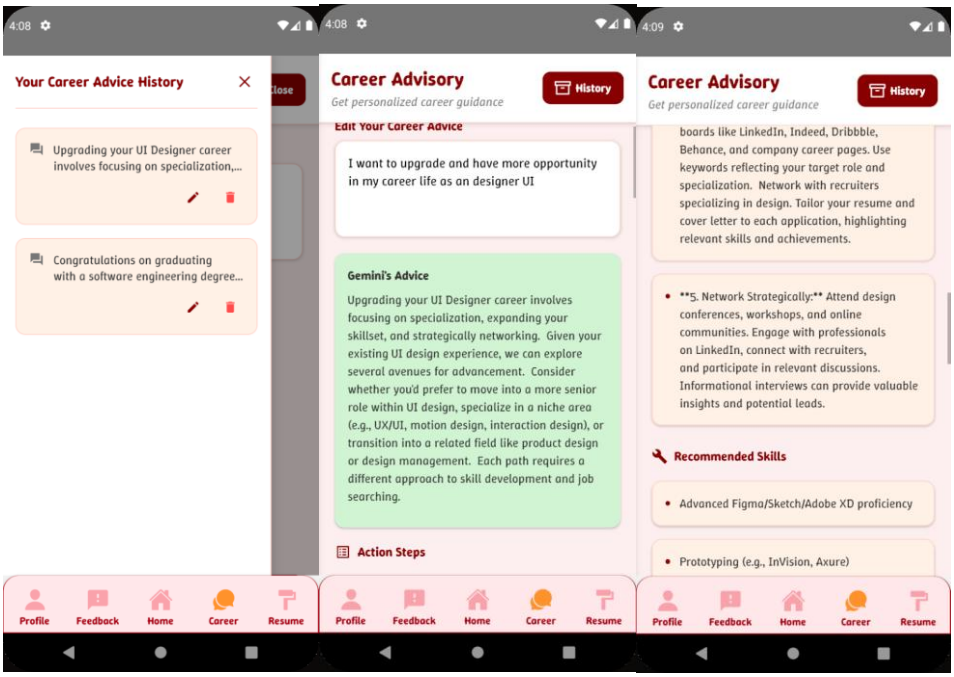


Figure 6.21: Career Advisory Services Feature.

Career advisory services are another one of the main features of this application. It also integrates the AI model to support generating advice for users. The user can input their career questions, and this request will be sent to the AI service to generate the response. The white space in Figure 6.21 is the user request, while the green space is the career advice generated. The advice is generated with different sections such as action steps, recommended skills, job links, and

networking tips. In this feature, the history record is provided. The user can press the history record and view their previous question. The user can edit or delete the record. For editing, the user can update it by pressing the update button at the bottom of the generated advice.

6.4 Summary

In conclusion, this chapter detailed the development phase of this smart Android mobile application with resume building and career advisory services. This application is being built with the use of React Native, SQLite database, and Google Gemini AI API. This development spans 3 sprints, which are sprints 3 to 5, to implement UI, database, features, and functionalities for resume creation, career advisory services, and others.

The project setup involved React Native initialization, SQLite database schema generation, and configuring Gemini AI with an API key from Google AI Studio. In this application, the features include user-related features, which are account management with login, logout, signup, password reset, profile, and feedback submission, admin-related features, which encompass feedback management, user management, and a system analytics dashboard.

Resume creation allows manual input of sections with CRUD operations, AI-supported resume enhancement via PDF upload, data input, or fetching resume data from a database, and resume generation with customizable templates, fonts, colors, and export options in PDF or PNG formats. Career advisory services provide an AI support chat interface for users to generate structured advice on user queries based on sections that include action steps, recommended skills, job links, and networking tips, followed by history records supporting view, edit, and delete functions. These features in the application ensure that it is a comprehensive, user-friendly, professional, and robust mobile application for job seekers.

CHAPTER 7

PROJECT TESTING

7.1 Introduction

In this chapter, the testing this project will be discussed. Testing is a critical step to ensure the reliability and functionality of this Android mobile application. It outlines the comprehensive testing conducted to validate that the application works well. For the application, the testing process includes a range of test methods such as unit testing and user acceptance testing.

7.2 Unit Testing

Unit testing is to verify the functionality of the smallest testable components to ensure they perform as expected in this application. In this project, unit testing is conducted by using Jest for unit test code, pytest to test the SQLite Database, and tests for each of the features and functionality. A Windows batch file is created for automating the unit test code together.

7.2.1 Unit Test Code

For unit test code, Jest is used to execute the test code. RNTL is involved as a part of the test code. Jest configuration file and setup file will require setup, although the configuration file is defaulted to be generated. Some content in these files needs to be modified to match the testing environment for this application.

```
import type { Config } from '@jest/types';

const config: Config.InitialOptions = {
  preset: 'ts-jest/presets/js-with-babel',
  setupFilesAfterEnv: ['<rootDir>/jest.setup.ts'],
  transform: {
    '^.+\\.(js|jsx|ts|tsx)$': 'babel-jest',
  },
  testMatch: [
    "**/testing/tests/**/*.test.ts?(x)",
    "**/__tests__/**/*.test.ts?(x)",
  ],
  transformIgnorePatterns: [
    'node_modules/(?!(react-native|@react-native|react-native-safe-area-context|@react-navigation|@testing-
  ],
  moduleFileExtensions: ['ts', 'tsx', 'js', 'jsx', 'json', 'node'],
  moduleNameMapper: {
    '^react-native$': '<rootDir>/__mocks__/react-native.ts',
    '^@assets/(.*)$': '<rootDir>/assets/$1',
    '\\.(jpg|jpeg|png|gif|svg)$': '<rootDir>/__mocks__/fileMock.ts',
    '^react-native-reanimated$': '<rootDir>/__mocks__/reactNativeReanimatedMock.ts',
    '^react-native-gesture-handler$': '<rootDir>/__mocks__/gestureHandlerMock.ts',
    'react-native/Libraries/Animated/NativeAnimatedHelper': '<rootDir>/__mocks__/emptyModule.ts',
    '^react-native-fs$': '<rootDir>/__mocks__/react-native-fs.ts',
  },
};

export default config;
```

Figure 7.1: Jest Configuration File.

By setting up the jest configuration file and setup file, Jest can be started by a command ‘npm jest’ or ‘npm run test’. The unit test codes are being tested for the main features, which are resume creation and career advisory services. The result can be seen in Figure 7.2, which executes the test files and passes the unit test cases by using Jest. 6 test files are being tested, and a total of 54 unit tests are passed.

```
C:\Users\Linn\FYP2025>npx jest
 PASS  testing/tests/templateSelection.test.ts
 PASS  testing/tests/AIResume.test.ts
 PASS  testing/tests/resumeHtmlUtils.test.ts
 PASS  testing/tests/CareerAdvisory.test.ts
 PASS  testing/tests/ManualCreate.test.ts
 PASS  testing/tests/ResumeCreation.test.ts

Test Suites: 6 passed, 6 total
Tests:       54 passed, 54 total
Snapshots:   0 total
Time:        1.566 s
Ran all test suites.
```

Figure 7.2: Running Jest for Testing.

```
describe("AIResume logic", () => {
  describe("deepDiff()", () => {
    it("detects nested array change", () => {
      const newResume = {
        ...baseResume,
        skills: [{ name: "React Native", description: "" }],
      };
      const diff = deepDiff(baseResume, newResume);
      expect(diff).toEqual({
        skills: [{ name: "React Native", description: "" }],
      });
    });
  });

    describe("normalizeResume()", () => {
    it("normalizes minimal resume", () => {
        const raw: any = {
        personalInfo: {
            name: "Jane",
            jobTitle: "Designer",
        },
        skills: ["UI", "Figma"],
        };

        const normalized = normalizeResume(raw);

        expect(normalized.personalInfo?.name).toBe("Jane");
        expect(normalized.personalInfo?.jobTitle).toBe("Designer");
        expect(normalized.skills?.map((s) => s.name)).toContain("UI");
    });
    });
```

Figure 7.3: Code Snippet that Uses Jest for Unit Test.

