

**CAMPUS RIDE APPLICATION**



**UNIVERSITI TEKNIKAL MALAYSIA MELAKA**

## CAMPUS RIDE APPLICATION



This report is submitted in partial fulfillment of the requirements for the Bachelor of [Computer Science (Software Development)] with Honours.

FACULTY OF INFORMATION AND COMMUNICATION TECHNOLOGY  
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

2024

## DECLARATION

I hereby declare that this project report entitled

### [CAMPUS RIDE APPLICATION]

is written by me and is my own effort and that no part has been plagiarized

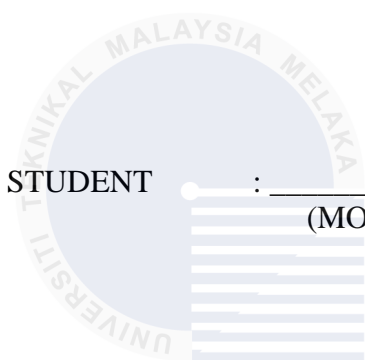
without citations.

STUDENT

:

  
(MOHAMAD FIKRI BIN AHMAD FADZIL)

Date : 09 June 2024



اونيورسي تيكنيكل مليسيا ملاك

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

I hereby declare that I have read this project report and found

this project report is sufficient in term of the scope and quality for the award of

Bachelor of [Computer Science (Software Development)] with Honours.

SUPERVISOR

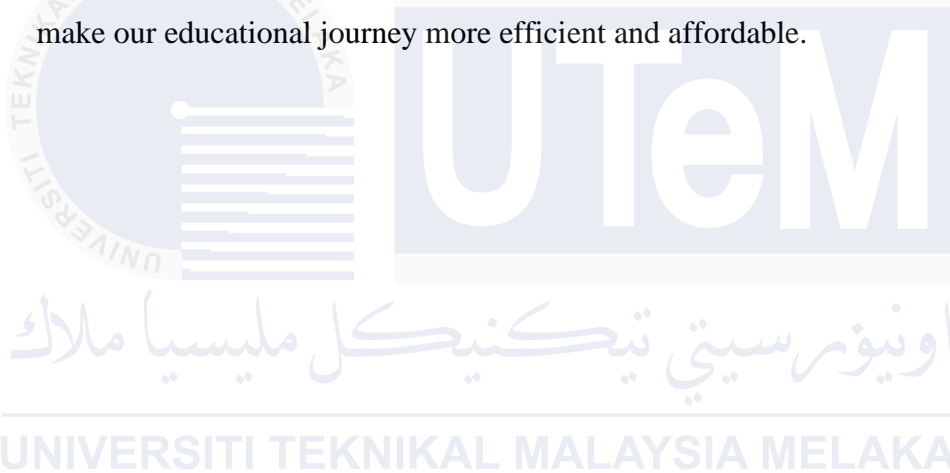
:

  
TS. DR. LIZAWATI SALAHUDDIN

Date : 29/08/2024

## DEDICATION

By the time, I would like to express my gratitude to Almighty Allah for granting me the strength and determination to plan, develop and complete this project by the end of the semester. This project is dedicated to my loving family and friends, whose unwavering support and encouragement have been my guiding light throughout this journey. To my esteemed advisor, especially my supervisor, whose wisdom, guidance and patience have been invaluable. To my fellow students, who inspire me every day with their determination and perseverance. This project is for all of us who strive to make our educational journey more efficient and affordable.



## ACKNOWLEDGEMENTS

I extend my deepest gratitude to my academic supervisor, Ts. Dr. Lizawati Binti Salahuddin, for her unwavering support, guidance, and insightful feedback throughout this project. Her expertise and encouragement have been invaluable. Special thanks to my evaluators, Ts. Dr. Kasturi A/P Kanchymalay for their insightful feedback and constructive criticism, which have significantly contributed to the refinement and success of this work. I am profoundly grateful to my parents for their unwavering love, encouragement, and belief in me. Their support has been the foundation of my success and has motivated me to persevere through challenges. I also wish to acknowledge my friends for their steadfast support, encouragement, and understanding during both the triumphs and challenges of this journey. Their presence has made the completion of FYP 1 more meaningful and memorable. Lastly, I express my heartfelt gratitude to myself for the unwavering determination and countless hours of hard work invested in this endeavor. This accomplishment stands as a testament to perseverance and dedication.

## ABSTRACT

This study focuses on e-hailing services, akin to conventional services like Grab and Maxim. It addresses transportation challenges faced by students and leverages shared ride services to improve efficiency and reduce costs. Frequently, students post on unofficial university social media platforms seeking to share rides to save money. The proposed solution is a system like current e-hailing services but with an added sharing option, enabling students to easily find others to share rides with. The project process involves analyzing existing studies on shared mobility and ride-sharing solutions in academic environments, followed by designing and developing a prototype mobile application featuring the ride-sharing option. The project was developed using an agile methodology. The application was built with Flutter, and data is stored in Google Firebase, an online database. The testing strategy involved getting user feedback by demonstrating the application in an online meeting and providing a video for users to watch and give their feedback. The final project has seven modules: user authentication, wallet, ride history, search, sharing, live tracking, and Google APIs. After testing, most users said that the application is easy to use, helps them save money on e-hailing, and they enjoy using it. The implementation of this project is expected to significantly reduce transportation costs for students and enhance efficiency with higher ride occupancy rates, thereby decreasing the overall number of rides needed.

## ABSTRAK

Kajian ini memberi tumpuan kepada perkhidmatan e-hailing, seakan-akan perkhidmatan konvensional seperti Grab dan Maxim. Ia menangani cabaran pengangkutan yang dihadapi oleh pelajar dan menggunakan perkhidmatan perkongsian perjalanan untuk meningkatkan kecekapan dan mengurangkan kos. Selalunya, pelajar memuat naik permintaan di platform media sosial universiti tidak rasmi untuk mencari rakan yang boleh berkongsi perjalanan bagi menjimatkan kos. Penyelesaian yang dicadangkan adalah satu sistem seperti perkhidmatan e-hailing semasa tetapi dengan pilihan perkongsian tambahan, yang membolehkan pelajar dengan mudah mencari rakan untuk berkongsi perjalanan. Proses projek ini melibatkan analisis kajian sedia ada mengenai mobiliti bersama dan penyelesaian perkongsian perjalanan dalam persekitaran akademik, diikuti dengan reka bentuk dan pembangunan prototaip aplikasi mudah alih yang menawarkan pilihan perkongsian perjalanan. Projek ini dibangunkan menggunakan metodologi agile. Aplikasi ini dibina dengan Flutter, dan data disimpan dalam Google Firebase, sebuah pangkalan data dalam talian. Strategi pengujian melibatkan mendapatkan maklum balas pengguna melalui demonstrasi aplikasi dalam mesyuarat dalam talian dan menyediakan video untuk pengguna menonton serta memberi maklum balas. Projek akhir ini mempunyai tujuh modul: pengesahan pengguna, dompet, sejarah perjalanan, carian, perkongsian, penjejakan secara langsung, dan Google APIs. Selepas ujian, kebanyakan pengguna menyatakan bahawa aplikasi ini mudah digunakan, membantu mereka menjimatkan wang untuk e-hailing, dan mereka menikmati menggunakannya. Pelaksanaan projek ini dijangka dapat mengurangkan kos pengangkutan bagi pelajar secara signifikan dan meningkatkan kecekapan dengan kadar penggunaan perjalanan yang lebih tinggi, sekaligus mengurangkan jumlah perjalanan yang diperlukan.

## TABLE OF CONTENTS

	<b>PAGE</b>
<b>DECLARATION.....</b>	<b>II</b>
<b>DEDICATION.....</b>	<b>III</b>
<b>ACKNOWLEDGEMENTS.....</b>	<b>IV</b>
<b>ABSTRACT .....</b>	<b>V</b>
<b>ABSTRAK .....</b>	<b>VI</b>
<b>TABLE OF CONTENTS.....</b>	<b>VII</b>
<b>LIST OF TABLES .....</b>	<b>XIII</b>
<b>LIST OF FIGURES .....</b>	<b>XV</b>
<b>LIST OF ABBREVIATIONS .....</b>	<b>XVIII</b>
<b>LIST OF ATTACHMENTS.....</b>	<b>XIXI</b>
<b>CHAPTER 1: INTRODUCTION.....</b>	<b>11</b>
1.1    Introduction.....	1
1.2    Problem Statements .....	2
1.3    Objectives .....	2
1.4    Scopes .....	3
1.4.1    Modules to be developed.....	3
1.4.2    Target User .....	4
1.5    Project Significance .....	4
1.6    Expected Output.....	5



1.7	Conclusion .....	6
<b>CHAPTER 2: LITERATURE REVIEW AND PROJECT METHODOLOGY . 7</b>		
2.1	Introduction.....	7
2.2	Facts and Findings .....	7
2.2.1	Domain .....	8
2.2.2	Existing System .....	8
2.2.3	Technique .....	10
2.3	Project Methodology.....	10
2.4	Project Requirement.....	12
2.4.1	Software Requirements.....	12
2.4.2	Hardware Requirements .....	14
2.5	Project Schedules and Milestones.....	15
2.6	Conclusion .....	16
<b>CHAPTER 3: ANALYSIS..... 17</b>		
3.1	Introduction.....	17
3.2	Problem Analysis .....	17
3.2.1	Overview of Current System .....	17
3.2.2	Overview of Proposed System.....	18
3.3	Requirement Analysis.....	21
3.3.1	Data Requirements.....	22
3.3.1.1	Data Dictionary.....	22
3.3.2	Functional Requirements .....	25
3.3.3	Non-Functional Requirements.....	26

3.3.4	Use Case Diagram .....	27
3.3.5	Sequence Diagram .....	28
3.4	Conclusion .....	28
<b>CHAPTER 4: DESIGN .....</b>		<b>29</b>
4.1	Introduction.....	29
4.2	High Level Design .....	29
4.2.1	System Architecture.....	29
4.2.2	User Interface Design .....	30
4.2.3	Conceptual and Logical Design.....	44
4.3	Conceptual and Logical Database Design .....	45
4.3.1	Software Design.....	46
4.3.2	Physical Design .....	47
4.4	Conclusion .....	48
<b>CHAPTER 5: IMPLEMENTATION.....</b>		<b>29</b>
5.1	Introduction.....	49
5.2	Software Development Environment Setup.....	49
5.2.1	Android Studio.....	49
5.2.2	Google Firebase .....	50
5.2.3	Programming Language.....	51
5.2.4	Environment Architecture .....	51
5.3	Software Configuration Management.....	52
5.3.1	Installation and Setup of Android Studio .....	52
5.3.2	Flutter Setup.....	58
5.3.3	Google Firebase Database Setup .....	61

5.4	Version Control Procedure .....	63
5.5	Implementation Status .....	64
5.6	Conclusion .....	65
<b>CHAPTER 6: IMPLEMENTATION.....</b>		<b>29</b>
6.1	Introduction.....	66
6.2	Test Plan.....	66
6.2.1	Test Organization.....	66
6.2.2	Test Environment.....	67
6.2.2.1	Environment Setup .....	68
6.2.2.2	Application Software .....	68
6.2.2.3	System Software .....	68
6.2.2.4	System Hardware.....	69
6.2.3	Test Schedule.....	69
6.3	Test Strategy .....	70
6.3.1	Dynamic Testing.....	70
6.3.2	User Acceptance Testing .....	70
6.4	Test Design .....	71
6.4.1	Test Description.....	71
6.4.1.1	Test Description for User Authentication.....	72
6.4.1.2	Test Description for Wallet Module .....	74
6.4.1.3	Test Description for Ride History Module .....	75
6.4.1.4	Test Description for Search Module.....	76
6.4.1.1	Test Description for Sharing Module .....	77
6.4.1.2	Test Description for Live Tracking Module.....	79
6.4.1.3	Test Description for Google APIs .....	80

6.4.2	Test Data for Dynamic Testing.....	81
6.4.2.1	Test Data for User Authentication .....	81
6.4.2.2	Test Data for Wallet Module .....	86
6.4.2.3	Test Data for Ride History Module .....	88
6.4.2.4	Test Data for Search Module .....	89
6.4.2.5	Test Data for Sharing Module .....	90
6.4.2.6	Test Data for Live Tracking Module .....	94
6.4.2.7	Test Data for Google APIs .....	96
6.5	User Acceptance Testing .....	97
6.5.1	Questionnaires for User Acceptance Testing.....	97
6.6	Test Result and Analysis.....	99
6.6.1	Test Result for Dynamic Testing .....	99
6.6.1.1	Test Result for User Authentication .....	99
6.6.1.2	Test Result for Wallet Module .....	100
6.6.1.3	Test Result for Ride History Module.....	100
6.6.1.4	Test Result for Search Module .....	101
6.6.1.5	Test Result for Sharing Module.....	101
6.6.1.6	Test Result for Live Tracking Module .....	102
6.6.1.7	Test Result for Google APIs.....	102
6.6.1.8	Summary of Recorded Test Case .....	102
6.6.2	User Acceptance Testing Analysis and Result .....	103
6.7	Conclusion .....	105
<b>CHAPTER 7: PROJECT CONCLUSION .....</b>		<b>29</b>
7.1	Introduction.....	106
7.2	Observation on Weakness and Strengths.....	106
7.2.1	System Strengths .....	106

7.2.2	System Weakness .....	107
7.3	Propositions for Improvement .....	107
7.4	Project Contribution.....	107
7.5	Conclusion .....	108
<b>REFERENCES.....</b>		<b>29</b>
<b>APPENDICES .....</b>		<b>29</b>



اونيورسيتي تيكنيكل مليسيا ملاك

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

## LIST OF TABLES

	PAGE
<b>Table 2.1: Table of comparison between current system and system developed..</b>	<b>9</b>
<b>Table 2.2: List of Hardware Requirement.....</b>	<b>14</b>
<b>Table 3.1: Firebase Authentication Dictionary .....</b>	<b>19</b>
<b>Table 3.2: User Data Dictionary .....</b>	<b>20</b>
<b>Table 3.3: Ride History Data Dictionary .....</b>	<b>20</b>
<b>Table 3.4: Wallet Data Dictionary .....</b>	<b>21</b>
<b>Table 3.5: Transaction Data Dictionary .....</b>	<b>21</b>
<b>Table 3.6: Driver Data Dictionary .....</b>	<b>22</b>
<b>Table 3.7: Functional Requirement.....</b>	<b>22</b>
<b>Table 3.8: Non-Functional Requirement .....</b>	<b>23</b>
<b>Table 6.1: Test Organization.....</b>	<b>67</b>
<b>Table 6.2: Application Software .....</b>	<b>68</b>
<b>Table 6.3: System Software .....</b>	<b>68</b>
<b>Table 6.4: System Hardware .....</b>	<b>69</b>
<b>Table 6.5: Test Schedule .....</b>	<b>69</b>
<b>Table 6.6: Test Case for User Authentication .....</b>	<b>73</b>
<b>Table 6.7: Test Case for Wallet Module.....</b>	<b>74</b>
<b>Table 6.8: Test Case for Ride History Module .....</b>	<b>75</b>
<b>Table 6.9: Test Case for Search Module .....</b>	<b>76</b>
<b>Table 6.10: Test Case for Sharing Module .....</b>	<b>77</b>
<b>Table 6.11: Test Case for Live Tracking.....</b>	<b>79</b>
<b>Table 6.12: Test Case for Google APIs .....</b>	<b>80</b>
<b>Table 6.13: User Acceptance Questoinnaires .....</b>	<b>97</b>
<b>Table 6.14: Test Result for User Authentication.....</b>	<b>99</b>

<b>Table 6.15: Test Case for Wallet Module.....</b>	<b>100</b>
<b>Table 6.16: Test Case for Ride History Module.....</b>	<b>100</b>
<b>Table 6.17: Test Case for Search Module .....</b>	<b>101</b>
<b>Table 6.18: Test Case for Sharing Module .....</b>	<b>101</b>
<b>Table 6.19: Test Case for Live Tracking.....</b>	<b>102</b>
<b>Table 6.20: Test Case for Google APIs .....</b>	<b>102</b>
<b>Table 6.21: Summary of Recorded Test Case .....</b>	<b>102</b>



اونيورسيتي تيكنيكل مليسيا ملاك

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

## LIST OF FIGURES

	PAGE
Figure 2.1: Screenshot of the Grab finding driver during peak hour.....	8
Figure 2.2: Screenshot of the Grab price during peak hour .....	9
Figure 2.3: Screenshot of the Procubex User Live Tracking .....	9
Figure 2.4: Agile Development Methodology .....	11
Figure 2.5: Android Studio IDE .....	13
Figure 2.6: Google Firebase .....	13
Figure 2.7: Gantt Chart for Campus Ride Application.....	16
Figure 3.1: Flowchart of the system (Part 1) .....	18
Figure 3.2: Flowchart of the system (Part 2) .....	19
Figure 3.3: Flowchart of the system (Part 3) .....	20
Figure 3.4: Use Case Diagram .....	27
Figure 3.5: Sequence Diagram .....	28
Figure 4.1: System Architecture of Campus Ride Application.....	30
Figure 4.2: Login Page.....	31
Figure 4.3: Register Page.....	32
Figure 4.4: Home Page.....	33
Figure 4.5: Drawer .....	34
Figure 4.6: Wallet Page.....	35
Figure 4.7: Setting Page.....	36
Figure 4.8: History Page .....	37
Figure 4.9: Searching Page.....	38
Figure 4.10: Request Page .....	39
Figure 4.11: Finding Driver Page .....	40
Figure 4.12: Driver Found Page .....	41



<b>Figure 4.13: Driver Arrived Confirmation Page</b> .....	42
<b>Figure 4.14: Live Tracking Page</b> .....	43
<b>Figure 4.15: Conceptual Database Design</b> .....	44
<b>Figure 4.16: Logical Database Design</b> .....	45
<b>Figure 4.17: Software Design</b> .....	46
<b>Figure 4.18: Entity Relationship Diagram</b> .....	47
<b>Figure 5.1: Android Studio IDE</b> .....	49
<b>Figure 5.2: Google Firebase</b> .....	50
<b>Figure 5.3: Flutter Programming Language</b> .....	51
<b>Figure 5.4: Google APIs</b> .....	51
<b>Figure 5.5: Places API</b> .....	52
<b>Figure 5.6: Cloud Firestore API</b> .....	52
<b>Figure 5.7: Map SDK For Android</b> .....	53
<b>Figure 5.8: Identity Toolkit API</b> .....	53
<b>Figure 5.9: Geocoding API</b> .....	54
<b>Figure 5.10: Directions API</b> .....	54
<b>Figure 5.11: Firebase Cloud Messaging API</b> .....	55
<b>Figure 5.12: Firebase Installations API</b> .....	55
<b>Figure 5.13: Environment Architecture</b> .....	56
<b>Figure 5.14: Android Studio Download Page</b> .....	56
<b>Figure 5.15: Android Studio Setup I</b> .....	57
<b>Figure 5.16: Android Studio Setup II</b> .....	57
<b>Figure 5.17: Android Studio Setup III</b> .....	58
<b>Figure 5.18: Android Studio Setup IV</b> .....	58
<b>Figure 5.19: Android Studio Setup V</b> .....	59
<b>Figure 5.20: Android Studio Setup VI</b> .....	59
<b>Figure 5.21: Android Studio Setup VII</b> .....	60
<b>Figure 5.22: Android Studio Setup VIII</b> .....	60
<b>Figure 5.23: Android Studio Setup IX</b> .....	61
<b>Figure 5.24: Android Studio Setup X</b> .....	61
<b>Figure 5.25: Android Studio Setup XI</b> .....	62
<b>Figure 5.26: Flutter Setup I</b> .....	62
<b>Figure 5.27: Flutter Setup II</b> .....	63
<b>Figure 5.28: Flutter Setup III</b> .....	63