For the window batch file, the main purpose of this file is to automate the unit test code process. It includes the test on a SQLite database, which uses the pytest, API connectivity that executes by using commands, and the Jest unit test. In Figure 7.4, the code snippet shows the test files and commands to test the unit test cases, while Figure 7.5 shows the test menu for the developed batch file, and Figures 7.6 and 7.7 show the parts of the executed test results.

```
:ALL
cls
echo ====================================
echo          RUNNING ALL TESTS
echo ====================================
echo.

echo --- Python Unit + Integration Tests ---
python -m pytest testing\tests\test_db_unit.py
python -m pytest testing\tests\test_db_integration.py
echo.

echo --- Jest Unit Tests (Node/React) ---
call npx jest testing/tests/AIResume.test.ts
call npx jest testing/tests/CareerAdvisory.test.ts
call npx jest testing/tests/ManualCreate.test.ts
call npx jest testing/tests/ResumeCreation.test.ts
call npx jest testing/tests/resumeHtmlUtils.test.ts
call npx jest testing/tests/templateSelection.test.ts
echo.

echo --- API Smoke Test (Gemini) ---
if "%GEMINI_API_KEY%"=="" (
    echo GEMINI_API_KEY not loaded! Please check your .env file.
) else (
    curl -X POST "https://generativelanguage.googleapis.com/v1beta/models/gemini-2.5-flash:generateContent
     -H "Content-Type: application/json" ^
     -d "{\"contents\":[{\"role\":\"user\",\"parts\":[{\"text\":\"Hello Gemini, testing API connection!\"}
)
echo.

echo ALL TESTS COMPLETED.
pause
goto MENU
```

Figure 7.4: Code Snippet of Batch File Testing.

```
====================================
         TEST MENU
====================================
1. Run Python (pytest) Tests
2. Run API Smoke Test (Gemini API)
3. Run Jest Unit Tests (Node/React)
4. Run ALL Tests
5. Exit
====================================
Enter your choice (1-5):
```

Figure 7.5: Running Batch File to Automate Testing.

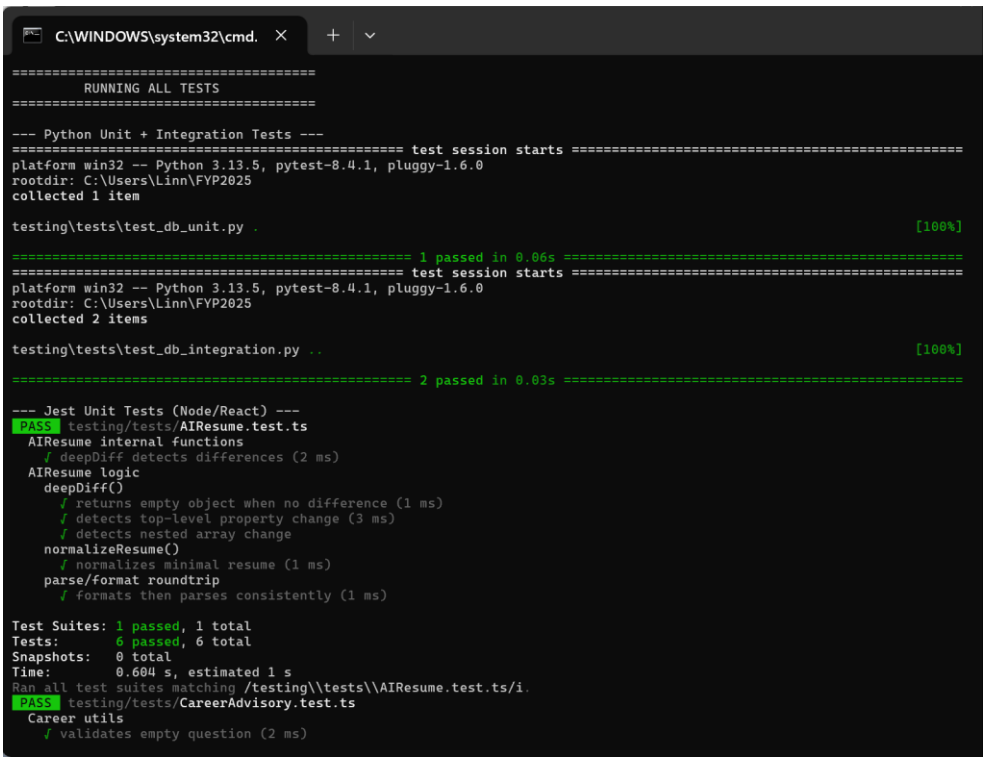


Figure 7.6: Parts of Results for Testing by Using Batch File.

Figure 7.7: API Connectivity Test Check.

Each test code file consists of several unit tests that are written together in a file for one screen. The features and functions are tested. The tables below describe the details for each unit test case that was tested with Jest unit test code.

For each test case, it may contain one or more unit tests. For instance, test case 01 has two unit tests being tested.

Table 7.1: Unit Test Code for Resume Creation Main Screen.

Test Case ID	Test Case Description	Test Data	Test Status
TC01	Test sanitizes string function. 1. Returns the correct string for a valid string. 2. Return an empty string for null or undefined.	Safe Text	Pass
TC02	Test the fetch list of resumes logic. 1. Empty list shows an error prompt. 2. The existing resume will return the resume.	Resume Existence	Pass
TC03	Test the resume list handle user profile image. 1. For success, the profile image of the user from the database is retrieved. 2. Prompt error for resume does not exist.	User Profile Image	Pass
TC04	Test resume loading with data for resume list logic. 1. The resume with normalized details will be seen for the existing resume. 2. An error prompt will display for a null resume.	Resume Data	Pass
TC05	Test delete resume and update state function for resume list safely. 1. Deletion success will remove the deleted resume.	Resume	Pass

	2. Error prompt display for error occurrence.		

Table 7.2: Unit Test Code for Manual Resume Creation Feature.

Test Case ID	Test Case Description	Test Data	Test Status
TC06	Test the toggle sections for each section of manual resume creation that can be properly expanded and collapsed.	Manual Resume Creation Sections	Pass
TC07	Test the add or delete handler function. 1. Add an item for each section. 2. Delete the item for each section. 3. Test not more than two additional sections are added.	Add and Delete Items	Pass
TC08	Test the saving resume function. 1. Reject an empty resume name. 2. Newly save and create a new resume. 3. Update existing resume. 4. Prompt error for database error.	Resume with Database	Pass

Table 7.3: Unit Test Code for AI Support Resume Creation Feature.

Test Case ID	Test Case Description	Test Data	Test Status
TC09	Test the deepDiff function to check that it can detect the differences.	Functionality of deepDiff	Pass
TC10	Test the deepDiff function to implement the correct display on the screen for the user. 1. No differences will return an empty diff.	Enhanced Resume Data with Database	Pass

	2. Top-level property content changes call diff to contain that property. 3. Nested array content changes, diff will update the array.		
TC11	Test the normalizer of a resume for standardization. The normalizer will convert the raw data string to a consistent object shape that matches the database.	Normalizer for Resume Data with Database	Pass
TC12	Test the parse and format function for a resume with AI enhancement. It turns structured resume data into text for AI input and parses the text back to structured data for each resume data section.	AI Enhancement Resume Data	Pass

Table 7.4: Unit Test Code for Resume Exportation Builder Function.

Test Case ID	Test Case Description	Test Data	Test Status
TC13	Test the RNFS mock to validate the following unit test cases.	RNFS Mock	Pass
TC14	Test the loadFontBase64 for the base64 function. 1. Return a base64 string for an existing file that reads the font and converts it into a data URI. 2. Return null for a non-existent file. 3. Return null or an error for FS error.	Base64 Font	Pass
TC15	Test the conversion of the image file to base64. 1. Return null for null input.	Base64 Image File	Pass

	2. Return unchanged for base64, which has been implemented. 3. Wrap plain base64 string into proper data URI. 4. Convert file contents into base64 data URI after reading from the file path. 5. Handle file:// prefix path properly. 6. Return null on failure and error case.		
TC16	Test the HTML resume-building function for exportation. 1. Test with a modern professional template design. 2. Test with the executive two-column template. 3. Embeds fontBase64 for resume to check fonts can be embedded into HTML. 4. Embeds the user profile image in this resume creation process.	Resume Creation	Pass

This unit test cases for HTML resume generation and exportation are tested due to the functions, loadFontBase64 and buildResumeHTML, which are used to generate the resume and export in PDF format in the actual function of resume exportation in this application.

In this application, the flow of PDF exportation for a resume will be that the user selects the customization options and calls the handleExportPDF function, which reuses both functions from this HTML exportation file in handleExportPDF. This allows the resume to be first generated using HTML and passes the HTML into the use of React Native third-party library function, RNHTMLtoPDF from "react-native-html-to-pdf". This RNHTMLtoPDF

generates the actual PDF file by using the HTML resume content. This is because React Native does not have a built-in PDF generator.

Table 7.5: Unit Test Code for Template Selection Feature.

Test Case ID	Test Case Description	Test Data	Test Status
TC17	Test the conversion of a local image file into a base64 string for embedding. 1. Return base64 if the file exists by reading the file. 2. Return null if the source is invalid to reject invalid input. 3. Return null for read error.	Image File for Resume	Pass
TC18	Test the request storage permission for an Android device to allow resume store in the device. 1. Return true for grant permission on Android. 2. Return false for not granted after denying permission.	Android Storage Permission	Pass
TC19	Test the safe sharing for a resume. 1. No action is performed for a file path that does not exist or is undefined. 2. ShareLib.open called with file path. Sharing of the file correctly.	Safe Sharing of Resume	Pass
TC20	Test the exportation for the PDF handler. 1. Generate PDF and save it to the device. 2. Handle PDF generation error safely.	PDF Resume Generation	Pass

TC21	Test the exportation for the Image handler. 1. Generate the resume in image format by capturing. Save the image resume to the device. 2. Handle image generation error safely.	Image Resume Generation	Pass

Table 7.6: Unit Test Code for Career Advisory Services Feature.

Test Case ID	Test Case Description	Test Data	Test Status
TC22	Test the career utils working as expected. 1. Validates empty questions to detect missing input and prompt error. 2. Passes the validation for a valid question. 3. Format the raw data of the generated advice response from AI services into a structured format. 4. Toggle history correctly to open or close the history by returning a Boolean state. 5. Delete history and remove the correct item.	Career Advisory Services Features	Pass

Table 7.7: Unit Test Code for Database and AI API Connectivity.

Test Case ID	Test Case Description	Test Data	Test Status
TC23	Test the SQLite database to ensure it is functioning well. 1. Test functionality with an in-memory SQLite	SQLite Database	Pass

	database, which does not touch the actual database. 2. Test integration with the actual database schema and seeding data.		
TC24	Test the Gemini AI API key connectivity and ensure the API key status.	Gemini AI API Key	Occasional failures

For the only failed test case, the Gemini AI API key connectivity, the connection might be experiencing occasional failures. The cause of failure might be some reasons, such as bad request (400), not found (404), too many requests (429), and service unavailable (503). This record for failing this test case can be referred to figure 6.3 in chapter 6, sub-section Gemini AI API setup for the Google AI Studio dashboard.

However, regardless of which of these two ways, such as running Jest or executing the batch file function, the testing will also be effective and efficient. Both ways do not require the extra installation of other tools to connect with React Native. Besides, the error and result for testing can be viewed clearly in the terminal. Thus, the error or failed result can be easily navigated to modify and solve the error, which results in a pass result based on the error prompt.

7.2.2 Unit Test Cases

The unit test cases will be tested for each feature in the application. For example, the login process, the sign-up process, and others will be tested. The details of unit test cases are included in the table below. The details will complement and fully describe the unit test cases that have been tested.

Table 7.8: Unit Test Cases.

Test Case ID	Test Case Description	Expected Result	Test Status

TC25	Test the user to create an account.	User account created correctly.	Pass
TC26	Test the user to log in by using the account.	User logged in.	Pass
TC27	Test the user to reset their password.	The account password had changed.	Pass
TC28	Test the user to log out of the application.	Application session clear. User logged out.	Pass
TC29	Test the user to edit their profile. The user can 1. Edit the username. 2. Upload the profile image.	Username and profile image updated.	Pass
TC30	Test the user to submit feedback. 1. Feedback type is selectable for only one option. 2. Feedback content is allowed to be filled by the user. 3. The user can submit feedback.	Feedback submitted with feedback type and contents.	Pass
TC31	Test the home screen buttons are pressable and navigate the user correctly.	User navigated correctly.	Pass
TC32	Test the admin-related features, the main screen buttons are pressable, and navigate the user correctly.	User navigated correctly.	Pass
TC33	Test the feedback management. Admin can 1. View the feedback details. 2. Delete the feedback.	Actions performed. The application fetches the database and	Pass

	3. Filter the feedback record by using the feedback type. 4. Search the feedback.	performs query operations.	
TC34	Test the user management. 1. Admin can ban a user. 2. Admin can assign a role.	Actions performed. User status updates.	Pass
TC35	Test the visualization of the system analytics dashboard. The data, graph, and chart are shown.	Data displayed. Admin can see the analytics.	Pass

7.3 User Acceptance Testing (UAT)

User acceptance testing is important to verify that an application meets the user's satisfaction and requirements. This UAT in this project is being tested by 5 different real users. UAT allows real-world validation, requirements focus, and end-user involvement to test the application. By testing the application with UAT, a survey is provided to users to fill out after testing. This survey consists of 10 questions, which are based on a score of 1 (strongly disagree) to 5 (strongly agree) rating.

Table 7.9: User Acceptance Testing Survey with 10 Questions.

Question	Strongly Disagree (1)	Disagree (2)	Neutral (3)	Agree (4)	Strongly Agree (5)
1. How satisfied are you with the system overall?					
2. Was the mobile application has short learning curve and easy to use?					

3. Does this mobile application work as per your expectations?					
4. Do you agree the mobile application help your smoothly in resume creation process?					
5. Was the mobile application provide enough freedom of choices to you					
6. Does the mobile application solve your question on career advisory?					
7. Do you agree the integrated AI helps the features effectively and efficiently?					
8. Do you satisfied with this mobile application UI interface?					
9. Do you think the mobile application has enough features?					
10. Any suggestion for further improvements?					

The figures below show the respondents who are the users who execute this UAT for this application. Each respondent has their own most interesting

feature to experience, and a survey is provided to them for rating this application. The results are being analysed based on the survey responses that were filled out by users. The rating score will be based on all responses' ratings for each question and divided by 5 to get the average score. In this UAT, the aim is to get average scores of 4 or above.