<b>Figure 5.29: Flutter Setup IV</b> .....	64
<b>Figure 5.30: Flutter Setup V</b> .....	64
<b>Figure 5.31: Flutter Setup VI</b> .....	65
<b>Figure 5.32: Flutter Setup VII</b> .....	65
<b>Figure 5.33: Google Firebase Database Setup I</b> .....	66
<b>Figure 5.34: Google Firebase Database Setup II</b> .....	66
<b>Figure 5.35: Google Firebase Database Setup III</b> .....	66
<b>Figure 5.36: Google Firebase Database Setup IV</b> .....	67
<b>Figure 5.37: Google Firebase Database Setup V</b> .....	67
<b>Figure 5.38: Google Firebase Database Setup VI</b> .....	67
<b>Figure 6.1: Pie Chart of Questionnaire Question</b> .....	107
<b>Figure 6.2: Bar Chart of Questionnaire Question</b> .....	108
<b>Figure 6.3: Bar Chart of Questionnaire Question</b> .....	108
<b>Figure 6.4: Bar Chart of Questionnaire Question</b> .....	109
<b>Figure 6.5: Bar Chart of Questionnaire Question</b> .....	109
<b>Figure 6.6: Bar Chart of Questionnaire Question</b> .....	110
<b>Figure 6.7: Bar Chart of Questionnaire Question</b> .....	110
<b>Figure 6.8: Bar Chart of Questionnaire Question</b> .....	111
<b>Figure 6.9: Bar Chart of Questionnaire Question</b> .....	111
<b>Figure 6.10: Bar Chart of Questionnaire Question</b> .....	112
<b>Figure 6.11: Bar Chart of Questionnaire Question</b> .....	112
<b>Figure 6.12: Bar Chart of Questionnaire Question</b> .....	113
<b>Figure 6.13: Bar Chart of Questionnaire Question</b> .....	113
<b>Figure 6.14: Bar Chart of Questionnaire Question</b> .....	114
<b>Figure 6.15: Bar Chart of Questionnaire Question</b> .....	114
<b>Figure 6.16: Bar Chart of Questionnaire Question</b> .....	115
<b>Figure 6.17: Bar Chart of Questionnaire Question</b> .....	115
<b>Figure 6.18: Bar Chart of Questionnaire Question</b> .....	116
<b>Figure 6.19: Bar Chart of Questionnaire Question</b> .....	116
<b>Figure 6.20: Bar Chart of Questionnaire Question</b> .....	117
<b>Figure 6.21: Bar Chart of Questionnaire Question</b> .....	117
<b>Figure 6.22: Bar Chart of Questionnaire Question</b> .....	118
<b>Figure 6.23: Bar Chart of Questionnaire Question</b> .....	118

## LIST OF ABBREVIATIONS

**FYP** - **Final Year Project**



## LIST OF ATTACHMENTS

	<b>PAGE</b>
<b>Appendix A: Demographic (User Information)</b> .....	124
<b>Appendix B: Demographic (Perceived Ease of Use)</b> .....	125
<b>Appendix C: Demographic (Perceived Usefulness)</b> .....	126
<b>Appendix D: Demographic (Capability)</b> .....	127
<b>Appendix E: Demographic (Trustworthiness)</b> .....	128
<b>Appendix F: Demographic (Attitude)</b> .....	129
<b>Appendix G: Demographic (Intention to Use)</b> .....	130

## CHAPTER 1: INTRODUCTION

### 1.1 Introduction

In recent years, the rise of e-hailing services such as Grab and Maxim has revolutionized urban transportation, providing convenient and efficient travel options for users. However, despite the convenience, individual rides can be costly, particularly for students who often operate on tight budgets. This project aims to address the transportation challenges faced by students by introducing a shared ride feature into the existing e-hailing framework. By enabling students to share rides, this system seeks to reduce transportation costs and improve overall efficiency.

The background of this project stems from the observation that students frequently use unofficial university social media platforms to coordinate shared rides informally. While this method has helped some students save on transportation costs, it is often inefficient and lacks a structured approach. Therefore, the proposed solution is to develop a dedicated system within the e-hailing service that facilitates ridesharing among students.

This project involves analyzing current shared mobility and ride-sharing solutions in academic environments, designing a user-friendly mobile application prototype, and implementing the ride-sharing feature. The goal is to create a sustainable and cost-effective transportation solution for students, enhancing their mobility while reducing the financial burden associated with individual rides. Through this innovative approach, the project aims to foster a more collaborative and economical transportation system within the university community.

## 1.2 Problem Statements

Existing hailing services present several challenges for students, particularly in terms of convenience, expense, and reliability. The high cost associated with individual rides often places a significant financial strain on students, who typically have limited financial resources. Furthermore, the inconsistency and unreliability of these services can lead to delays and missed appointments, complicating students' schedules and daily routines. These issues underscore the necessity for a more cost-effective and reliable transportation solution specifically tailored to the needs of students.

Additionally, current e-hailing services lack features and discounts that would attract students to use them more frequently. Many of these services do not offer special pricing, or group ride options that cater to the student demographic, who are consistently seeking ways to save money. The absence of such incentives reduces the appeal of these services among students. This gap reveals a significant opportunity for the development of a ride-sharing system that incorporates features specifically designed to meet the financial and practical needs of students.

Finally, safety concerns remain a significant issue for students who utilize e-hailing services. Reports of safety incidents and a general lack of trust in the security measures provided by current services deter students from relying on them for their transportation needs. Safety is a paramount concern for students, particularly those traveling at night or in unfamiliar areas. Addressing these safety concerns through new security features for increasing student usage and confidence in e-hailing services.

## 1.3 Objectives

- I. To design a solution that facilitates students in getting e-hailing services and enabling ridesharing for cost effectiveness and community building.
- II. To develop a student e-hailing service application using Flutter and Firebase, integrating additional features such as live tracking and an emergency button to enhance the transportation experience.

III. To test the functionality of the developed system.

## 1.4 Scopes

### 1.4.1 Modules to be developed.

#### I. Login Module

This module allows registered users to access the application by entering their matric number and password.

#### II. Register Module

This module will register the new user by providing necessary information and storing the data into the system database.

#### III. Searching Module

This module, user needs to input drop off locations and finding the available driver to fulfill the requests.

#### IV. Wallet Module

This module allows users to manage their money within the app. User can make payment using the funds in the wallets, reload the wallet, check their balance and view transactions history.

#### V. Ride History Module

This module provides users with a detailed record of their past rides. It includes information such as date, time, pickup and drop-off locations.

## VI. Live Tracking Module

This module offers real-time tracking of the ride. Users can see the exact location of their ride until they reach the destination.

## VII. Sharing Module

This module helps users to find users who want to share their ride who have similar routes.

## VIII. Emergency Module

This module has an emergency button that users can press to alert the pre-selected contacts just in case of danger.

## IX. Rating Module

This module allows users to rate their ride experience and provide feedback on drivers.

### 1.4.2 Target Users

The Campus Ride Application is designed for students who want to share rides to reduce costs. Users can search for their destination, and the system will help them find other users with the same pick-up and drop-off locations. If both parties agree to share the ride, they can split the cost, making it more affordable for everyone involved.

## 1.5 Project Significance

The idea for the Campus Ride Application came from the common transportation problems that students face. Many students find it hard to afford transportation costs while also managing their studies and personal lives. Seeing many students post on university social media to find ride-sharing partners showed the need for a better solution. Additionally, the increasing focus on protecting the environment and reducing pollution inspired the creation of a ride-sharing system to help make



campuses greener. This platform aims to make transportation easier, cheaper, and more eco-friendly, improving the overall student experience.

## **1.6 Expected Output**

The Campus Ride Application aims to meet the project's goals and solve existing problems with smart solutions. With the ride-sharing system, students don't have to post on unofficial university social media to find someone to share a ride with. The app will automatically search its database to find suitable matches based on the users' ride details like departure, destination, and preferred times. When the system finds potential matches, it sends notifications to the users, asking if they want to share the ride. This automatic matching saves users time and effort, making it easier to find ride-sharing partners. If both users accept the ride-sharing offer, the app helps arrange the ride, ensuring they both pay a reduced fare. This cost-saving feature helps students manage their transportation expenses better, allowing them to use their budget for other important needs. The Campus Ride Application makes transportation more convenient and affordable for the student community.

## **1.7 Conclusion**

The Campus Ride Application effectively addresses students' transportation challenges by offering a convenient and cost-effective ride-sharing solution. By automating the matching process and reducing fares, the app saves time and money, promoting community and sustainability. This user-friendly tool enhances the overall transportation experience for students.

## **CHAPTER 2: LITERATURE REVIEW AND PROJECT METHODOLOGY**

### **2.1 Introduction**

A literature review is an academic writing task that situates and showcases a deep understanding of the academic literature on a particular topic. Unlike a mere literature report, a literature review involves a critical evaluation of the sources. It encompasses both the process of reading and writing about literature. On the other hand, project methodology, also known as System Development Methodology (SDM), is a standardized process used within organizations to complete all essential phases of the software system life cycle. This includes planning, analysis, design, development, implementation, testing, and maintenance. It is crucial for enterprises to employ a systems development methodology when creating new systems.

### **2.2 Facts and Findings**

The technique of gathering facts and findings in project management involves the systematic collection of data and information from various sources, including existing documents, research materials, observations, and prototypes. These facts and findings are typically used to enhance the current system under development. Employing fact-finding techniques is essential in the System Development Life Cycle, as it facilitates the efficient and effective extraction of relevant information. This section emphasizes the collection of information through comprehensive research and studies. In this chapter, the facts and findings pertaining to the Campus Ride Application will be detailed.

### **2.2.1 Domain**

The Campus Ride Application operates within the transportation domain, specifically focusing on student transportation needs. It addresses the unique requirements of students for cost-effective and efficient ride-sharing solutions, allowing users to search for nearby ride-sharing opportunities, schedule shared rides, and receive information about ride-sharing options. By facilitating the efficient management of ridesharing, the application helps reduce transportation costs and improves convenience for students.

Additionally, the Campus Ride Application incorporates e-hailing services, providing an on-demand transportation option where users can book rides through the app. This integration ensures students have access to various transportation options, including the ability to hail a ride instantly when no scheduled ride-sharing opportunities are available. E-hailing services offer flexibility and convenience, making it easier for students to get to their destinations promptly and safely.

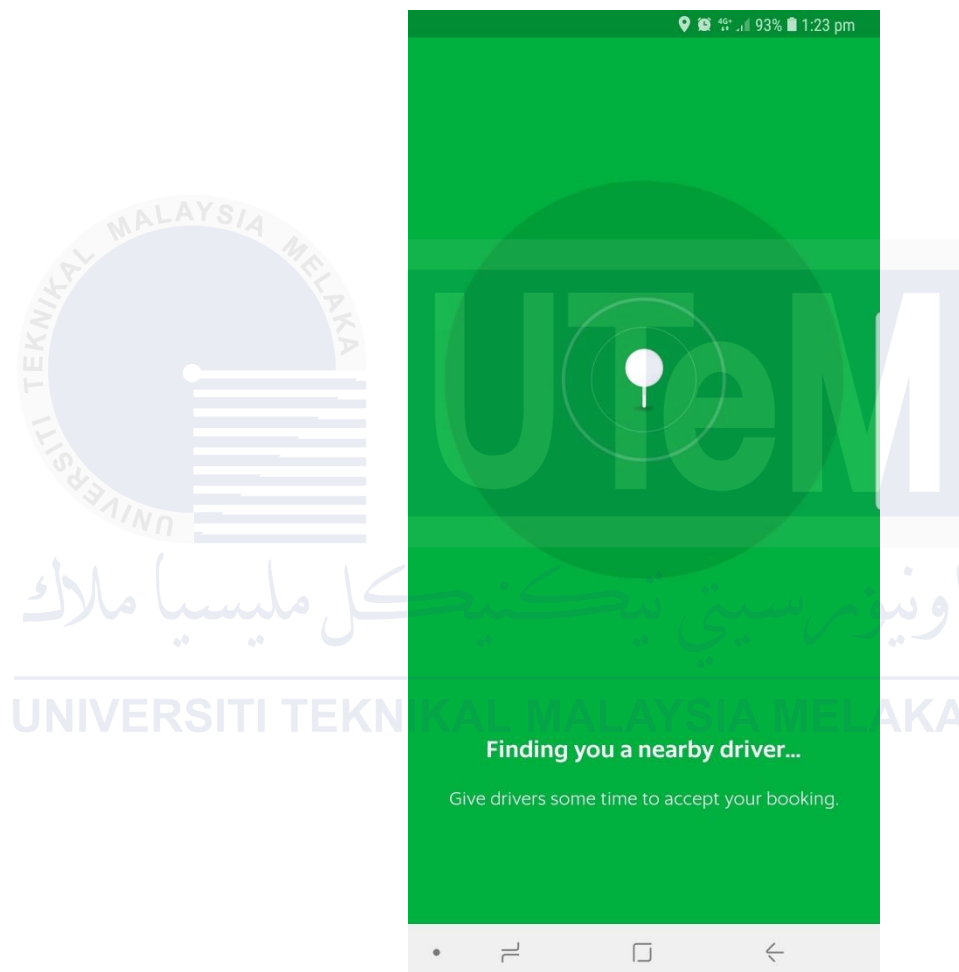
The application also raises awareness about the benefits of ride-sharing and e-hailing, encouraging students to participate in shared transportation. By educating the student body about the economic and environmental advantages of these services and highlighting their role in fostering community connections, the Campus Ride Application promotes sustainable transportation practices and enhances student engagement.

### **2.2.2 Existing System**

The Campus Ride Application is a comprehensive system designed to enable students to share their rides, thereby reducing transportation costs and helping them save money. This innovative solution addresses a gap in the current e-hailing services, which typically do not offer ride-sharing features tailored to the needs of students.

As an example, if we take the currently used e-hailing service called Grab, and a project by V3Cube where the application called Procubex User, there are some weaknesses in this system which can be improvised in the Campus Ride. Grab system will be very difficult to book during peak hour. This will result in long waiting times

causing inconvenience and frustration, especially for students who have tight schedules for classes and other commitments. For Procubex User, although it has a live tracking function, it lacks an emergency call button. If anything were to happen to a student, having an emergency button would be extremely helpful, allowing students to make a call even when they are scared. As an example, the figure 2.1 show that finding driver take almost one hour.



**Figure 2.1: Screenshot of the Grab finding driver during peak hour**

Next is the unjust fare during surge. This problem is the fares are increasing significantly during periods of high demand. This surge in pricing can lead to prohibitively expensive fares for students who may already be operating on limited budgets. Campus Ride Application offers more stable and predictable pricing by promoting shared rides. As example, figure 2.2 showing that the price for 10Km route that usually costs RM17 is increased to RM44 because of high demand.



Figure 2.2: Screenshot of the Grab price during peak hour

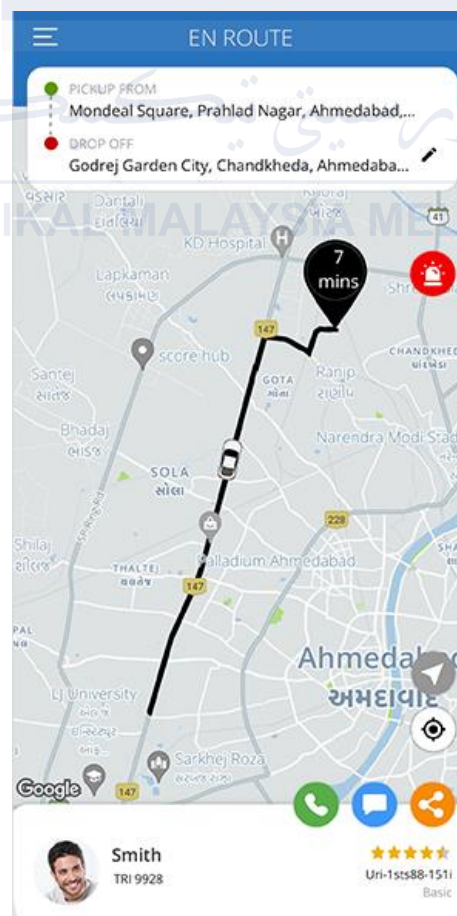


Figure 2.3: Screenshot of the Procubex User Live Tracking

**Table 2.1 Table of comparison between current system and system developed**

	Grab	Procubex User	Campus Ride Application
Shared features	No	No	Yes
Difficulty to book during peak hour	Yes	Yes	No
Fares pricing	Can be increased during peak hour	More stable and difficult to increase	More stable and difficult to increase
Live Tracking	No	No	Yes
Emergency Button	No	No	Yes

### 2.2.3 Technique

One technique to overcome the problems of the existing system is to implement ride-sharing features. This allows users to share their rides, thereby decreasing the overall cost.

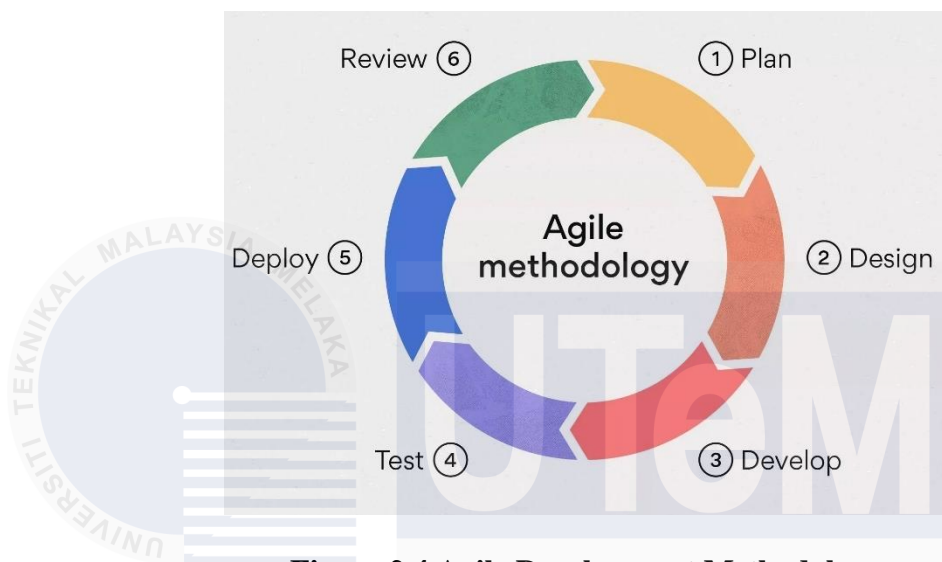
Another solution is to design the system to automatically reject requests when demand exceeds supply. This means that if a driver is busy, the system will promptly reject the user's request, preventing users from wasting time waiting for driver confirmation.