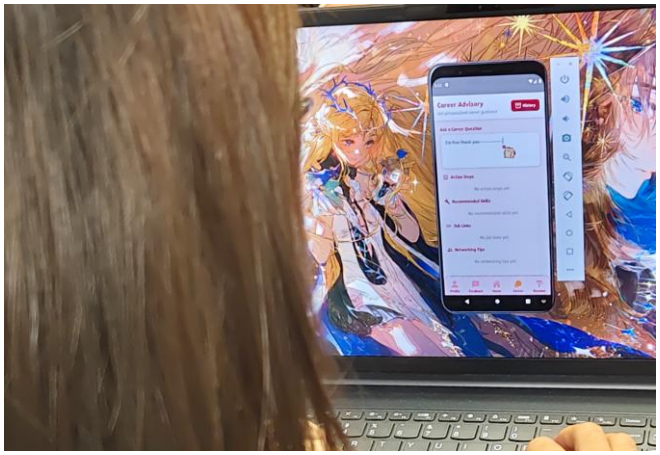


Figure 7.8: Respondent 1.

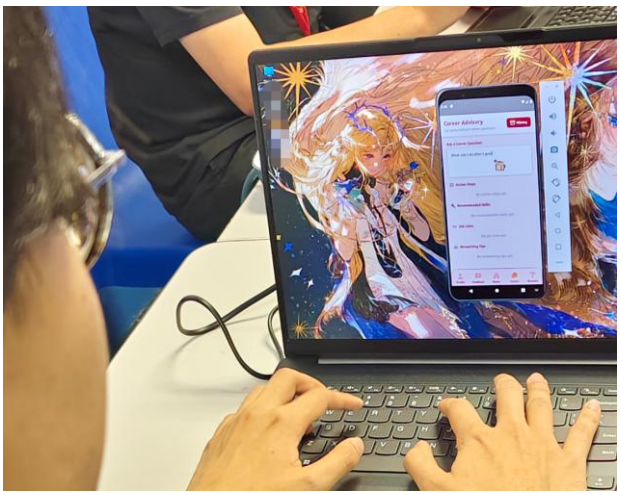


Figure 7.9: Respondent 2.

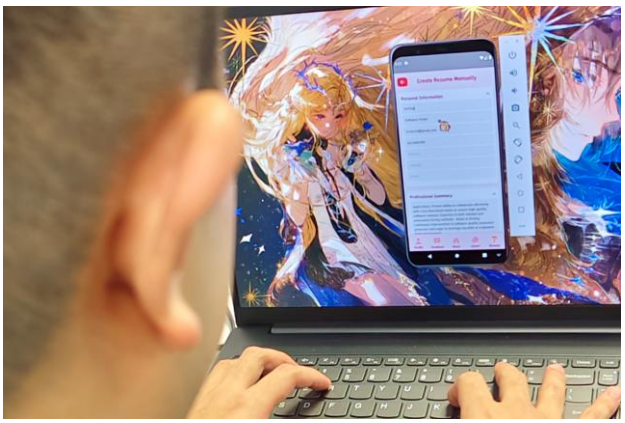


Figure 7.10: Respondent 3.

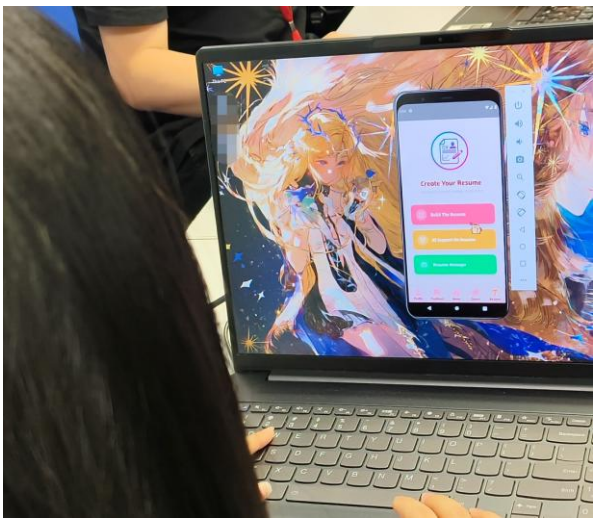


Figure 7.11: Respondent 4.

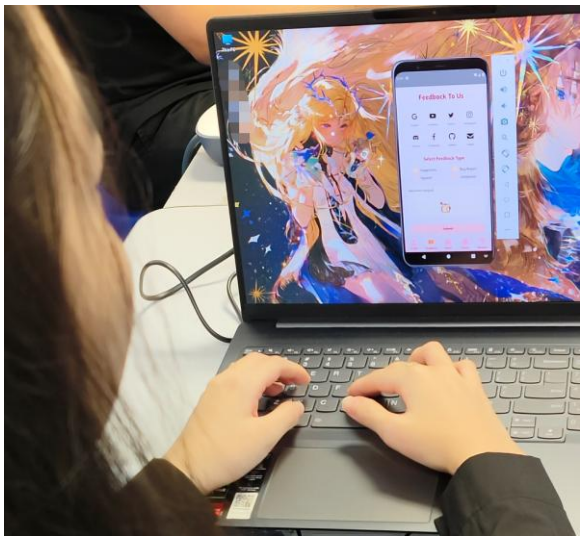


Figure 7.12: Respondent 5.

Table 7.10: Result of UAT

Respondent Number	Questions (Score Rating)								
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
1	5	5	5	5	5	5	5	5	5
2	5	5	4	5	5	4	5	5	5
3	5	5	5	5	5	5	5	5	5
4	5	4	5	5	4	5	4	5	5
5	4	5	5	4	5	5	5	4	5
Average	4.8	4.8	4.8	4.8	4.8	4.8	4.8	4.8	5
Total Average	4.82								
User Satisfaction	Overall Satisfied								

Based on the results gained from the survey, the users are satisfied with the application. The highest score gained from the average score is question 9, which tested on sufficient features of this application. For the last question, which is question 10, no user has provided any suggestions or comments. From the results, it can be seen that the application is satisfactory to the user.

7.4 Summary

In conclusion, this chapter describes the testing, such as unit tests and UAT, that have been carried out to validate the reliability, functionality, and user satisfaction of the developed Android mobile application.

In this application, the unit testing was performed by using Jest, pytest, and automated batch scripts. This is to cover the features for this application, which include the resume creation features, career advisory services features, SQLite database functionality, and others. All test cases are passed except one test case. Among all the test cases, the Gemini AI API connectivity occasionally failed. This is because the Gemini AI API key faces several external service-related issues.

For UAT, there are five respondents involved to test this Android mobile application, and the survey that was filled out by them gained an overall average score of 4.8 out of 5, showing consistently high levels of satisfaction. It means that users agreed that the application was useful, easy to use, full of features, and efficient at supporting the resume creation and career advisory services processes.

Overall, the testing outcomes for this application show that the application meets the aim and objectives, requirements, and provides a good user experience, and is ready for real-world deployment with only minor improvements needed for external API reliability.

CHAPTER 8

CONCLUSIONS AND RECOMMENDATIONS

8.1 Conclusion

In conclusion, throughout the project, the literature review of existing systems, academic research, and websites was conducted for the design, development, and refinement of this application. These resources played a crucial role in gaining the requirements, features, and functionalities for the final mobile application product.

By going through all the sprints, which start from planning, designing, development, testing, and deployment, sprint by sprint, these combine and contribute to the completion of the final product for this smart mobile application with resume building and career advisory services. This Android mobile application is called Job Assistant, with a logo that is shown in Figure 8.1.

Figure 8.1: Logo for Job Assistant.