Additionally, the system will ensure that prices remain consistent, even during peak hours, so users are not subjected to higher fares when demand is high.

## 2.3 Project Methodology

There are several system development methods, like Agile, DevOps, Waterfall, and Rapid Application Development. The Agile Development methodology is best for

this project because it allows for small, manageable steps, making it easy to adapt to changes based on student needs. Agile focuses on continuous improvement, collaboration, and regular user feedback, ensuring the application meets student requirements effectively. This approach also enables quicker releases of functional features, helping to address transportation issues for students sooner.



**Figure 2.4 Agile Development Methodology**

Figure 2.3 shows the model of Agile Development Methodology, which consists of six steps: plan, design, develop, test, deploy, and review (What is Agile Methodology, 2024). In the planning phase, the project team identifies the goals, requirements, and tasks needed to develop the student e-hailing service. This involves understanding the specific needs of students, such as cost-effective ride sharing, live tracking, and emergency features. The team creates a detailed project roadmap, outlining the steps required to achieve these goals and setting clear milestones for progress.

During the design phase, the team focuses on creating a blueprint for the application. This includes designing the user interface (UI) and user experience (UX) to ensure the app is user-friendly and intuitive for students. The design process involves creating wireframes, mockups, and prototypes to visualize how the app will look and function, making sure it meets the students' expectations and preferences.

In the development phase, the team begins coding the application using Flutter and Firebase. This step involves building the core features of the app, such as the ride-sharing functionality, user accounts, and integration with mapping services for live

tracking. The developers work iteratively, focusing on small, manageable pieces of work to ensure steady progress and maintain flexibility to adapt to changes.

The testing phase involves rigorously checking the functionality of the application to ensure it works as intended. The team tests for bugs, performance issues, and security vulnerabilities. They also gather feedback from a small group of students to identify any usability issues and refine the app's features. This step is crucial to ensure the app provides a smooth and reliable experience for users.

Once the application has been thoroughly tested, it is deployed to a live environment where students can start using it. The team ensures that the app is available for download on relevant app stores and monitors its initial performance. This phase also includes setting up the necessary infrastructure to support the app and addressing any issues that arise during the initial rollout.

In the review phase, the team collects feedback from users and evaluates the app's performance. They identify areas for improvement and prioritize new features based on user needs and suggestions. This continuous feedback loop allows the team to plan the next iteration of development, ensuring the app evolves to better meet the students' requirements and enhances their overall transportation experience.

## **2.4 Project Requirement**

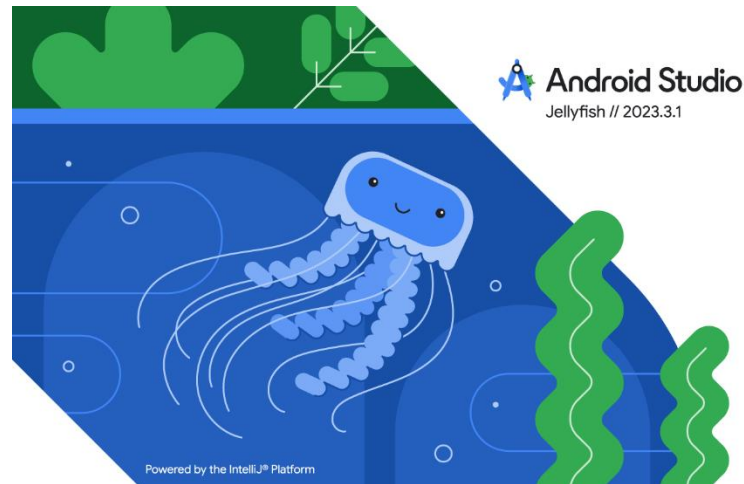
Project requirements are the features, functions, and tasks that need to be completed for a project to be successful. They establish various objectives for stakeholders to achieve and offer a clear set of guidelines for everyone involved to follow.

### **2.4.1 Software Requirements**

In developing this system, preparing the software requirements in the initial phase is crucial to ensure a smooth development process. For this project, the necessary software requirements are listed below:

#### **I. Android Studio IDE**



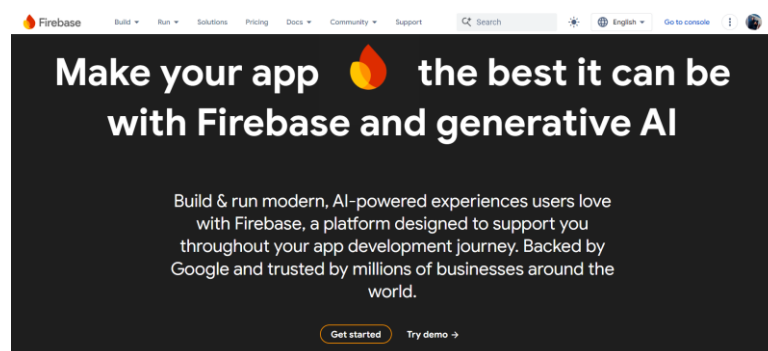


**Figure 2.5 Android Studio IDE**

Figure 2.4 shows Android Studio as the official IDE for Android app development, provided by Google. It offers a powerful code editor, visual layout editor, and various testing and debugging tools. Features like real-time code analysis, intelligent code completion, and an integrated emulator make it easy to develop and test high-quality Android apps efficiently.

اونيورسيتي تيكنيكل مليسيا ملاك  
UNIVERSITI TEKNIKAL MALAYSIA MELAKA

## II. Google Firebase



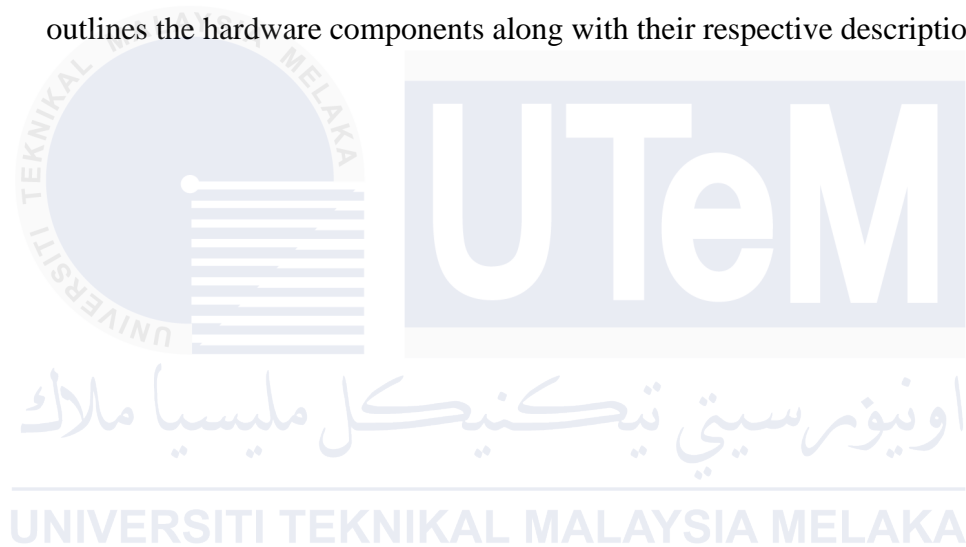
**Figure 2.6 Google Firebase**

Figure 2.5 shows Google Firebase is a platform that provides various tools and services to help developers build, improve, and grow their applications.

It offers backend services like real-time databases, authentication, cloud storage, and hosting. Firebase also includes analytics, performance monitoring, and crash reporting to help developers understand and improve app performance. With its comprehensive suite of features, Firebase simplifies the development process and enables developers to focus more on building great user experiences.

#### 2.4.2 Hardware Requirements

Each hardware component plays a specific role in supporting the research. Table 2.2 outlines the hardware components along with their respective descriptions.

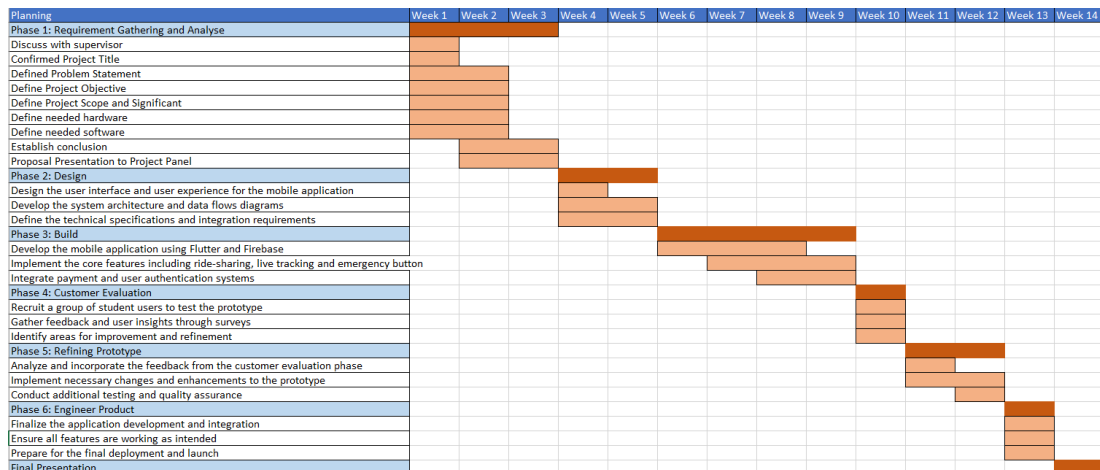


**Table 2.2 List of Hardware Requirements**

<b>Hardware</b>	<b>Specification</b>	<b>Description</b>
Android Phone	Android 11 and above	The program will be executed on an Android phone.
Laptop	Windows 10 Home Single Language or higher, with a minimum of 16GB RAM and an Intel Core i5 9th Gen or equivalent processor or better.	A laptop will be used for a variety of functions, such as running software required for program development and helping with project documentation.
USB Cable	USB 2.0 with supported data transfer.	Before the program can be launched, an Android phone and laptop must be connected via a USB cable.

## 2.5 Project Schedules and Milestones

A milestone is a specific point in a project's life cycle used to measure progress toward the final goal. In project management, milestones indicate key events such as the project's start and finish dates, external reviews and feedback, budget evaluations, and the submission of major deliverables. Figure 2.6 shows the project schedule and milestones for developing the Campus Ride system.



**Figure 2.7 Gantt Chart for Campus Ride Application**

## 2.6 Conclusion

This chapter offered a thorough examination of the literature pertinent to the development of the Campus Ride Application, emphasizing the urgent necessity for cost-effective and efficient student transportation solutions. It underscored the significance of comprehending the domain, assessing current systems, and implementing effective strategies to resolve identified challenges. Furthermore, the chapter delineated the project methodology, emphasizing that the Agile Development approach was the optimal choice for this project. The dynamic requirements of the student body are well-suited to the methodology's emphasis on adaptability, collaboration, and continuous improvement. Additionally, the development process was guaranteed a clear roadmap by specifying the requisite software and hardware requirements. Finally, the project schedule and milestones were established to assure the successful implementation of the Campus Ride Application and to measure progress. To effectively and efficiently address the transportation requirements of students, this structured approach guarantees that the project is well-positioned.

## CHAPTER 3: ANALYSIS

### 3.1 Introduction

The analysis phase in software development is the first step of the software development life cycle (SDLC) in which the requirements and goals of the software project are established and examined. During this phase, data will be collected and a comprehensive analysis will be conducted on the issue domain, user requirements, and current systems. This chapter presents an outline of the requirements gathered for both the current system (referred to as the "as-is system") and the proposed system, which is a Campus Ride Application. The requirements are described based on many factors, including business processes, data flow diagrams, functional requirements, and non-functional requirements.

### 3.2 Problem Analysis

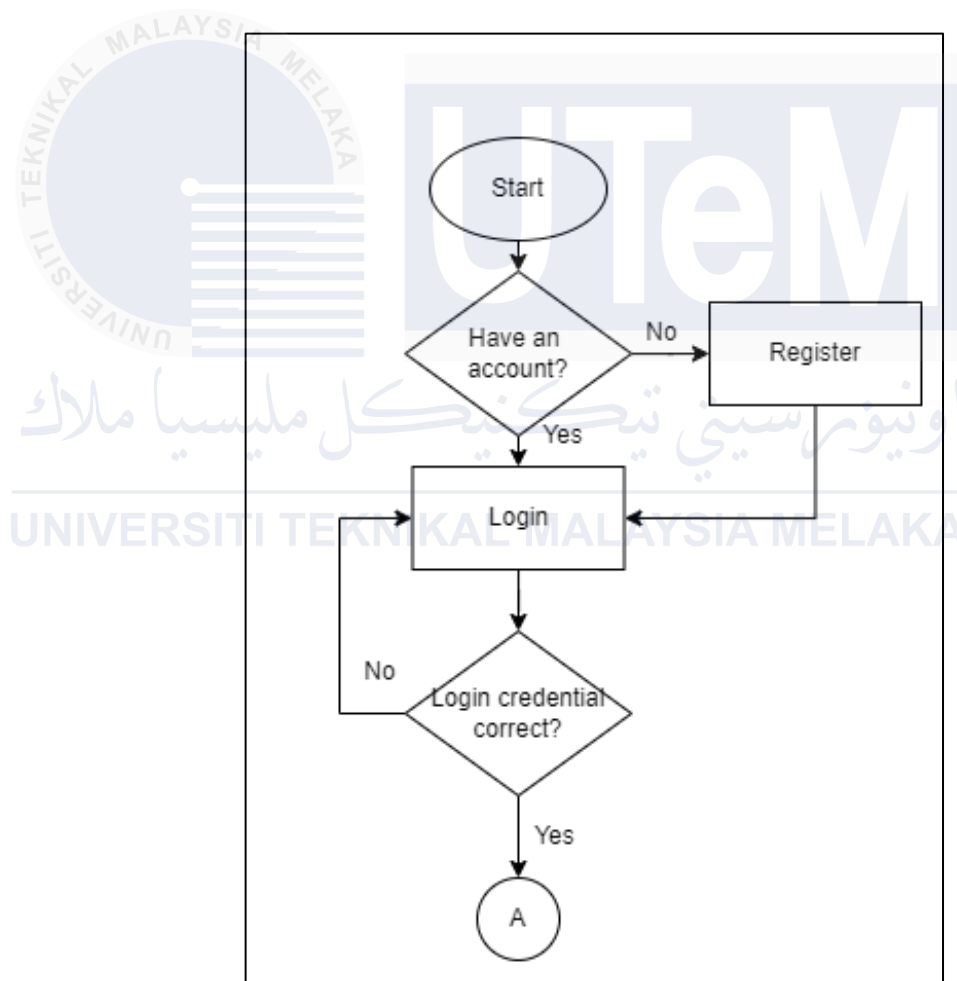
The chapter 1.2 covers the problem statement. This part presents the problem with the existing system and describes how the suggested system is to be implemented based on the issue with the existing system.

#### 3.2.1 Overview of Current System

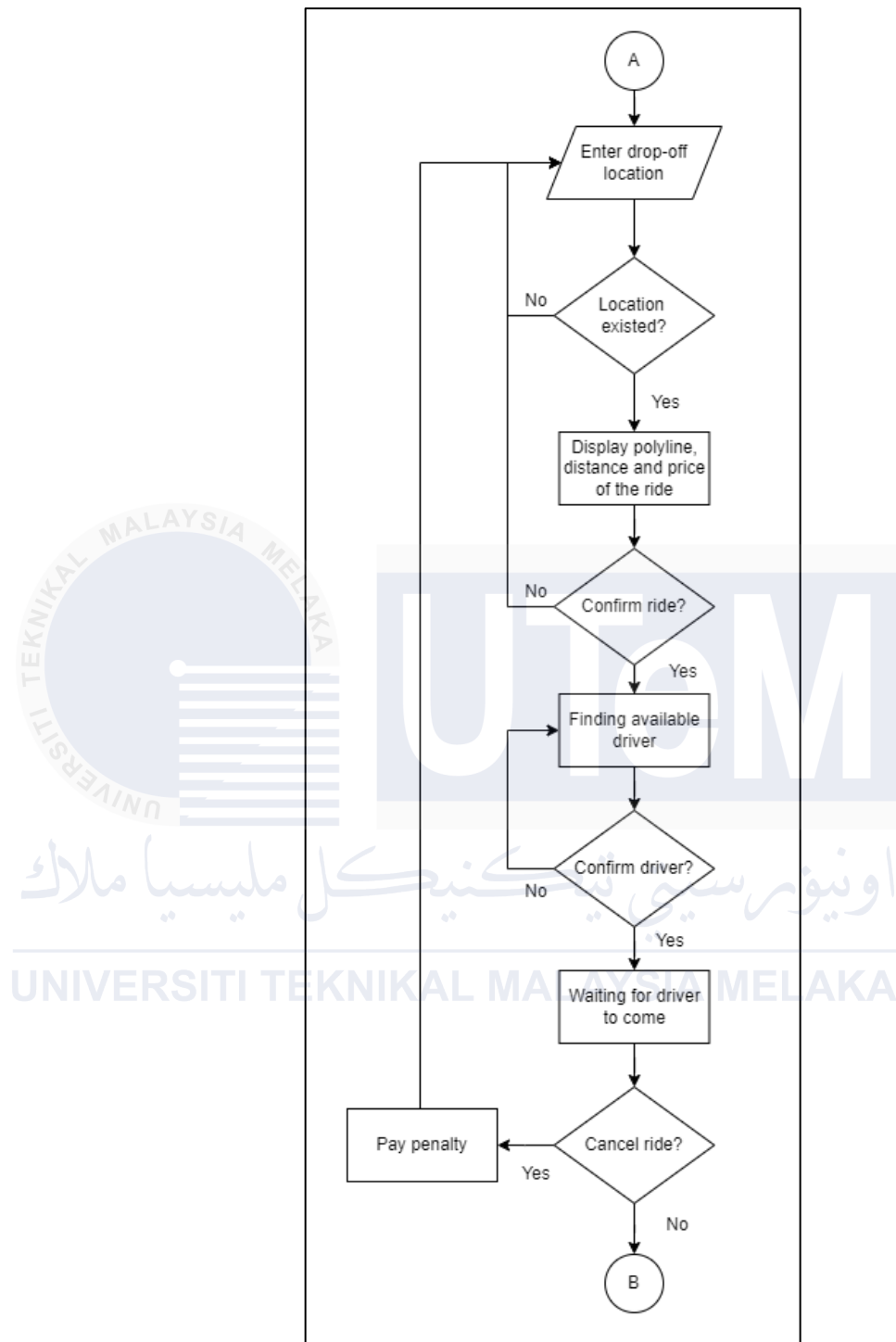
For students, the Grab system as it stands now poses problems with dependability, affordability, and convenience. Students with limited funds often cannot afford the high travel costs. Furthermore, Grab does not provide group ride options or special rates designed for students. The system currently offers features such as ride-hailing, food delivery, package delivery, and digital payments, which are all accessible through

the Grab mobile app. The system flow typically involves users opening the app, selecting the desired service, inputting their location and destination, and then confirming the booking or order. Despite these features, students are discouraged from utilizing Grab due to safety concerns, especially at night or in unfamiliar places. Enhancing security measures and offering incentives tailored to students can help address these problems and raise service confidence.

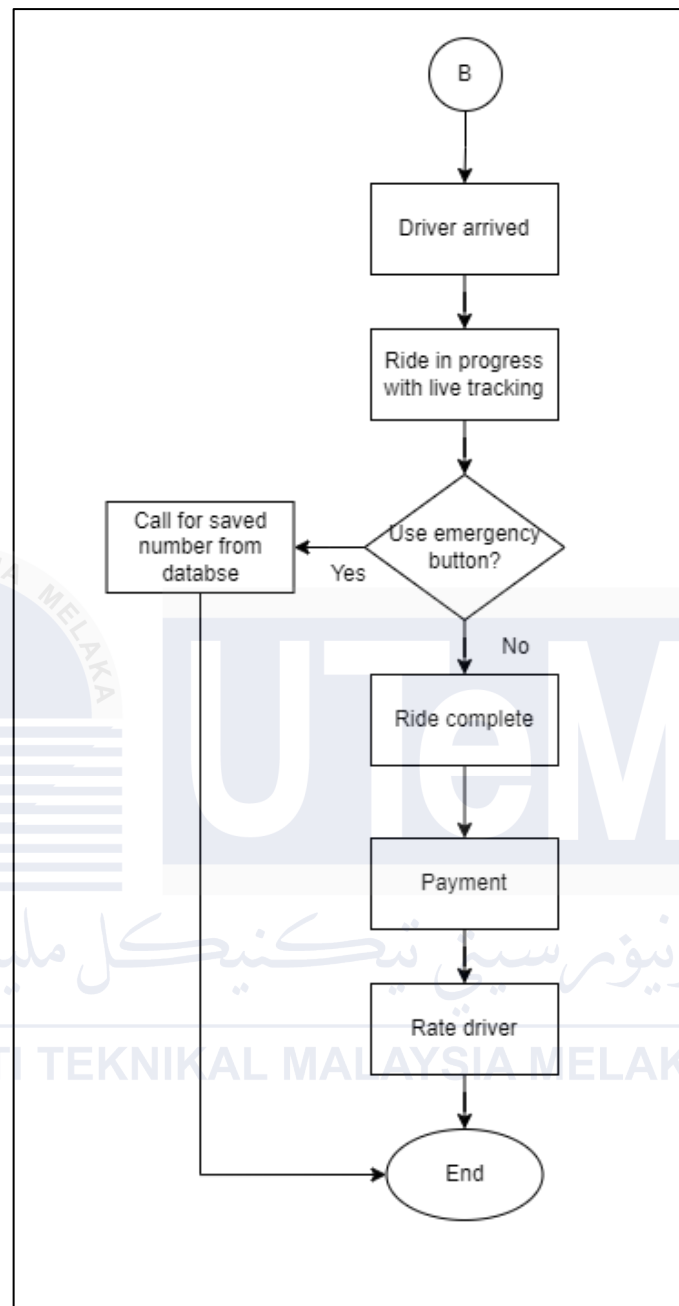
### 3.2.2 Overview of Proposed System



**Figure 3.1 Flowchart of the system (Part 1)**



**Figure 3.2 Flowchart of the system (Part 2)**



**Figure 3.3 Flowchart of the system (Part 3)**

Figure 3.1, 3.2 and 3.3 displays the suggested system flowchart for the Campus Ride Application. This flowchart illustrates the progression of the flow system. In this suggested system, users are required to log in before accessing the system. If they do not have an account, they must first register. Upon successful login, the user is required to input the drop-off location. If the specified location does not exist, the user must input another location that is closest to the desired drop-off location. If the system



exists, it will generate a polyline representing the course of the ride, together with the precise distance and price of the ride. Once the user confirms the ride, the system will initiate the process of locating the closest available driver. Users have the option to cancel their ride request if the price exceeds their budget. Once the driver is located, the system will provide a prompt to the user, asking if they wish to install this driver. When the user cancels, the system will propose an alternative driver. Once the driver has been confirmed, the user must wait patiently until the driver arrives. At this point, if the user decides to cancel the ride, they will be required to pay a penalty. Once the driver has arrived, the user can monitor the ride in real-time on the tracking website. This page is equipped with an emergency button to be used in case the user is in a perilous situation. The system will autonomously initiate a phone call to the individual designated by the user in their user profile. Upon completing the ride, the user is required to make payment to the driver using either an e-wallet or cash. Additionally, the user has the option to rate the driver. This ride will be recorded in the user's ride history.

### **3.3 Requirement Analysis**

Requirement analysis is the systematic procedure of comprehending, documenting, and scrutinizing the necessities, goals, and limitations of a software project or system. It entails the process of collecting, clarifying, and arranging the needs from different stakeholders, such as clients, users, and other pertinent parties. The objective of requirement analysis is to identify and specify the fundamental functionalities, features, and qualities that the program or system must have to achieve the desired goals. It aids in ascertaining the required functionalities and behaviours of the software or system, as well as the limits and limitations that should be considered during the development process. This section involves the comprehensive evaluation of all essential tasks associated with requirements, including the creation of a data dictionary, the establishment of functional requirements, and the specification of non-functional needs. Furthermore, it entails requesting further explanation from the client to obtain a precise comprehension of the comparative significance of various criteria.

### 3.3.1 Data Requirements

Data requirements pertain to the precise necessities and distinctive attributes of the data that a software system or application necessitates to operate efficiently. These requirements specify the necessary data components, their attributes, relationships, and limitations for the system to function. Data requirements are usually established during the requirement analysis stage of software development and are recorded to ensure a precise comprehension of the data necessary to support the system's functionality and objectives. The data dictionary will address the data requirements, namely which tables and attributes should be stored in the database.

#### 3.3.1.1 Data Dictionary

A data dictionary, usually referred to as a metadata repository, is a fundamental element of a database management system (DBMS). It functions as a thorough catalog or repository that offers intricate details on the data included within a database. For the Campus Ride Application, I'm utilizing Firebase as my DBMS. Below are the data definitions and descriptions for the application.

- Firebase Authentication

Table 3.1 shows the attributes that will be created in Firebase Authentication table which are email and password.

**Table 3.1 Firebase Authentication Dictionary**

Field Name	Data Format	Constraint	Description
Identifier	xxxx@xxxx	Not null	User email
Password	xxxxxxxxxx	Not null	User password

- Firebase Firestore User

Table 3.2 shows the attributes that will be created in User table which are name, shared\_Permission, emergencyNumber and fcmTokens.

**Table 3.2 User Data Dictionary**

User			
Field Name	Data Type	Data Format	Description
name	String	xxxxx	User fullname
shared_permission	Boolean	True/False	User share permission
emergencyNumber	String	xxxxx	User emergency number to call
fcmTokens	String	xxxxx	User's FCM Token to show notification

- Firebase Firestore Ride History

Table 3.3 shows the attributes that will be created in Ride History table which are date, from, to and status.

**Table 3.3 Ride History Data Dictionary**

Ride History			
Field Name	Data Type	Data Format	Description
date	Timestamps	Date and Time	User ride timestamp
from	String	xxxxx	User pickup location
to	String	xxxxx	User drop off location
status	String	xxxxx	User ride status

- Firebase Firestore Wallet

Table 3.4 shows the attributes that will be created in Wallet table which are date, from, to and status.

**Table 3.4 Wallet Data Dictionary**

<b>Ride History</b>			
<b>Field Name</b>	<b>Data Type</b>	<b>Data Format</b>	<b>Description</b>
totalBalance	Number	xxxx	User e-wallet balance
lastUpdated	Timestamps	Date and Time	User last balance reload
to	String	xxxxx	User drop off location

- Firebase Firestore Transaction History

Table 3.5 shows the attributes that will be created in Transaction History table which are amount, name, time and type.

**Table 3.5 Transaction History Data Dictionary**

<b>Ride History</b>			
<b>Field Name</b>	<b>Data Type</b>	<b>Data Format</b>	<b>Description</b>
amount	Number	xxxx	Amount enter or out from wallet
name	String	xxxx	Title for the transaction
time	Timestamps	Date and Time	User transaction made
type	String	xxxx	Value to determine if the amount is enter or out from wallet

- Firebase Firestore Driver

Table 3.6 shows the attributes that will be created in Transaction History table which is name.

**Table 3.6 Driver Data Dictionary**

<b>Ride History</b>			
<b>Field Name</b>	<b>Data Type</b>	<b>Data Format</b>	<b>Description</b>
name	String	xxxx	Fullname of the driver

### 3.3.2 Functional Requirements

Functional requirements are precise and detailed descriptions of the actions, tasks, or capabilities that a system, program, or product must have to effectively serve its intended purpose and satisfy the requirements of its users. These requirements delineate the operational behavior and capabilities of the system, specifying its expected actions or performance in different contexts. Table 3.7 displays the Functional Requirements of the Campus Ride Application.

**Table 3.7 Functional Requirements**

<b>FR No.</b>	<b>Module</b>	<b>Functional Requirement</b>
FR01	Login Module	Users need to enter Matric Number and Password before entering the system
FR02	Registration Module	Users need to enter Full name, Matric Number, Password and Confirm Password to register new account into system.
FR03	Home Page	System displays the location of user in the map.
FR04	Searching Module	Users need to enter Drop Off location to start using the service.
FR05	Live Tracking Module	Users can monitor current ride on the map.

FR06	Emergency Module	Users can click Emergency Button to quickly call the number that has been setup in profile.
FR07	Wallet Module	Users can track current balance in e-wallet and monitor transaction history made using wallet.

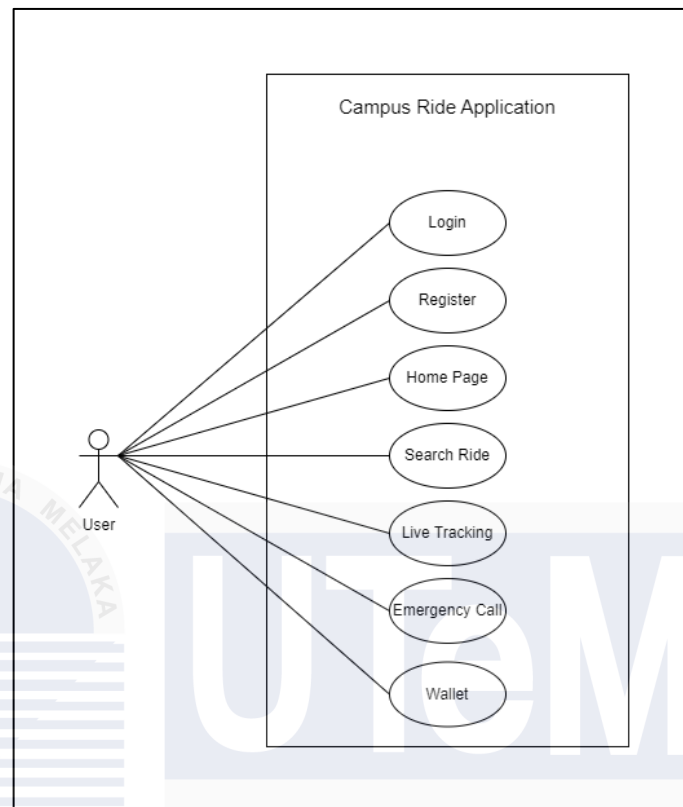
### 3.3.3 Non-Functional Requirements

Non-functional requirements, commonly referred to as quality attributes or system qualities, are the specific criteria that establish the overall behaviour, performance, and characteristics of a system, software, or product. Nonfunctional requirements differ from functional requirements in that they specify the desired behaviour and performance of the system in terms of other attributes, rather than specific capabilities. Table 3.8 displays the Non-Functional Requirements of the Campus Ride Application.

**Table 3.8 Non-Functional Requirements**

<b>NFR No.</b>	<b>Non-Functional Requirements</b>
NFR01	The system should provide search results and display ride options within 2 seconds.
NFR02	The application should be compatible with mobile operating systems (iOS, Android).
NFR03	The application should load within 3 seconds on a standard mobile internet connection.
NFR04	The system should handle errors gracefully, providing meaningful error messages and allowing users to retry actions where appropriate.

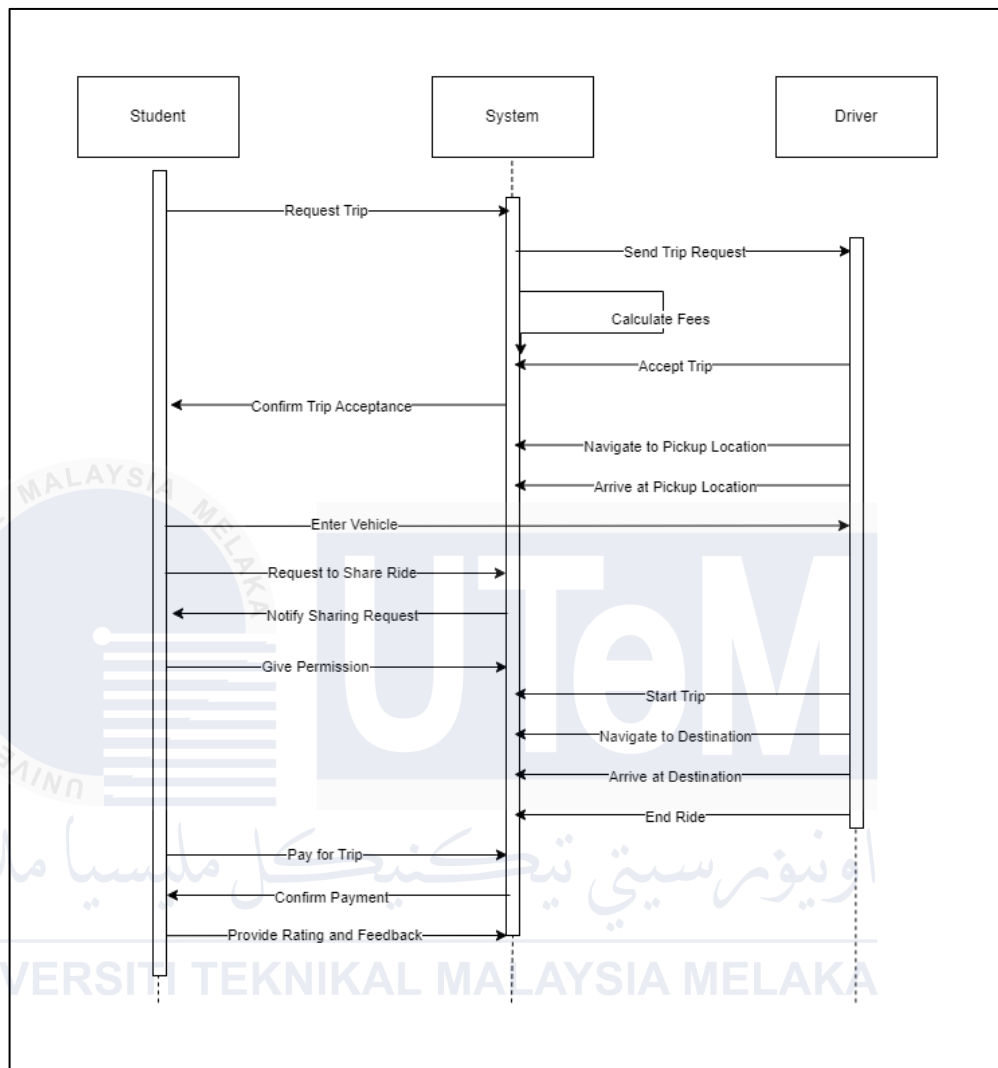
### 3.3.4 Use Case Diagram



**Figure 3.4 Use Case Diagram**

Figure 3.2 depicts the Use Case Diagram of the Campus Ride Application. A use case diagram is a graphical depiction of the functional specifications of a system, program, or product. It demonstrates the exchanges between actors (users or external systems) and the system, highlighting the different scenarios or activities carried out by the actors within the system.

### 3.3.5 Sequence Diagram



**Figure 3.5 Sequence Diagram**

Figure 3.2 depicts the sequence diagram of the Campus Ride Application. The user assumes the role of the actor in this diagram, which also demonstrates the flow of the system with its main module.

### 3.4 Conclusion

The problems with analysis, functional requirements, and non-functional needs will be covered in end of this chapter. This chapter aims to analyse the present system to help the developer improve the system more smoothly by pointing up and fixing possible problems.



## CHAPTER 4: DESIGN

### 4.1 Introduction

The design of a software system refers to the systematic process of creating a detailed plan or blueprint. Prior to commencing the actual implementation of the software application, it is essential to make decisions regarding its architecture, components, and overall structure. Software system design is to ensure that a system is capable of being expanded, easily controlled, meets the desired functional requirements, and performs efficiently.

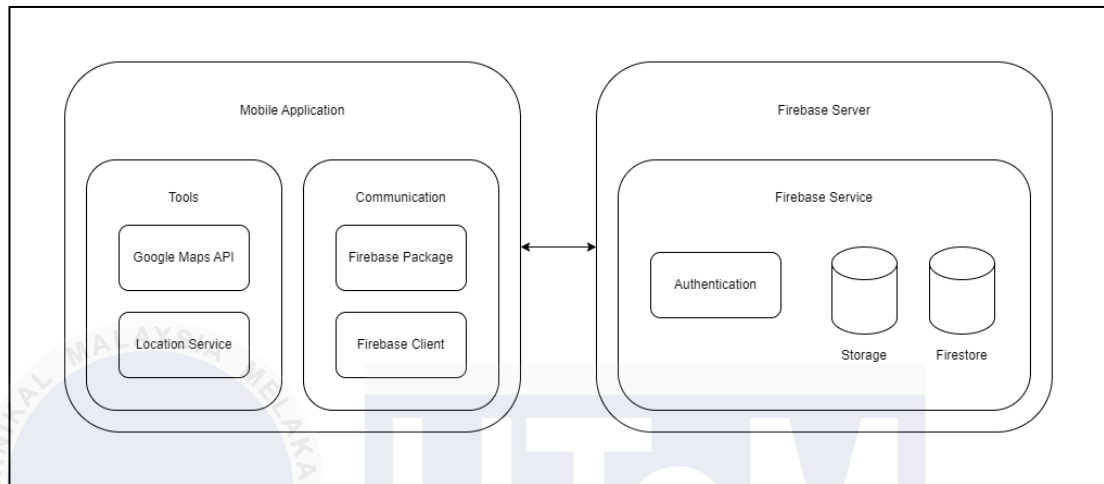
### 4.2 High Level Design

High-level design (HLD) refers to the process of creating a detailed and overarching plan for the structure and concept of a software system or application. The main emphasis is on the overall architecture, modules, and their interconnections, without delving into the technical implementation details. This section provides an overview of the system's high-level design, which includes the architecture, user interface design, and database design.

#### 4.2.1 System Architecture

System architecture refers to the conceptual structure and organization of a software system or application. The purpose of the document is to define the basic framework, elements, and their interconnections, serving as a blueprint for the design and development of the system. System architecture provides a comprehensive view of the system, highlighting its key components and their interrelationships. Users will

access and interact with the Campus Ride Application through mobile devices. User input data will be kept in the online database, specifically Google Firebase. The system's architecture is depicted below.