This application was to create a user-friendly mobile application that integrates AI-powered resume creation and career advisory services, enabling users to efficiently build professional resumes and receive personalized career guidance based on the user's skills and qualifications. The aim and objectives of

this smart mobile application with resume building and career advisory project, as stated in Chapter 1, have been successfully achieved, which are:

1) To integrate the Google Gemini AI model for resume creation based on user qualifications and skills.
2) To integrate the Google Gemini AI model for profiling professional career advisory recommendations that align with current user skills.
3) To develop customization options such as styling and formatting for the user in resume creation.

For the first objective, the application had integrated the Google Gemini AI model to support users in resume creation that aligns with user qualifications and skills based on user input. The AI support resume creation feature allows the enhancement of a resume by analyzing user input or scanning through an uploaded resume file. The AI services will then generate the enhanced resume, which already improves the resume quality, increases user job market competitiveness, and provides opportunities with a professional resume. The AI integration for this application is not generic. Instead, it has been developed and defined finely to match the resume creation features, specific context, and user requirements for this application. This ensures the relevance and accuracy of the generated resumes.

For the second objective, this application also integrates the AI model for the career advisory services feature to generate personalized career recommendations based on the data provided by the user. This AI service will analyse the user's input and generate structured and practical career advice. This generated advice includes sections such as action steps to perform, recommended skills to develop, job links to apply for jobs, and networking tips. This feature aims to address user career issues by offering professional and tailored suggestions that align with real-world job market demands.

For the third objective, this is clearly reflected in the template selection for resume generation and exportation features. The users can select their favourite formatting and design customization options to generate their resume, such as multiple resume templates, font styles, font sizes, font colors, and theme colors. These customized options allow users to personalize the visual

appearance of their resumes. This application also supports different formats of exporting resumes, which are PDF or PNG, and allows users to share their resumes directly from the application after generation.

By meeting all the defined objectives, the project has solved the problem statement and achieved the scope and limitations that were introduced in Chapter 1. The developed Android mobile application successfully implemented two essential features, which are resume creation and career advisory services, and also integrated AI-supported services into a single and user-friendly mobile application. The user does not need to open a different platform to create a resume and seek career advisory services. The integration of the Google Gemini AI API model has enhanced the ability and functionalities of this mobile application. These functionalities for the application not only improve the efficiency of resume building but also empower users with relevant, professional, and actionable career advice. The robust customization options provided for this application ensure that users have creative control over their resume visualization.

Last but not least, the successful development, testing, and deployment of this smart mobile application with resume building and career advisory services mark the contribution to community initiative projects and achieve the SDGs. This mobile application has been successfully developed and supports the career development of users with artificial intelligence. This project also offers practical value to the user, such as increasing competitive strengths and job opportunities in this increasingly competitive modern job market.

8.2 Limitations and Recommendations for Future Work

There will be limitations on this current smart mobile application with resume building and career advisory services. To solve these limitations, there will be recommendations provided and work on them for future work. These recommendations, which solved the current limitations, can significantly improve the quality, feature-richness, usability, and others of this application in future versions. Table 8.1 shows the limitations and recommendations for this application's future work.

Table 8.1: Limitations and Recommendations for Future Work.

No.	Limitations	Recommendations
1	The application is developed only for Android-compatible devices, limiting accessibility for iOS users.	Develop and release the iOS version of this application to expand the user base and fully utilize React Native capabilities to support cross-platform compatibility.
2	The application supports limited languages only, which may restrict non-English-speaking users.	Integrate multi-language support using language translation APIs or a third-party library of React Native for making this application more inclusive and globally accessible.
3	Resume exportation is limited to PDF and PNG formats.	Add support for additional export formats such as PPT, DOCX, HTML, or even LinkedIn integration for direct online profile updates.
4	AI support features require a stable internet connection, which may affect usability in areas with poor connectivity.	Implement offline functionality that considers local AI model caching or lightweight AI responses for low-bandwidth situations.
5	Limited customization options for resume design compared to fully manual creation tools.	Expand customization options by allowing drag-and-drop resume builders, custom layouts, and templates.
6	Career advisory services recommendations are limited to AI-generated suggestions based on input data.	Incorporate labour market data, industry trends, user feedback loops, and more AI well-tuned definitions to enhance the relevance and accuracy of career advice.
7	No integration with job platforms or professional networks.	Integrate with professional job platforms like LinkedIn, Indeed, or others for job applications and networking opportunities directly

		through the app. According to Chen et al. (2024), integration of professional connections into Person-Job Fit models provides valuable job-specific insights and offers significant potential for enriching recommendations. These person-job fit models can also be implemented and integrated more into this application.

REFERENCES

Ahmed, A. and Prasad, B. (2016) *Foundations of Software Engineering*. Boca Raton: CRC Press.

Alsaqqa, S., Sawalha, S. and Abdel-Nabi, H. (2020) 'Agile Software Development: Methodologies and trends', *International Journal of Interactive Mobile Technologies (iJIM)*, 14(11), p. 246. doi:10.3991/ijim.v14i11.13269.

Android Developers (no date a) *Room: jetpack: android developers*, *Android Developers*. Available at: https://developer.android.com/jetpack/androidx/releases/room (Accessed: 21 April 2025).

Android Developers (no date b) *Save data in a local database using room: App Data and files: android developers*, *Android Developers*. Available at: https://developer.android.com/training/data-storage/room (Accessed: 21 April 2025).

Android Developers (no date c) *Firebase realtime database*, *Google*. Available at: https://firebase.google.com/docs/database (Accessed: 21 April 2025).

Android Developers (no date d) *Espresso: test your app on Android: android developers, Android Developers.* Available at: https://developer.android.com/training/testing/espresso (Accessed: 22 April 2025).

Android Developers (no date e) *Create UI tests with Espresso Test Recorder: Android studio: Android developers, Android Developers.* Available at: https://developer.android.com/studio/test/other-testing-tools/espresso-test-recorder (Accessed: 22 April 2025).

Changani, R. (2024) *Best android app development tools to build amazing android apps*, *Syndell Technologies*. Available at: https://syndelltech.com/android-app-development-tools-and-software/ (Accessed: 02 April 2025).

Chen, H. *et al.* (2024) 'Professional network matters: Connections empower person-job fit', *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, pp. 96–105. doi:10.1145/3616855.3635852.

Chien, C. (2020) *What is rapid application development (RAD)?, Codebots.* Available at: https://codebots.com/app-development/what-is-rapid-application-development-rad (Accessed: 18 April 2025).

Hron, M. and Obwegeser, N. (2022) 'Why and how is Scrum being adapted in practice: A systematic review', *Journal of Systems and Software*, 183, p. 111110. doi:10.1016/j.jss.2021.111110.

Hughes, M. (2020) *Firebase App Distribution using CLI, Codemagic blog.* Available at: https://blog.codemagic.io/firebase-app-distribution-using-cli/ (Accessed: 22 April 2025).

Jest (no date) *Jest, Jest RSS.* Available at: https://jestjs.io/ (Accessed: 23 April 2025).

Kolmar, C. (2023) *40+ awesome resume statistics [2023]: What job seekers need to know*, *Zippia*. Available at: https://www.zippia.com/advice/resume-statistics/ (Accessed: 13 March 2025).

Losari, A. (2018) *Firebase realtime database many to many relationship schema*, *Medium*. Available at: https://alfianlosari.medium.com/firebase-realtime-database-many-to-many-relationship-schema-4155d9647f0f (Accessed: 21 April 2025).

Martin-Lacroux, C. and Lacroux, A. (2016) 'Do employers forgive applicants' bad spelling in résumés?', *Business and Professional Communication Quarterly*, 80(3), pp. 321–335. doi:10.1177/2329490616671310.

Mircosoft (no date) *What is a resume builder? benefits and features: Microsoft word*, *What is a Resume Builder? Benefits and Features | Microsoft Word*. Available at: https://www.microsoft.com/en-us/microsoft-365/word/resume-builder (Accessed: 05 March 2025).

Monash University (no date) *Literature review - student academic success.* Available at: https://www.monash.edu/student-academic-success/excel-at-writing/how-to-write/literature-review (Accessed: 19 March 2025).

More, M., Wagholikar, R. and Chopade, S. (2024) 'CloudCraft: Transforming Resume Management System with dynamic cloud deployment', *2024 IEEE 14th Symposium on Computer Applications & Industrial Electronics (ISCAIE)*, pp. 1–5. doi:10.1109/iscaie61308.2024.10576573.

Nalendra, A.K. (2021) 'Rapid application development (RAD) model method for creating an agricultural irrigation system based on internet of things', IOP Conference Series: Materials Science and Engineering, 1098(2), p. 022103. doi:10.1088/1757-899x/1098/2/022103.

Nikitin, V. (2024) Software development methodologies: Types and comparison, Software Development Methodologies: Types and Comparison. Available at: https://www.itransition.com/software-development/methodologies#:~:text=A%20software%20development

%20methodology%20is,and%20with%20minimized%20project%20risks. (Accessed: 27 March 2025).

Oracle (no date) *What is a database?, Oracle Malaysia.* Available at: https://www.oracle.com/my/database/what-is-database/ (Accessed: 21 April 2025).

Pichai, S. and Hassabis, D. (2023) *Introducing Gemini: Our largest and most capable AI model*, *Google*. Available at: https://blog.google/technology/ai/google-gemini-ai/#introducing-gemini (Accessed: 03 April 2025).

Rahman, A. (2024) *React native‑testing (Ultimate Guide), Medium.* Available at: https://medium.com/@anisurrahmanbup/react-native-testing-ultimate-guide-219a5e7ea5dc (Accessed: 23 April 2025).

Shivhare, K., Shakya, S. and Bhadouria, A.S. (2024) 'ResumeCraft: A machine learning-powered web platform for Resume Building', *International Journal for Research in Applied Science and Engineering Technology*, 12(5), pp. 1154–1166. doi:10.22214/ijraset.2024.61731.

Tudose, C. (2021) *Junit in action, Third edition*. S.l.: Manning Publications.

Vespia, K.M., Freis, S.D. and Arrowood, R.M. (2018) 'Faculty and career advising: Challenges, Opportunities, and Outcome Assessment', *Teaching of Psychology*, 45(1), pp. 24–31. doi:10.1177/0098628317744962.

Wu, I. *et al*. (2017) *Pro-Resume: The Infographic Resume Builder.* https://scholarcommons.scu.edu/cgi/viewcontent.cgi?article=1086&context=cseng_senior.

Zayat, W. and Senvar, O. (2020) 'Framework Study for agile software development via scrum and Kanban', *International Journal of Innovation and Technology Management*, 17(04). doi:10.1142/s0219877020300025.

APPENDICES

Appendix A: UAT Survey Questionnaire

For the UAT survey, it is created by using Google Form. The questionnaire URL link for this survey is https://forms.gle/jGwcJBUeYsgS9Mtv5.

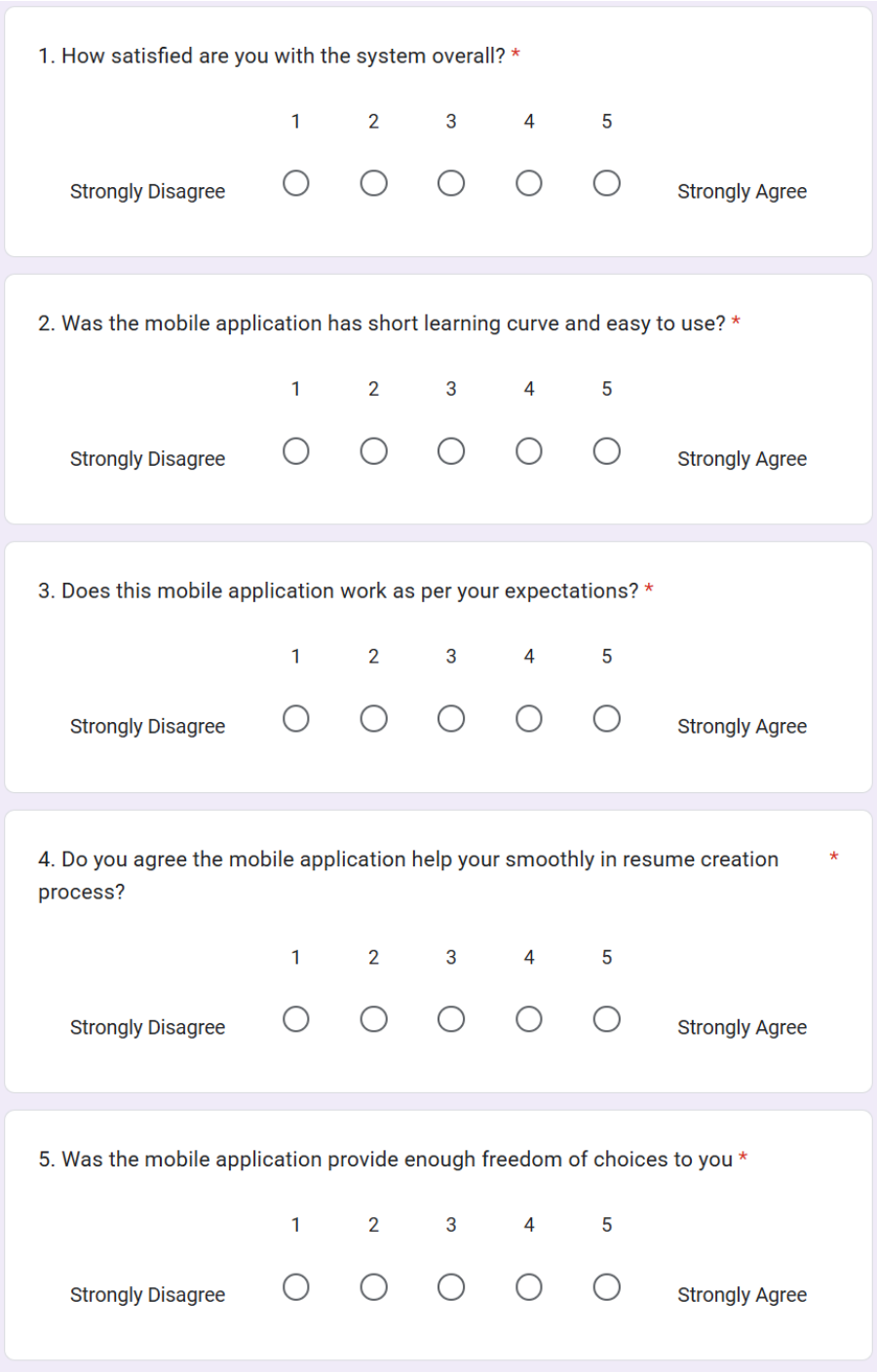


Figure A-1: First 5 Questions in the Survey.

6. Does the mobile application solve your question on career advisory? *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

7. Do you agree the integrated AI helps the features effectively and efficiently? *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

8. Do you satisfied with this mobile application UI interface? *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

9. Do you think the mobile application has enough features? *

	1	2	3	4	5	
Strongly Disagree	○	○	○	○	○	Strongly Agree

10. Any suggestion for further improvements? *

○ No

○ Other: _____

Figure A-2: Next 5 Questions in the Survey.