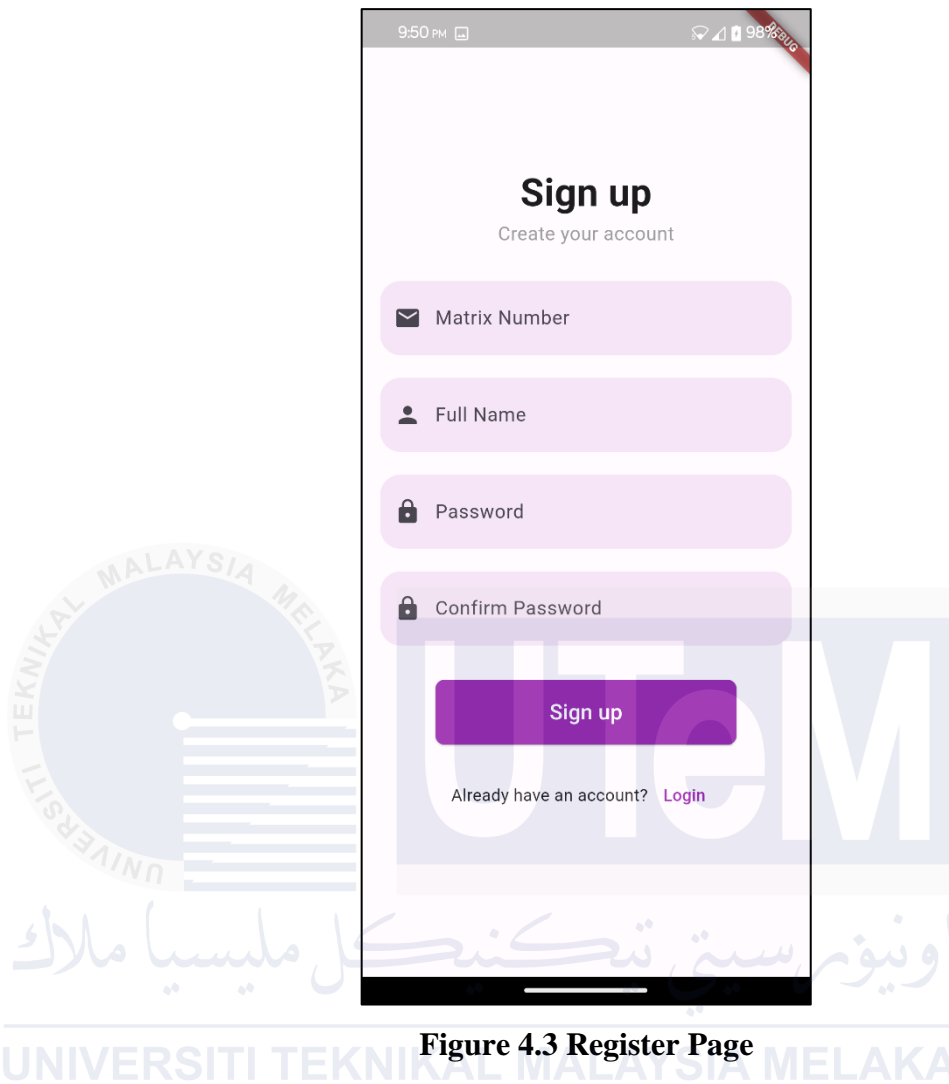
**Figure 4.1 System Architecture of Campus Ride Application**

Figure 4.1 shows the system architecture for Campus Ride Application which retrieve and fetch data from Firebase database.

#### 4.2.2 User Interface Design

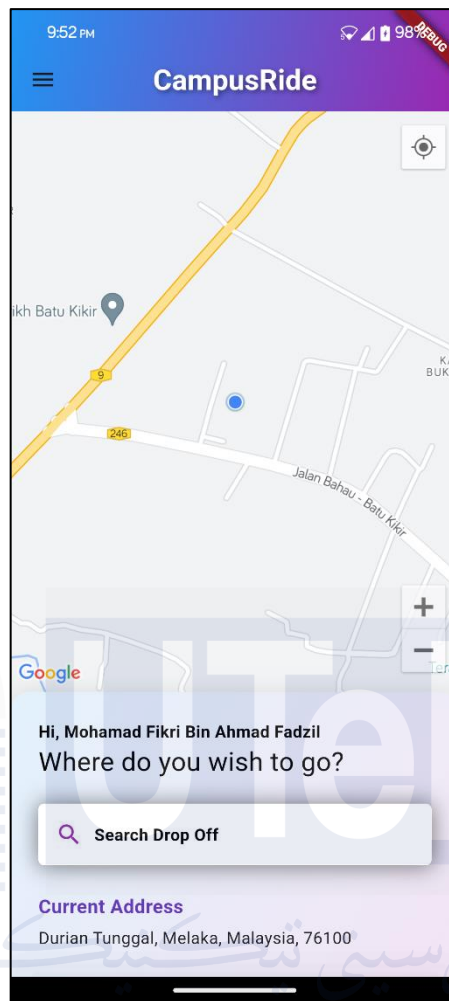
User interface design encompasses the activities involved in the creation and arrangement of the visual and interactive components of a system product, which facilitate user interaction. The main objective is to improve the user experience (UX) by creating a visually attractive, intuitive, and user-friendly interface. This section will provide an explanation of the UI design screenshots of the Blood Care Application.





**Figure 4.3 Register Page**

Figure 4.3 shows that on the Register Page, the user needs to enter their matrix number, full name, password, and confirm password to create a new account.



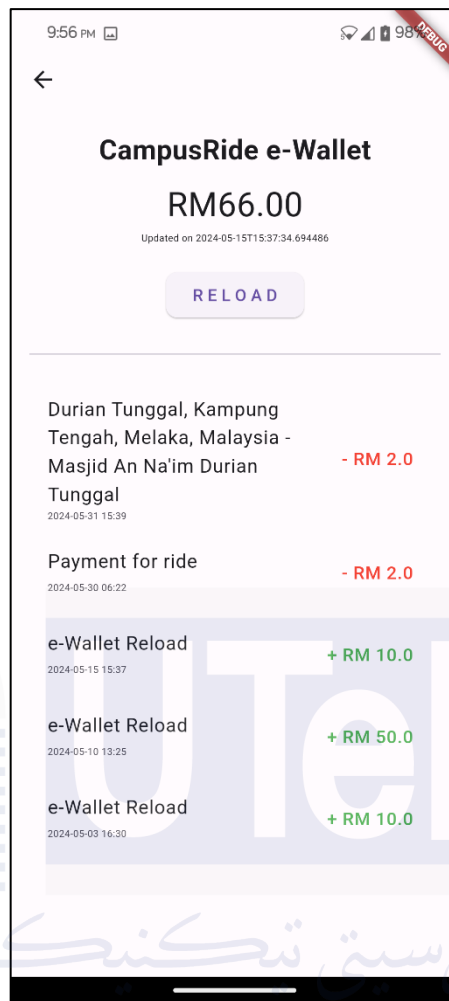
**Figure 4.4 Homepage**

Figure 4.4 show the homepage of the application to the user where user can click on the search drop off to start searching for the drop off location.



**Figure 4.5 Drawer**

Figure 4.5 shows the drawer of the application where the user can navigate to other pages within the application.



**Figure 4.6 Wallet Page**

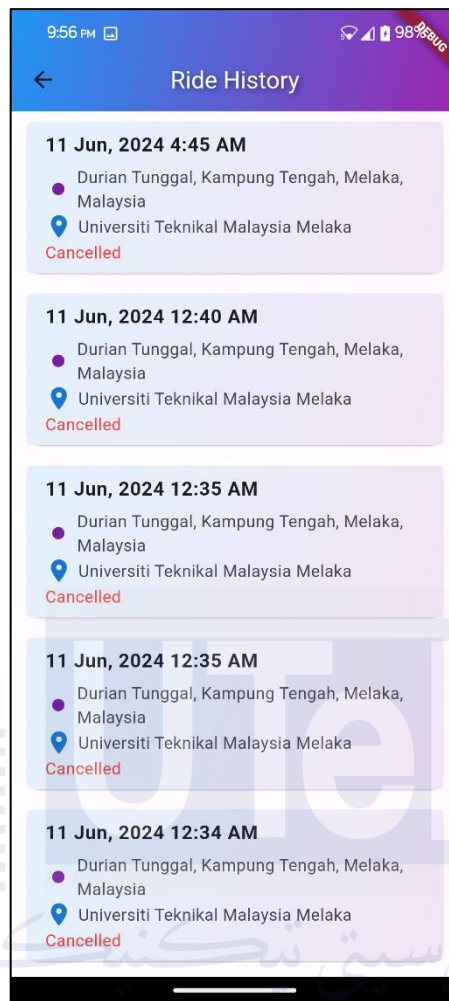
Figure 4.6 shows the e-wallet page where the user can track their balance and transaction history. Green indicates funds entered, while red indicates funds used.



**Figure 4.7 Setting Page**

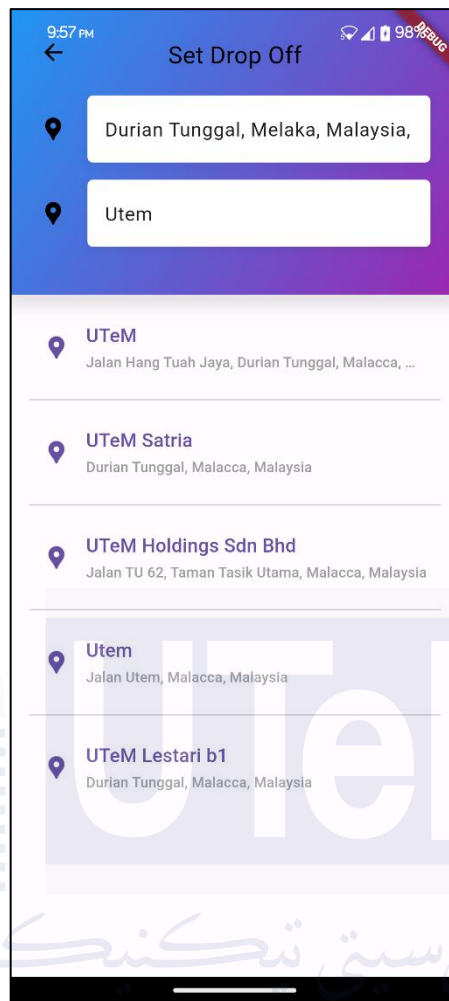
Figure 4.7 shows the settings page where the user can edit their profile, adjust privacy settings, manage notification preferences, change their password, and view the terms and conditions..





**Figure 4.8 History Page**

Figure 4.8 displays the history of ride requests. It includes details of all ride requests, even if they have been canceled.



**Figure 4.9 Searching Page**

Figure 4.9 shows the drop-off location search page. Here, the user can enter any place name, and the system will provide an autocomplete list based on the input. The user can then click on a suggestion to proceed with the ride booking.



**Figure 4.10 Request Page**

Figure 4.10 shows the request page. Here, the user can see the price and distance to the drop-off location, and enter the total number of passengers before requesting the ride.



**Figure 4.11 Finding Driver Page**

Figure 4.11 shows the system searching for an available driver.



**Figure 4.12 Driver Found Page**

Figure 4.12 shows that the system has found a driver. If the user does not want the assigned driver, they can cancel the ride and start a new request.



**Figure 4.13 Driver Arrived Confirmation**

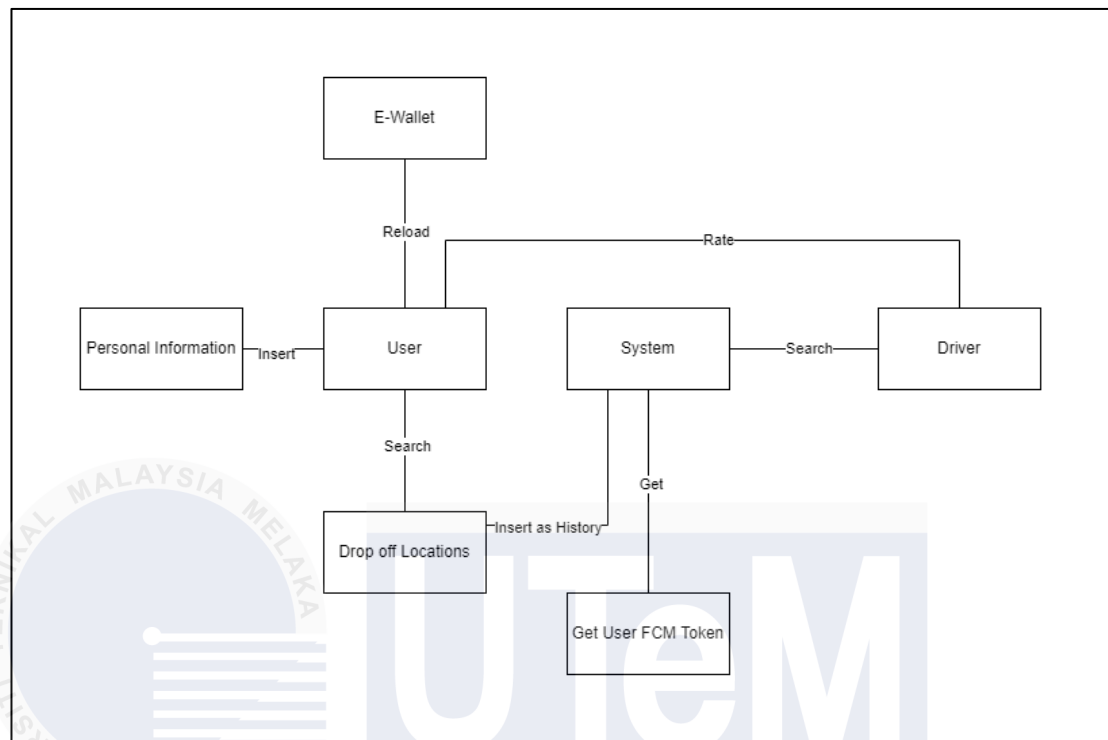
Figure 4.13 shows that the driver is on their way to the drop-off location. The user needs to click the "Driver Arrived" button to inform the system that the driver has arrived.



**Figure 4.14 Live Tracking Page**

Figure 4.14 shows the live tracking page where the user can monitor their location and ensure the correct path is being followed. The user can click the SOS button to immediately call the contact set up in their profile if they feel unsafe.

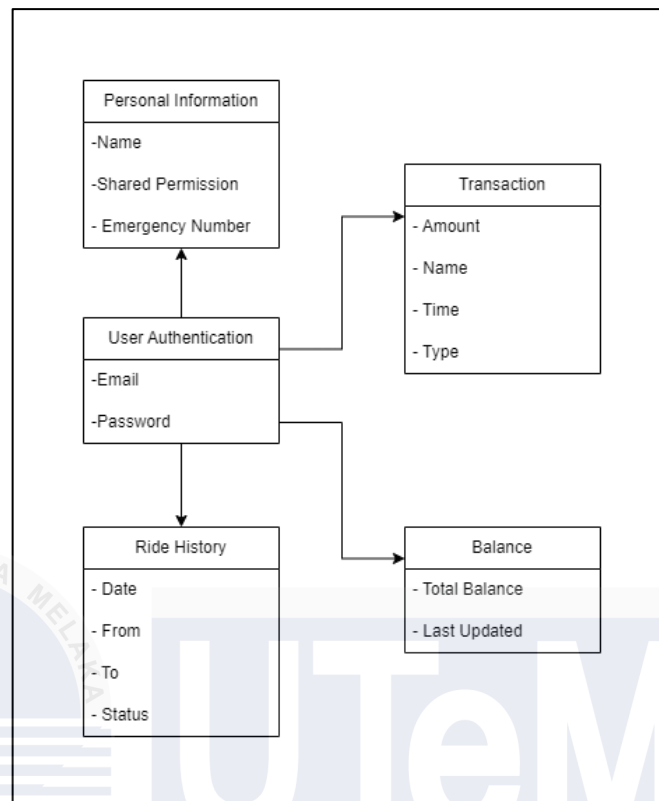
### 4.2.3 Conceptual and Logical Database Design



**Figure 4.15 Conceptual Database Design**

Figure 4.15 shows the conceptual database design for Campus Ride Application. The design consists of seven entities which are User, Balance, Personal Information, Drop Off Location, System, Get User FCM Token, and Driver. User can reload the balance, insert the personal information, search the drop off location and rate the driver. The system will save the drop off location into the ride history and system will automatically save the user FCM token into database.





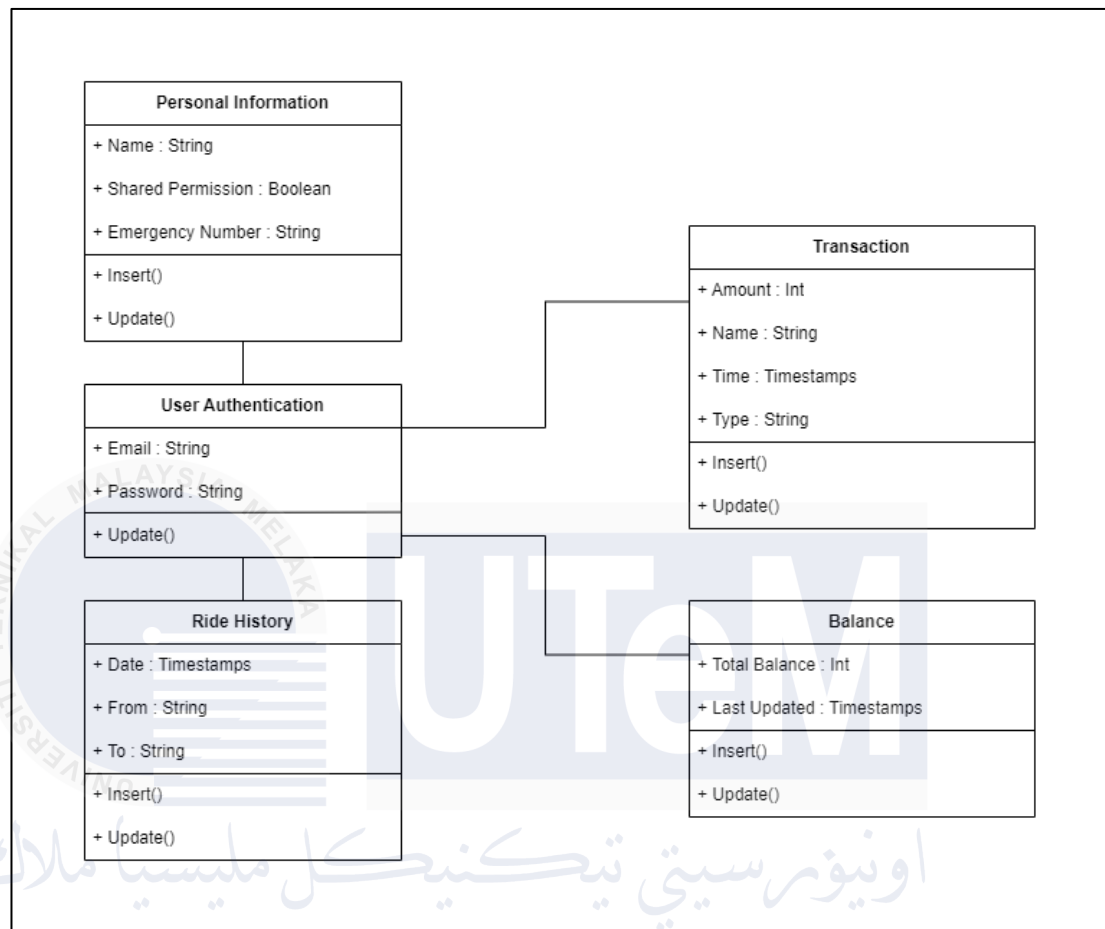
**Figure 4.16 Logical Database Design**

Figure 4.16 depicts the logical database design of Campus Ride Application that consists of five entities and every entity has their own attributes in the table.