Appendix B: UAT Survey Results

These figures show the result that gained from the survey after UAT testing of this application.

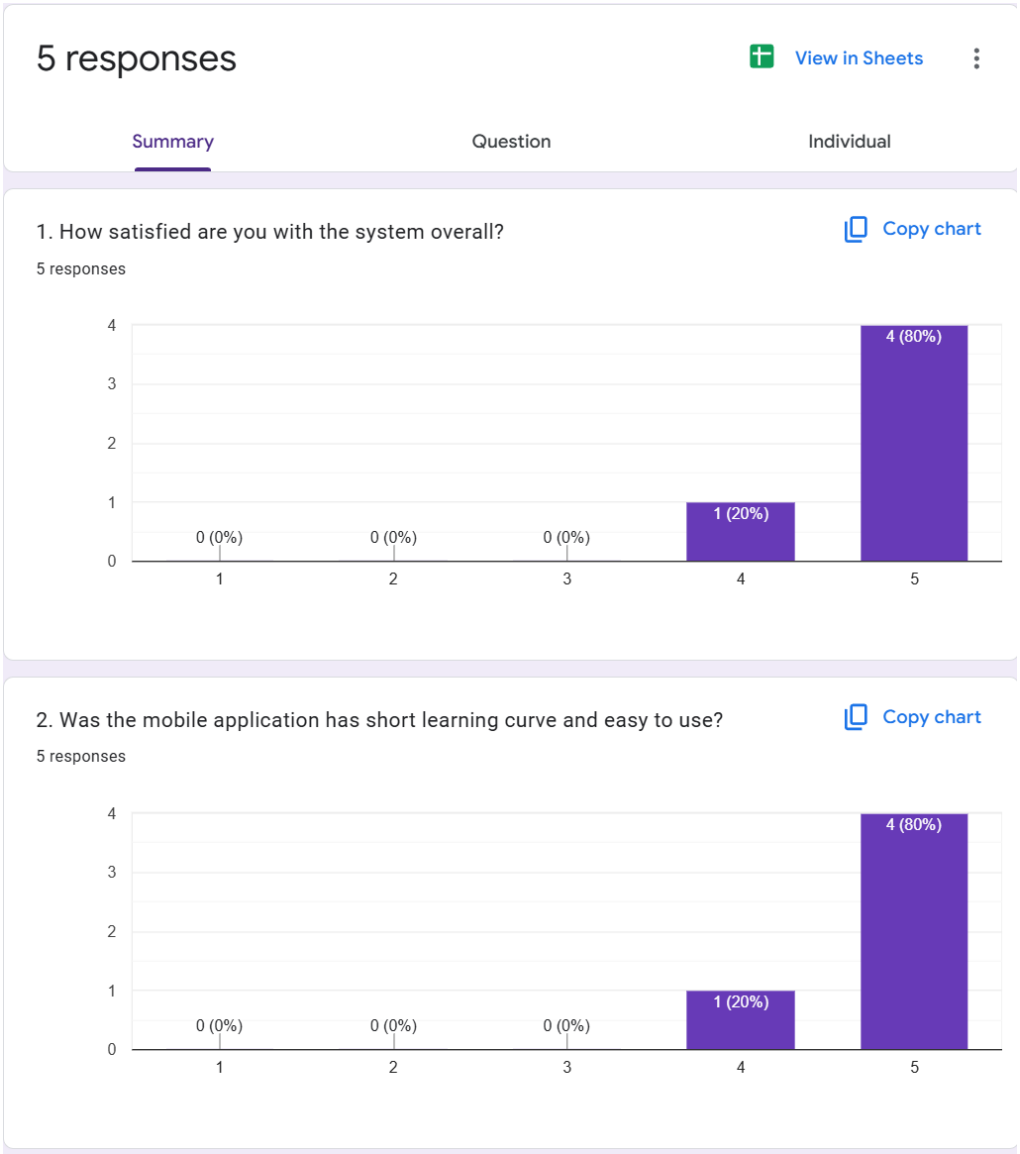


Figure B-1: Result for Question 1 and 2.

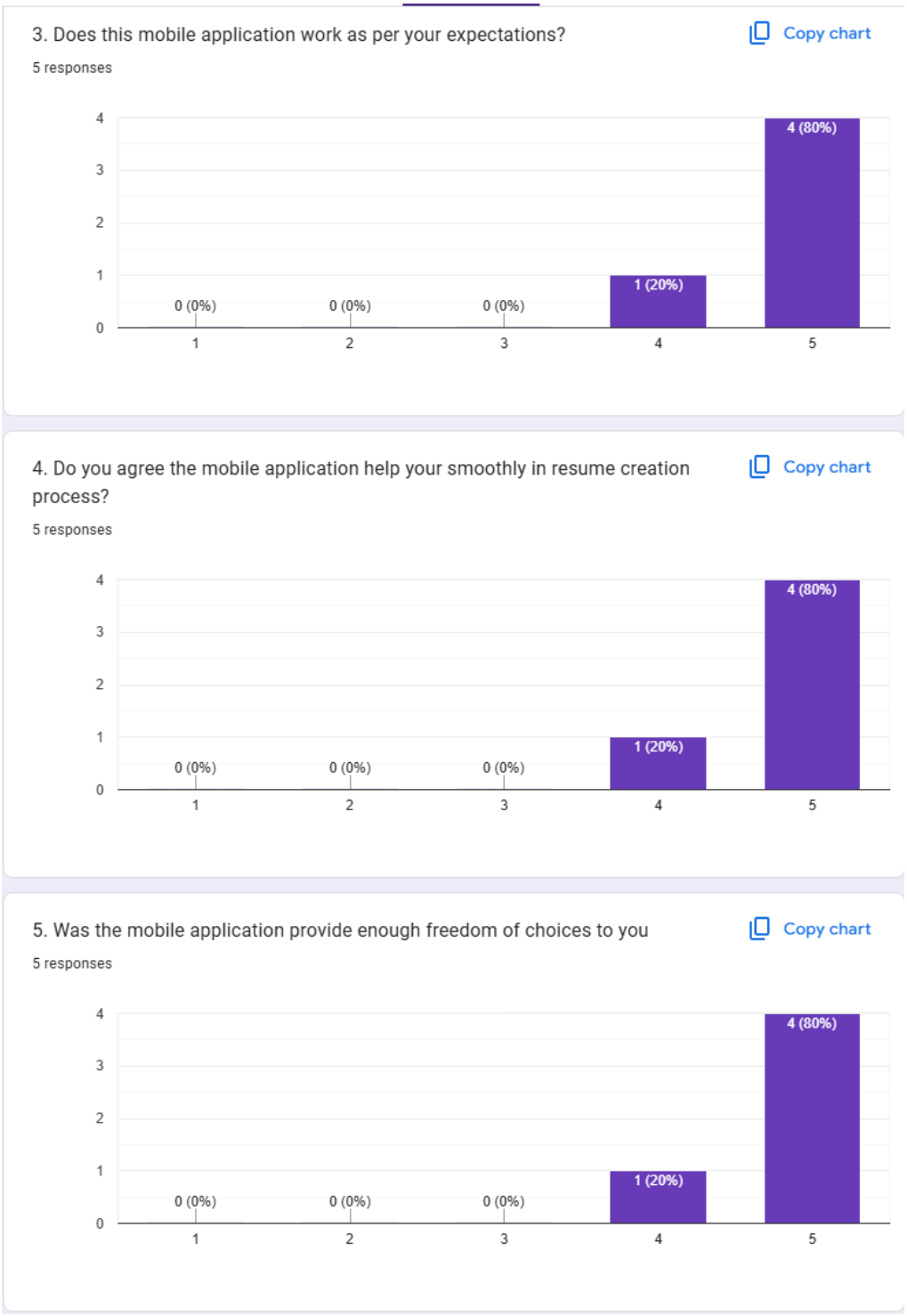


Figure B-2: Result for Question 3, 4 and 5.