### 4.3 Conceptual and Logical Database Design

Detailed design is the stage in which the overall structure and specifications of a software system are transformed into a more intricate and precise design. It entails developing the technical blueprint or strategy for executing the software solution. This section will provide a comprehensive explanation of the intricate design of the Blood Care Application.

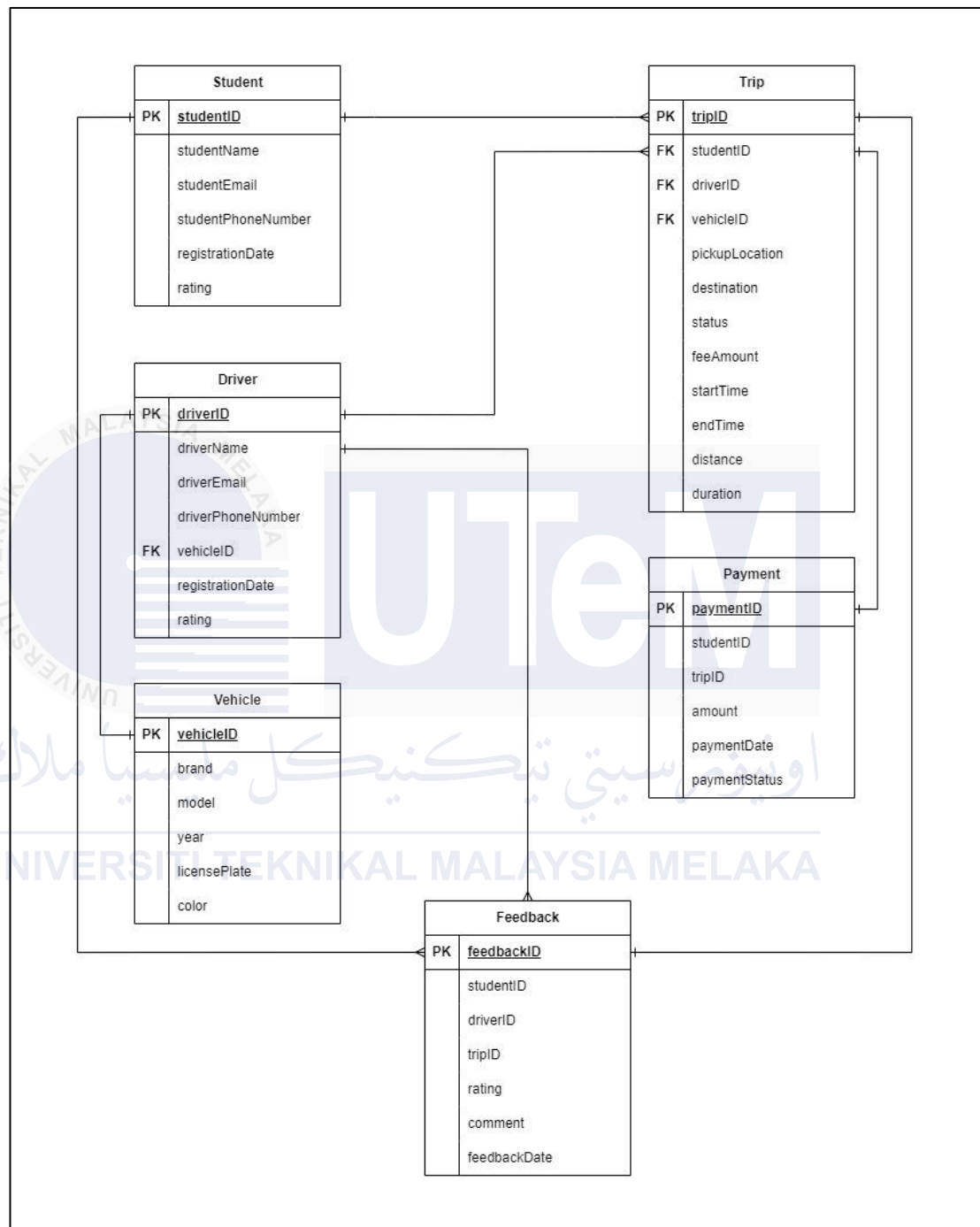
### 4.3.1 Software Design



**Figure 4.17 Software Design**

Figure 4.17 shows the class diagram for Blood Care Application which displays seven tables which show all the attributes contained for each of the tables.

### 4.3.2 Physical Design



**Figure 4.18 Entity Relationship Diagram**

Figure 4.18 displays the Entity Relationship Diagram (ERD) for the Campus Ride Application. This Entity-link Diagram (ERD) illustrates the specific link between each table and the attributes associated with each table.

#### 4.4 Conclusion

In this chapter, the design of the project was delineated, encompassing an Entity Relationship Diagram for both the Conceptual and Logical Design. The system's architecture, Graphic User Interface (GUI), and Database Design were all discussed in a comprehensive manner. The output obtained can be directly employed for the development of programming languages.

Before starting system development, it is essential to complete the design process. This step facilitates the developer in determining the most suitable techniques and tactics to utilize prior to moving forward with the implementation phase. Software design can be accomplished using several methodologies. An effective design can result in a successful development phase.

## CHAPTER 5: IMPLEMENTATION

### 5.1 Introduction

All the planning and designing will begin to manifest as the actual product during system installation. The team members will focus on creating, testing, debugging, and installing compilers all tasks necessary to construct the system. Ensuring that the finished product functions as intended during the design phase is the aim of implementation.

### 5.2 Software Development Environment Setup

#### 5.2.1 Android Studio

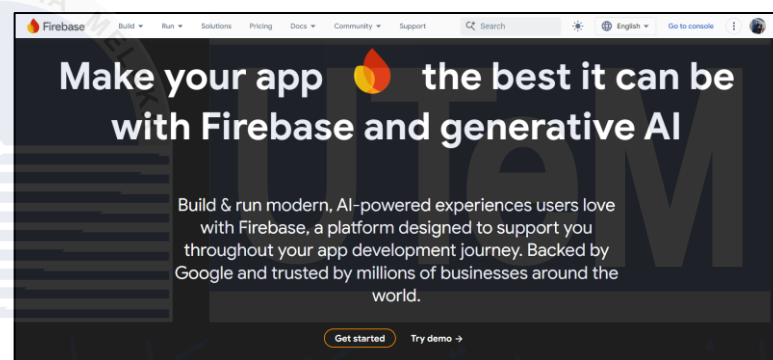


**Figure 5.1 Android Studio IDE**

Android Studio is the primary software development environment utilized in the creation of the Campus Ride Application, as seen in Figure 5.1. An integrated development environment (IDE) designed especially for creating Android apps is

called Android Studio. It is the official platform for creating Android apps, and developers use it extensively for development, testing, and debugging. A comprehensive suite of tools and capabilities is provided by Android Studio to facilitate the development process and assist developers in producing high-calibre Android apps. It is continually being updated and enhanced, which makes it a useful resource for Android app developers. It makes the process of creating apps simpler and offers a wealth of information and documentation to support developers of all experience levels.

### 5.2.2 Google Firebase



**Figure 5.2 Google Firebase**

Figure 5.2 illustrates the tools and services that Google Firebase, a cloud-based platform, offers for developing mobile and web applications. It offers a NoSQL cloud-based database called Firebase Realtime Database, but it's not just a database administration tool like phpMyAdmin. Applications that require real-time changes, such as chat apps or collaboration tools, are best suited for this database. Firebase Realtime Database is scalable and adaptable since it saves data as JSON objects, in contrast to conventional relational databases. Other database choices provided by Firebase include Cloud Firestore, a document-oriented database with powerful query capabilities and offline support. Developers creating contemporary apps frequently use Firebase because of its interoperability with other services like cloud storage, hosting, and authentication.

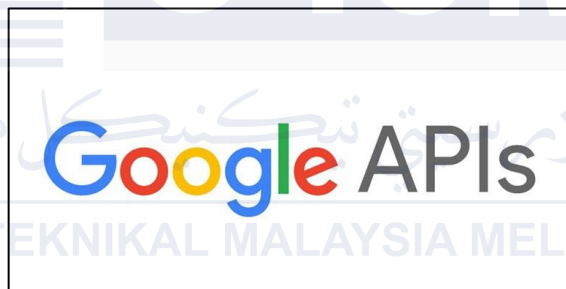
### 5.2.3 Programming Language



**Figure 5.3 Flutter Programming Language**

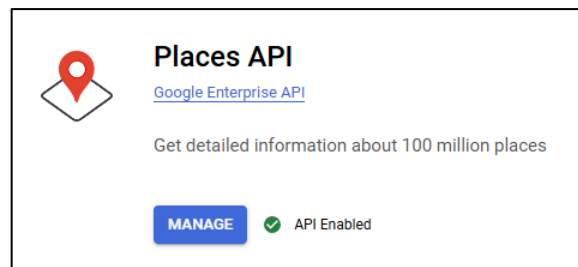
Figure 5.3 displays the programming languages that are employed in this system. Flutter, a cross-platform UI software development kit renowned for its quick development and expressive UI capabilities, is mostly used in the program's development. Flutter is built on top of Dart, a contemporary object-oriented language that makes app development quick and easy.

### 5.2.4 Google APIs



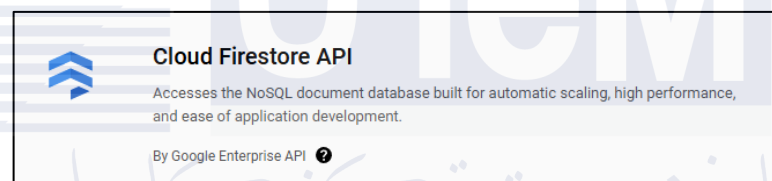
**Figure 5.4 Google APIs**

Figure 5.4 shows the APIs used to complete this project, which are Google APIs. Google APIs were chosen because they offer several advantages. They are easy to integrate into the project and there are many tutorials online to help with the process. If any issues arise, there is also support and a community available for assistance.



**Figure 5.5 Places API**

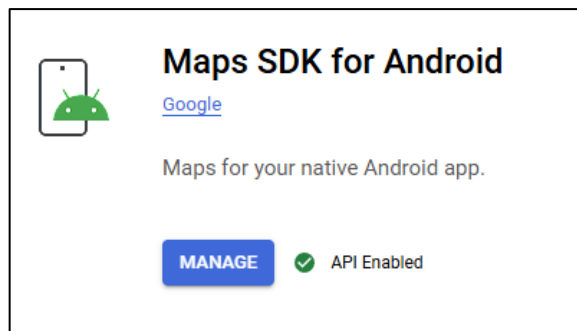
Figure 5.5 showing the Google Places API is a tool that helps find information about places around the world. It can be used to search for places like restaurants or shops, get details about them, and even see photos and reviews. This API is useful for apps that need to show nearby places or help users choose a location.



**Figure 5.6 Cloud Firestore API**

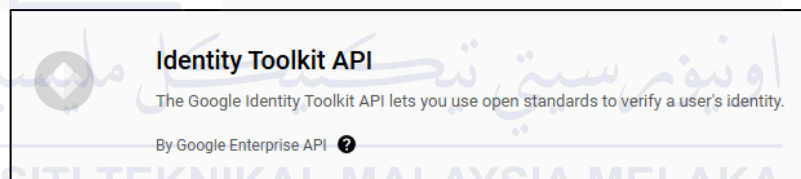
Figure 5.6 showing the Cloud Firestore API is a service that helps store and manage data in the cloud. It allows developers to save and retrieve data in real-time, making it easy to build apps that need to update information quickly. The API is especially useful for apps that require real-time syncing of data across multiple users and devices.





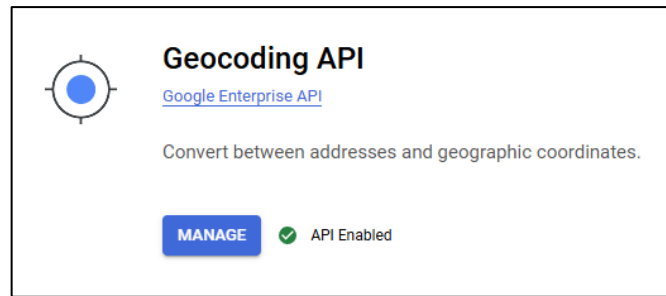
**Figure 5.7 Maps SDK for Android**

Figure 5.7 showing the Maps SDK for Android is a tool that lets developers add interactive maps to their Android apps. It allows users to view maps, explore locations, and get directions. The SDK also supports features like zooming, panning, and displaying markers to highlight specific points on the map, making it useful for apps that need location-based services.



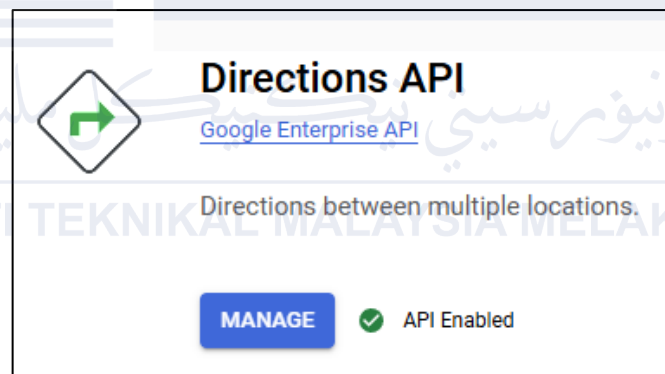
**Figure 5.8 Identity Toolkit API**

Figure 5.8 showing the Identity Toolkit API is a service that helps manage user authentication in apps. It allows developers to easily add sign-up, sign-in, and password management features. This API supports different authentication methods, like email and password, social media logins, and more, making it simple and secure for users to access the app.



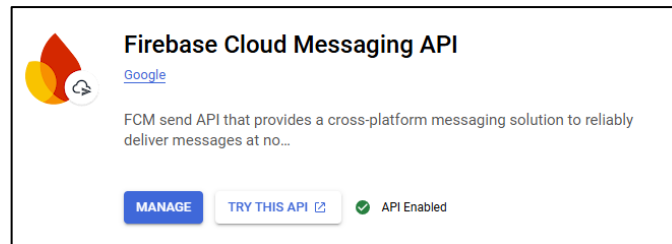
**Figure 5.9 Geocoding API**

Figure 5.9 showing the Geocoding API converts addresses into geographic coordinates (latitude and longitude) and vice versa. This means you can use it to find the exact location of an address or get an address from a set of coordinates. It's useful for applications that need to display locations on a map or perform location-based searches.



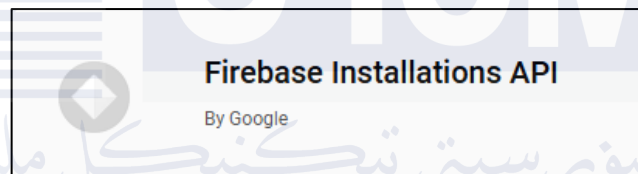
**Figure 5.10 Directions API**

Figure 5.10 showing the Directions API provides routes and directions between locations. It helps calculate the best path from one place to another, including turn-by-turn navigation. This API can also consider traffic conditions and offer alternative routes, making it useful for apps that need to guide users or plan travel routes.



**Figure 5.11 Firebase Cloud Messaging API**

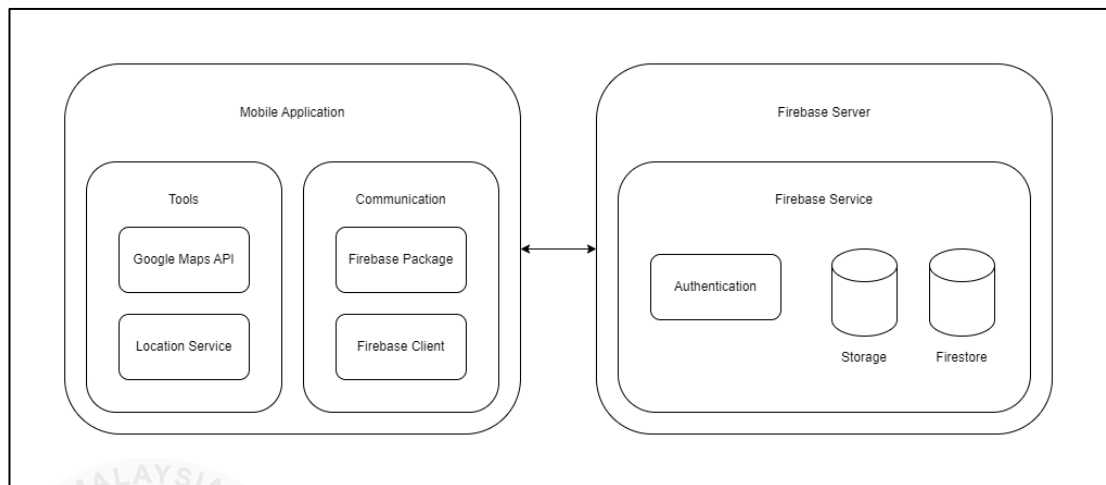
Figure 5.11 showing the Firebase Cloud Messaging (FCM) API allows developers to send notifications and messages to users' devices. It supports sending messages to single devices, groups of devices, or topics, and can include various types of content, like text or multimedia. This API is useful for keeping users informed and engaged with real-time updates and notifications.



**Figure 5.12 Firebase Installations API**

Figure 5.12 showing the Firebase Installation API helps manage unique identifiers for your app's installations. It generates and handles installation IDs that are used to track and manage app instances. This is useful for services like Firebase Cloud Messaging, which need to identify devices or app instances for sending targeted notifications or data.

### 5.2.5 Environment Architecture

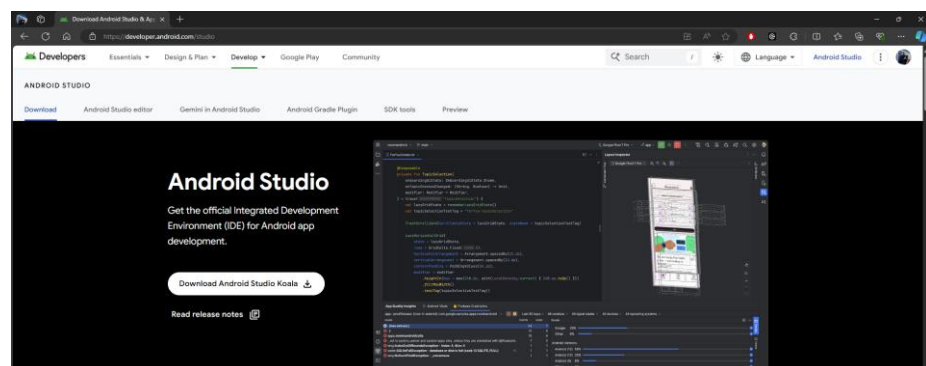


**Figure 5.13 Environment Architecture**

The system's environmental architecture is made up of two main parts: the mobile application and the Firebase server. The mobile application has two layers: tools and communication. The tools layer uses Google Maps API for location tracking and a location service to get the device's location. The communication layer includes a Firebase package and client, which help the app connect with Firebase services. On the server side, Firebase provides services like authentication, storage, and Firestore, a database that stores and syncs data in real-time. Together, these parts allow the app to work smoothly and securely.