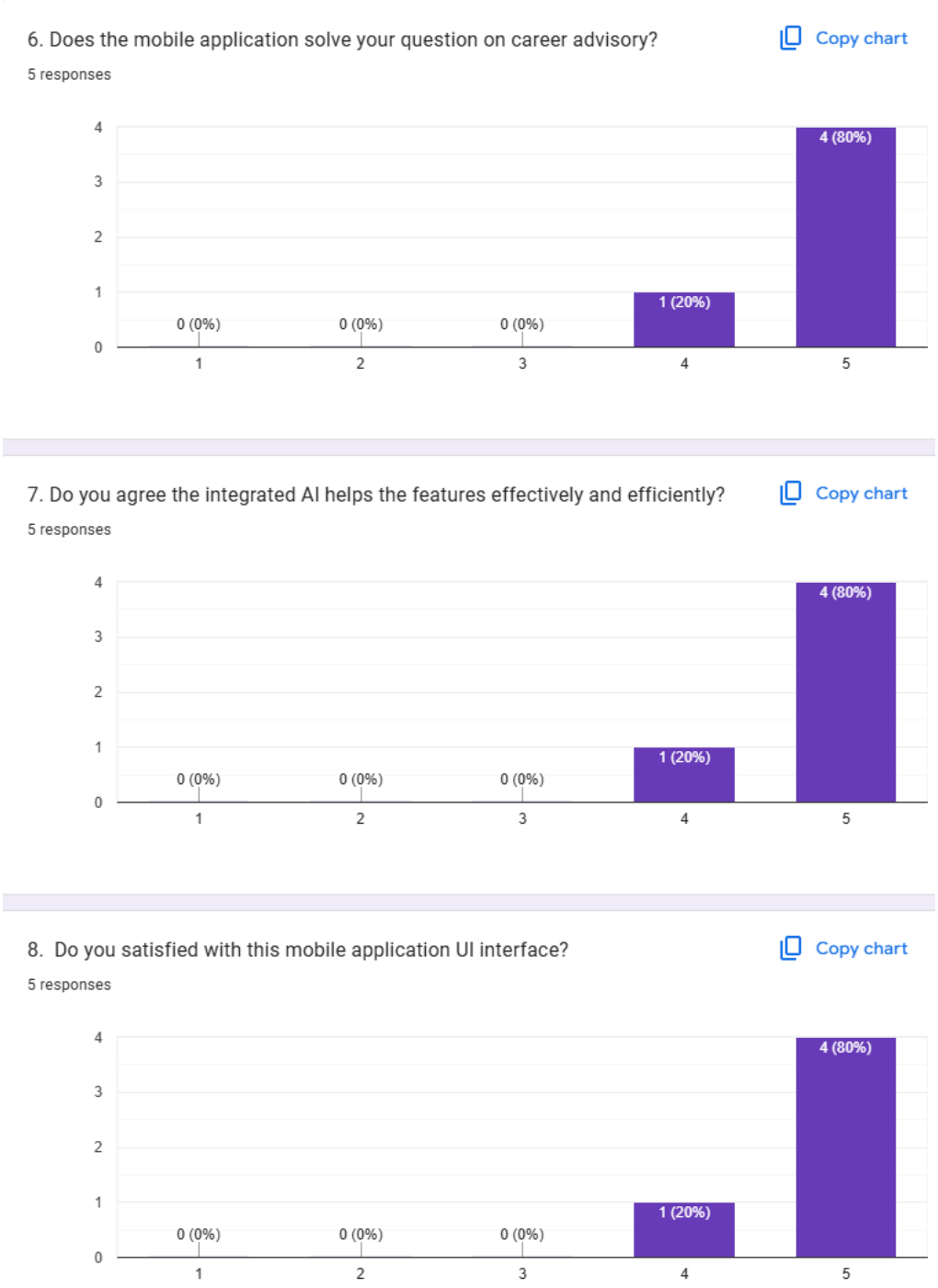


Figure B-3: Result for Question 6, 7 and 8.

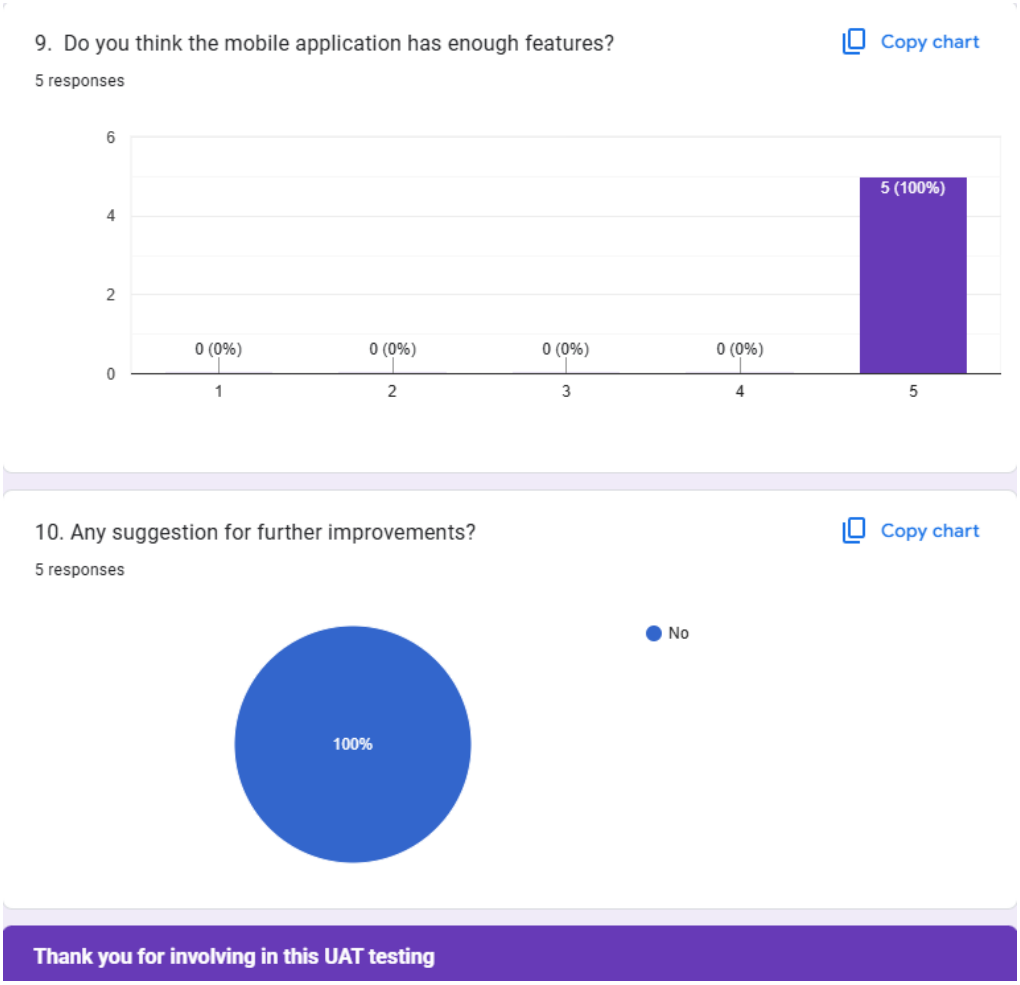


Figure B-4: Result for Question 9 and 10 with the survey ending.