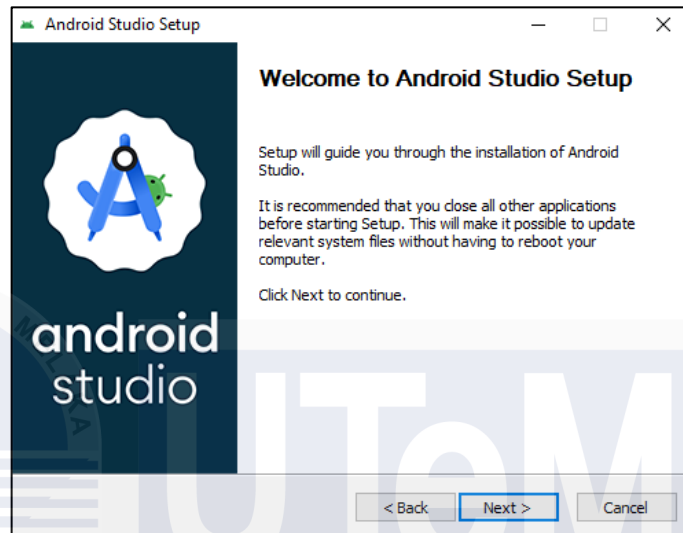
## 5.3 Software Configuration Management

### 5.3.1 Installation and Setup of Android Studio



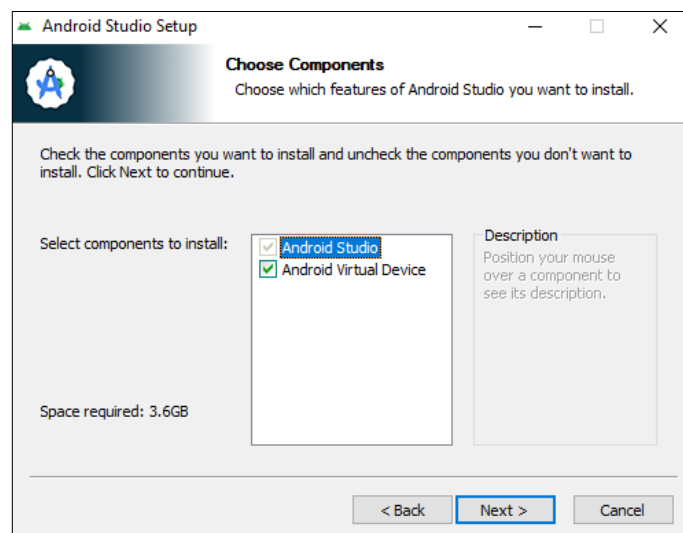
**Figure 5.14 Android Studio Download Page**

- Download Android Studio installer from the link: [Download Android Studio & App Tools - Android Developers](#)
- Run the installer file.

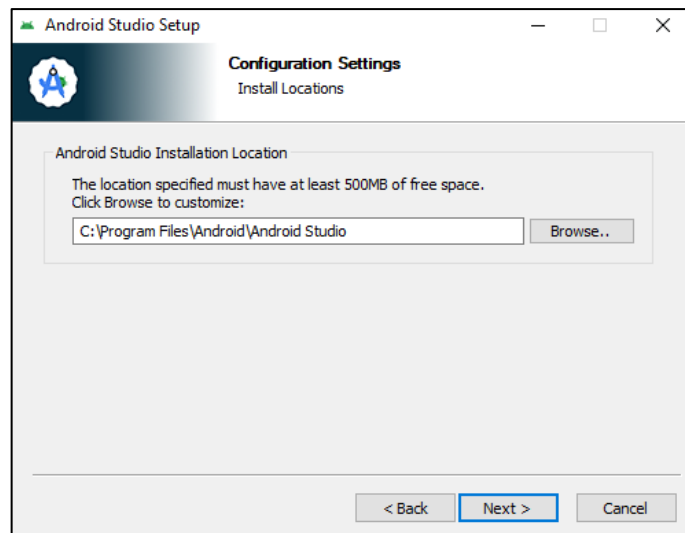


**Figure 5.15 Android Studio Setup I**

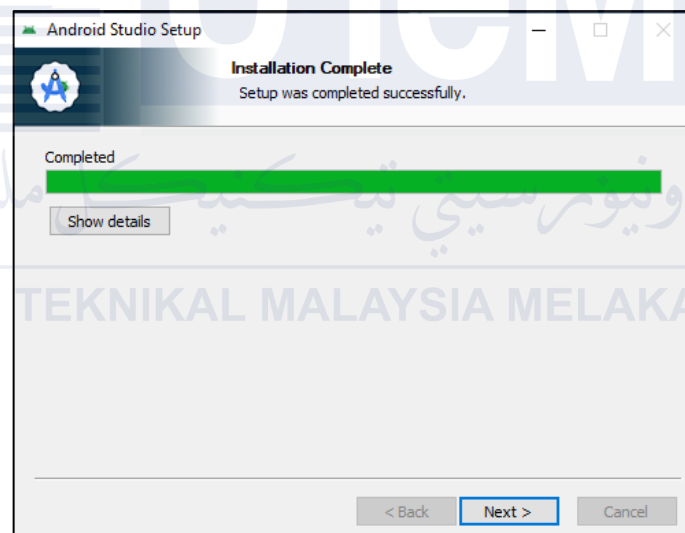
- Click “Next” button. Follow the default configuration given until the installation is finished.



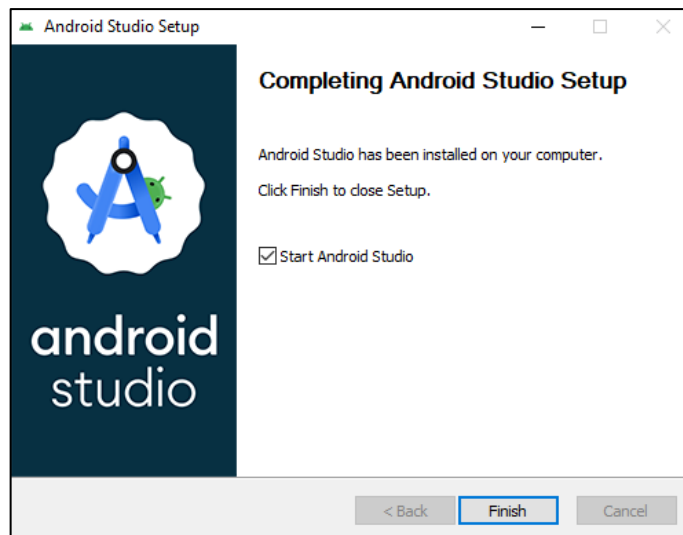
**Figure 5.16 Android Studio Setup II**



**Figure 5.17 Android Studio Setup III**

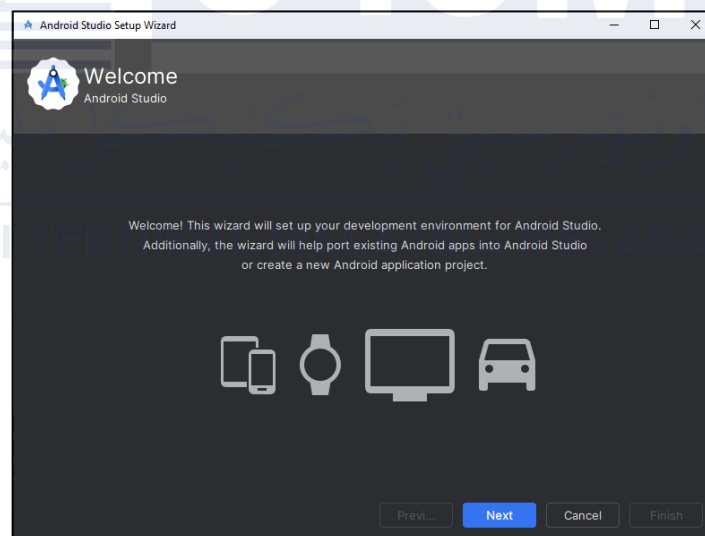


**Figure 5.18 Android Studio Setup IV**



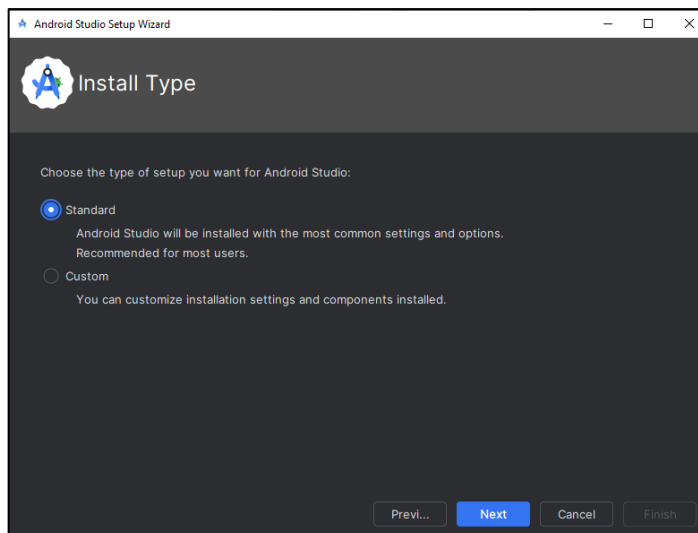
**Figure 5.19 Android Studio Setup V**

- The installation is done and click “Finish” to open Android Studio.

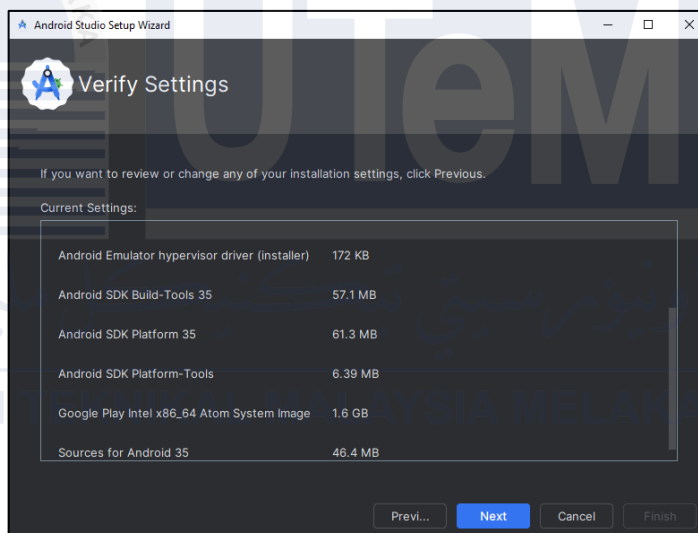


**Figure 5.20 Android Studio Setup VI**

- Upon opening the Android Studio, it has another setup wizard to be done. Click “Next” to continue. Follow the default configuration till the setup is done.

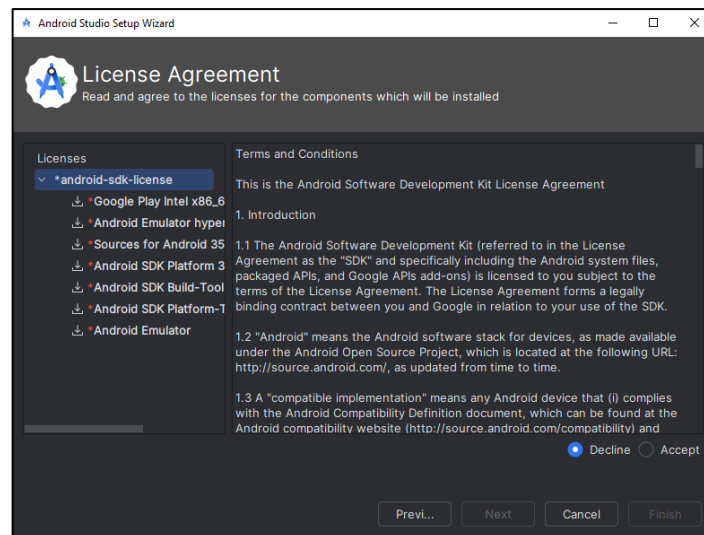


**Figure 5.21 Android Studio Setup VII**



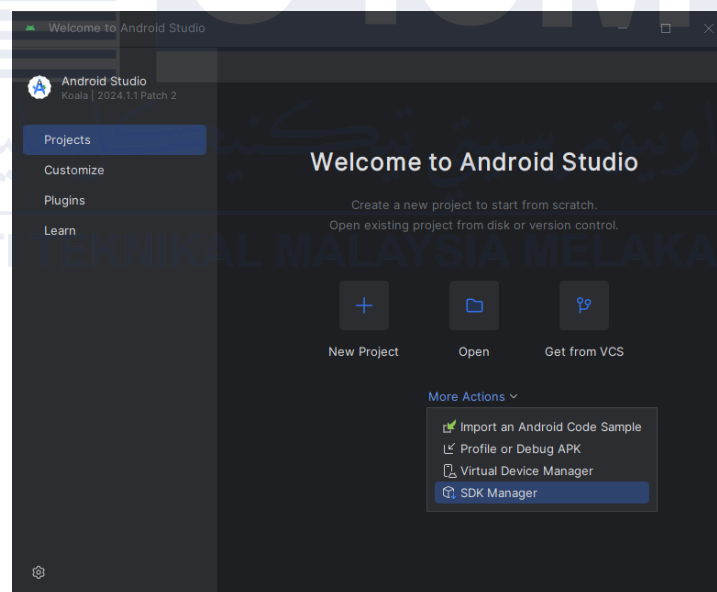
**Figure 5.22 Android Studio Setup VIII**





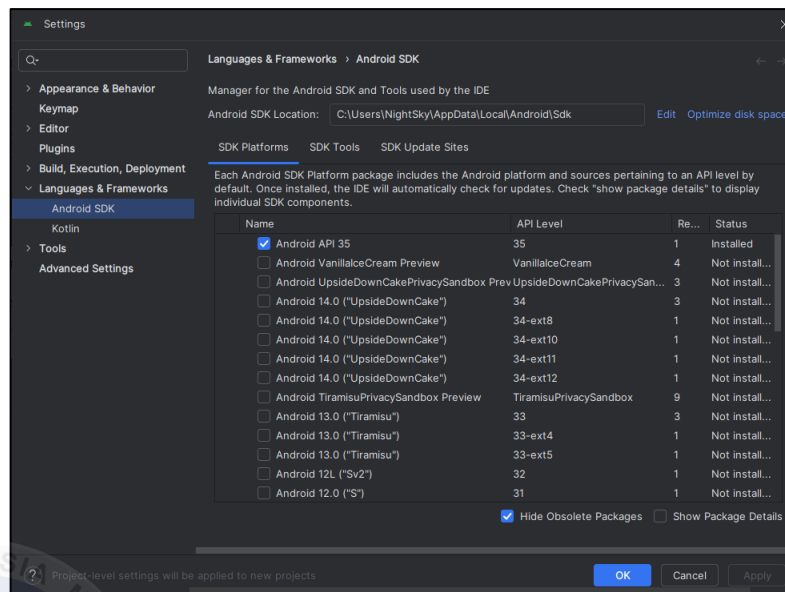
**Figure 5.23 Android Studio Setup IX**

- Accept the Terms and Conditions and click Finish. Android Studio will start to download its component. Stay still until the installation is complete.



**Figure 5.24 Android Studio Setup X**

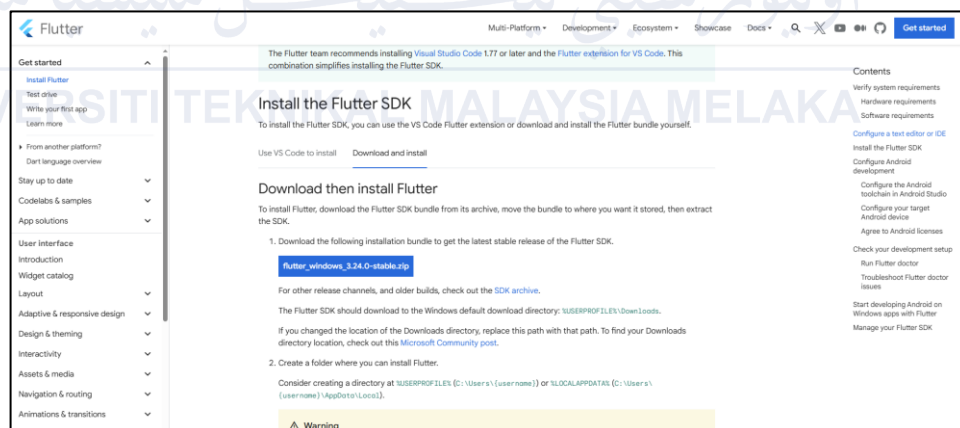
- Click “More Actions” and select SDK Manager.



**Figure 5.25 Android Studio Setup X1**

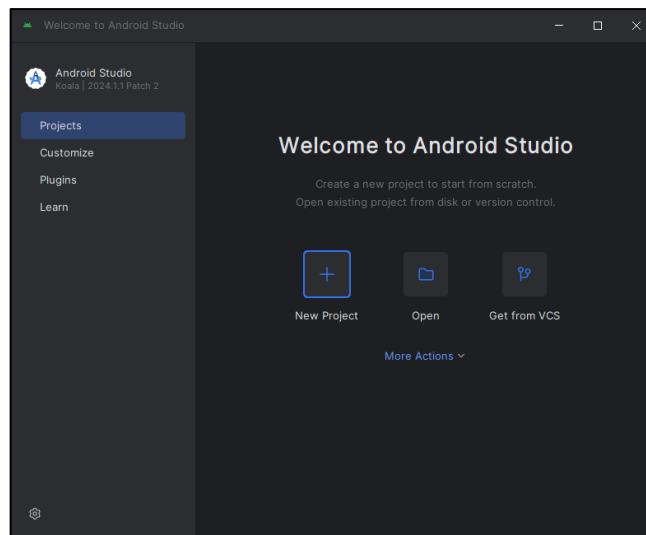
- In SDK Platform tab, ensure it use the latest Android API. Click “OK” and the installation is done.

### 5.3.2 Flutter Setup



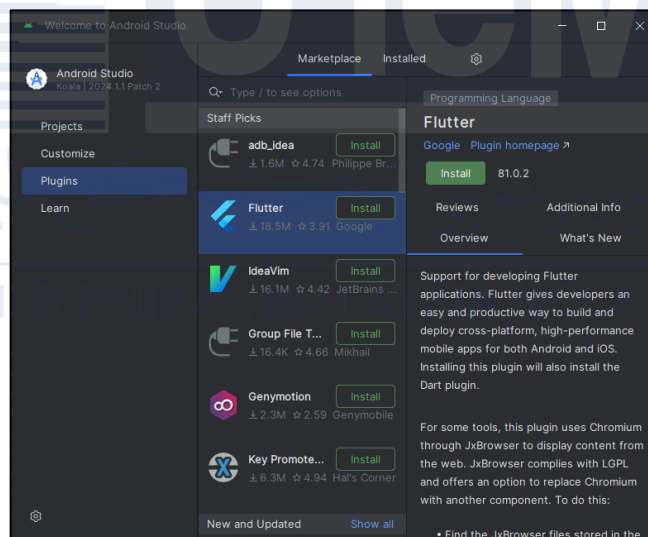
**Figure 5.26 Flutter Setup I**

- Go to this link [Make Android apps | Flutter](#) and download Flutter zip file.



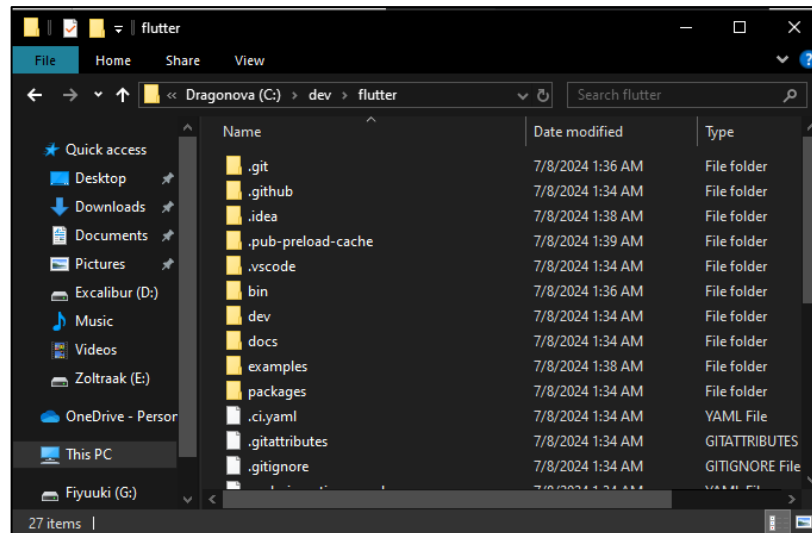
**Figure 5.27 Flutter Setup II**

- On this Android Studio start page, click Plugins.



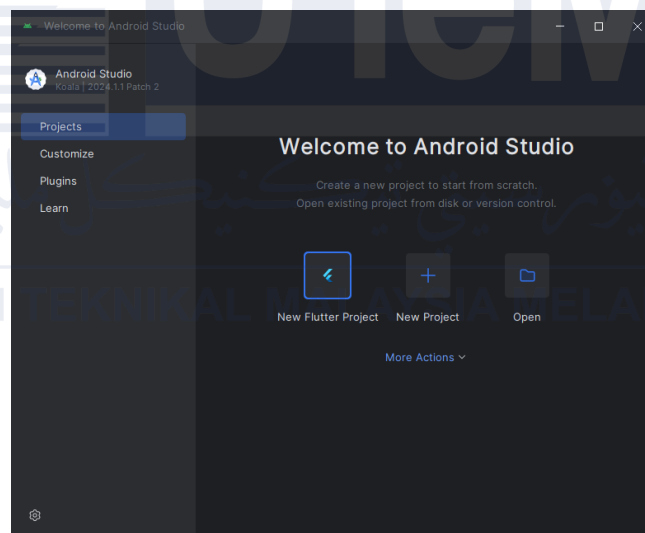
**Figure 5.28 Flutter Setup III**

- Install Flutter plugin. Accept if any prompt occurrence. Wait until Android Studio asks to restart itself.



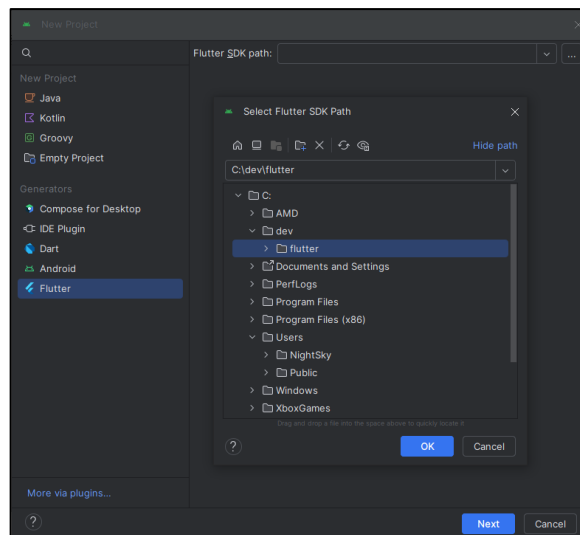
**Figure 5.29 Flutter Setup IV**

- Extract zip files downloaded from Flutter page and put it in C: drive (optional).



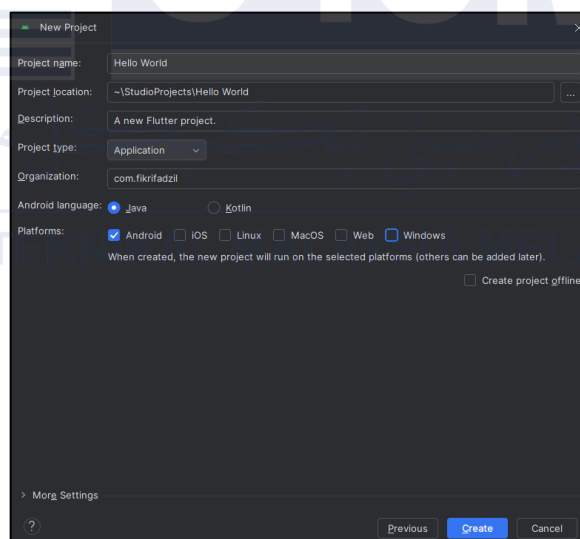
**Figure 5.30 Flutter Setup V**

- Go to Android Studio and click New Flutter Project.



**Figure 5.31 Flutter Setup VI**

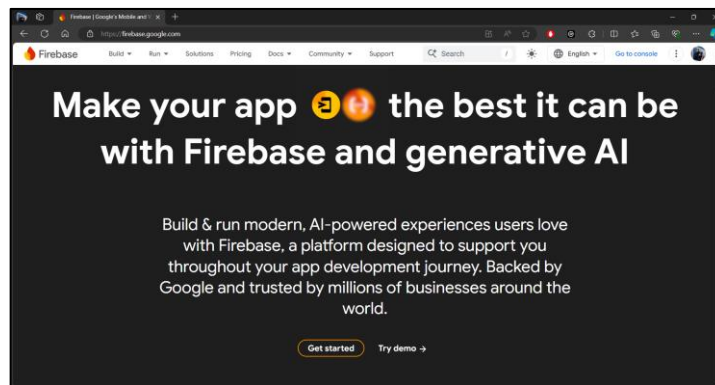
- Make sure on the left side you click Flutter Generators. Click on the three dots and select the files extracted before. Click “OK” and “Next”.



**Figure 5.32 Flutter Setup VII**

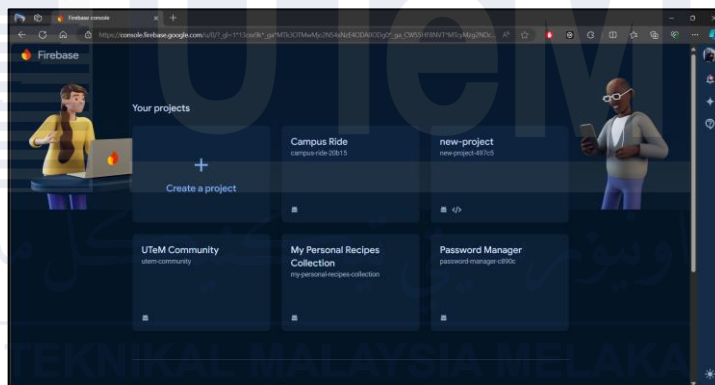
- Type the project name according to your choices. To construct an Android app, select Java as the Android Language and check Android only. Check iOS together if you also wish to build for iOS. Click “Create” and wait till it is finished.

### 5.3.3 Google Firebase Database Setup



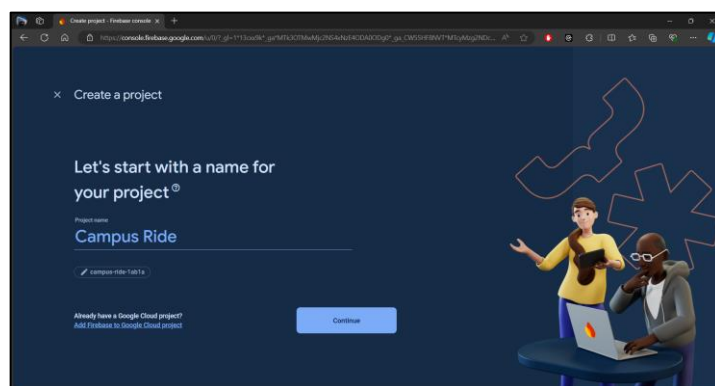
**Figure 5.33 Google Firebase Database Setup I**

- Go to the Google Firebase Home Page at [Firebase | Google's Mobile and Web App Development Platform](https://firebase.google.com) and click on “Get Started” button.



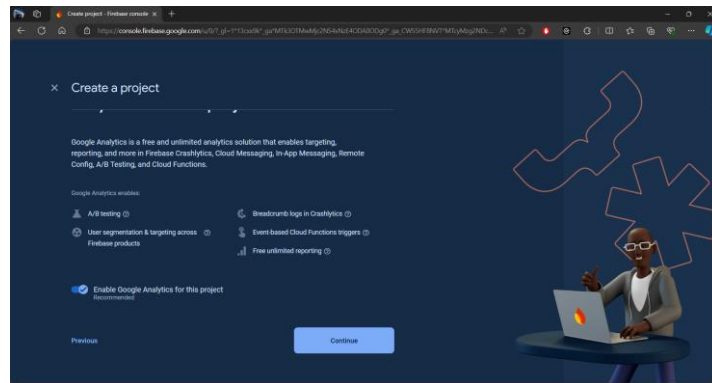
**Figure 5.34 Google Firebase Database Setup II**

- Click on “Create a project” to start creating a new project.



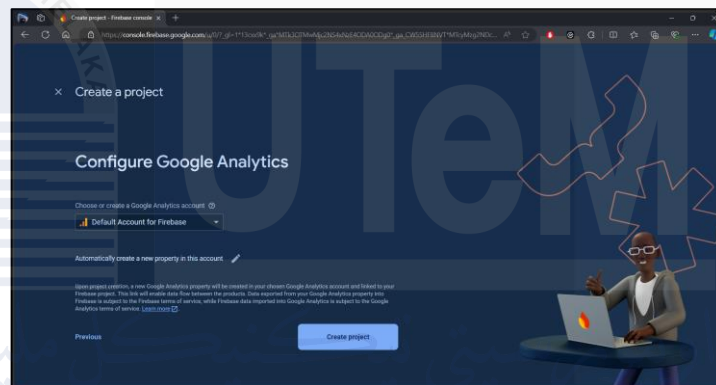
**Figure 5.35 Google Firebase Database Setup III**

- Enter the project name you wish to use. Then click Continue.



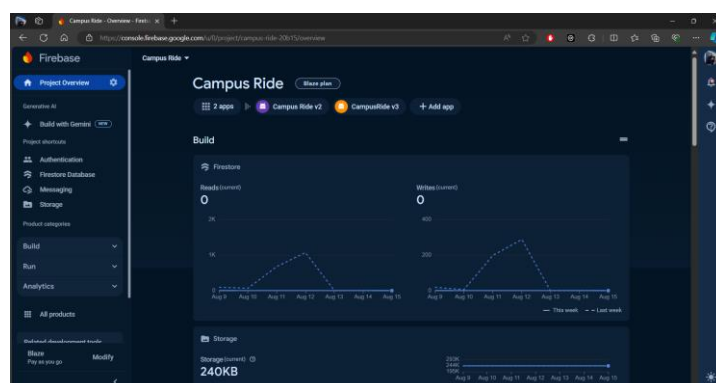
**Figure 5.36 Google Firebase Database Setup IV**

- Click Continue. Make sure to Enable Google Analytics.



**Figure 5.37 Google Firebase Database Setup V**

- Choose Default Account for Firebase then click Create project. Wait till it finishes and go to the console of the project.



**Figure 5.38 Google Firebase Database Setup VI**

- This is the console showing the project creation is successful.

#### 5.4 Version Control Procedure

The project will have its source code version maintained by routinely backing it up to an external flash drive. Every change made, this backup will take place, and after each backup, the version number will increase by one. For example, the version v1.0 will change to v1.1 in the following backup. This procedure is in place to protect against any potential mishaps that can cause the project or folder to get lost or corrupted. Using these backups is a preventative approach to lessen these hazards.

#### 5.5 Implementation Status

No	Module Name	Description	Duration to Complete
1	Login Module	<ul style="list-style-type: none"> <li>System will require users to log in before access the application</li> <li>Require matrix number and password</li> <li>Logout functionality will redirect users to login page.</li> </ul>	5 days
2	Register Module	<ul style="list-style-type: none"> <li>Users need to register their account before using the system</li> <li>Registration will require a valid matric number, username and password.</li> <li>Users must confirm their password during registration.</li> </ul>	5 days
3	Searching Module	<ul style="list-style-type: none"> <li>System will search for available drivers to give a ride to the users.</li> </ul>	6 days
4	Wallet Module	<ul style="list-style-type: none"> <li>Users can manage their in-app wallet balance.</li> <li>This module allows users to reload e-wallets.</li> <li>E-Wallet shows transactions history and current balance.</li> </ul>	7days



5	Ride History Module	<ul style="list-style-type: none"> <li>• The system will display a history of all rides taken by the user.</li> <li>• Each entry includes ride details such as date, time, drop and pickup location.</li> <li>• Users can view ride history data.</li> </ul>	3 days
6	Live Tracking Module	<ul style="list-style-type: none"> <li>• The system provides real-time tracking of the user's ride.</li> <li>• Users can view the current location of their self during ride on the map.</li> </ul>	7 days
7	Sharing Module	<ul style="list-style-type: none"> <li>• Users can share the ride details with others.</li> <li>• Ride sharing option are available for multiple users going in the same direction.</li> </ul>	10 days
8	Google APIs	<ul style="list-style-type: none"> <li>• This module integrates Google APIs for mapping, location, and route services. It uses Google Maps for tracking rides, Google Places for selecting locations, and Google Directions for finding the best routes.</li> </ul>	4 days

## 5.6 Conclusion

In summary, this implementation phase usually comes after the project's planning and design stages and before the closing and assessment phases. This dynamic phase calls for thorough coordination and supervision to guarantee that the project's goals are met successfully and efficiently.

## CHAPTER 6: TESTING

### 6.1 Introduction

In this chapter, the Campus Ride Application will go through software testing. This stage is where the software product is carefully tested to find and fix any mistakes or problems before it is given to users or clients. This step makes sure that the software meets the set requirements, works as expected, and gives a good and reliable experience for users. The goal of this testing phase is to make sure the application works correctly without any errors.

### 6.2 Test Plan

A test plan describes the method, scope, objectives, resources, and timeline for testing a software application or system. It serves as a roadmap for testing throughout the Software Development Life Cycle (SDLC). A strong test plan ensures that testing is systematic, planned, and meets the project's goals and needs. It also acts as a valuable reference for everyone to understand the testing objectives, steps, and expectations, resulting in more effective and successful testing.

#### 6.2.1 Test Organization

A test organization establishes the roles and responsibilities for various tasks within the testing process. It describes the roles, facilities, and actions associated with testing. It also specifies the skills and knowledge required by persons performing these tasks. This section has two primary aims. The tester should focus on the first goal while reviewing both. The test manager is responsible for ensuring that the project proceeds smoothly. Meanwhile, the tester will evaluate the system based on its assigned

interface and functionality. The results of these tests will be recorded for future development.

Tester_ID	Name	Roles	Responsibilities
T01	Fikri Fadzil	App Developer	Creating the application's front and back ends, executing it, integrating it, and doing testing
T02	Badrul Muhymin	Software Tester	Responsible for testing how the modules of the application work and do not work.
T03	Khalis Zakwan	End User	Responsible for testing the application as if you were the user.

**Table 6.1 Test Organization**

### 6.2.2 Test Environment

An environment that is managed and configured to replicate real-world software application or system usage situations is called a test environment. It provides a platform for testing the software's functionality and performance to make sure it works as planned before it's distributed to end users or put into production. Test environments are necessary to do comprehensive and accurate testing without modifying the real operational environment.

### 6.2.2.1 Environment Setup

The environment's configuration serves as a framework for monitoring the testing of this mobile application, which is done to ensure that each module is operating as intended.

### 6.2.2.2 Application Software

The term "application software" describes a computer program designed to carry out a specific job independent of computer-related duties, often for end-user usage. The Campus Ride Application's compatible applications are listed in the table below. The application software utilized to create this Blood Care Mobile Application is displayed in Table 6.2.

<b>Application Software</b>	User Authentication
	Wallet Module
	Ride History
	Search Module
	Sharing Module
	Live Tracking Module
	Google APIs

**Table 6.2 Application Software**

### 6.2.2.3 System Software

This mobile application was created using several different software tools. The Campus Ride Application was created using the software shown in the table below. The System Software utilized to create the Campus Ride Application is displayed in Table 6.

<b>System Software</b>	Android Studio
	Android Emulator
	Microsoft Edge

**Table 6.3 System Software**

#### 6.2.2.4 System Hardware

The System Hardware utilized to create this Blood Care Mobile Application is displayed in Table 6.4.

<b>System Hardware</b>	Acer Nitro 5
	Poco F4 GT
	USB Type-C
	Custom PC

**Table 6.4 System Hardware**

#### 6.2.3 Test Schedule

A test schedule, in the testing phase of a software development project, is a detailed plan that details the scheduling and sequencing of activities to be completed for testing. It includes information on who will oversee them, when certain testing tasks will be completed, and an estimate of how long each task will take. The test schedule guarantees that testing activities are organized, executed well, and completed within the allotted time frame, making it a crucial component of the project calendar.

The Campus Ride Application's test schedule is displayed in Table 6.5.

<b>Testing Module</b>	<b>Start Date</b>	<b>End Date</b>	<b>Duration</b>
User Authentication	6/8/2024	7/8/2024	1 Day
Wallet Module	8/8/2024	9/8/2024	1 Day
Ride History	10/8/2024	11/8/2024	1 Day
Search Module	12/8/2024	13/8/2024	1 Day
Sharing Module	14/8/2024	15/8/2024	1 Day
Live Tracking Module	16/8/2024	17/8/2024	1 Day
Google APIs	18/8/2024	19/8/2024	1 Day

**Table 6.5 Test Schedule**

### **6.3 Test Strategy**

A Test Strategy is a detailed document in software testing that clearly defines the approach and goals for testing a software application. It answers important questions like what needs to be achieved and how it will be done. In this phase, two types of testing will be performed: dynamic testing and user acceptance testing. The user acceptance testing will involve gathering feedback from end users through questionnaires.

#### **6.3.1 Dynamic Testing**

Dynamic testing is a way to check how a software application behaves when it is running. Unlike static testing, which looks at the code and documents without running the software, dynamic testing focuses on how the software works in real-time. For the Campus Ride Application, only Black Box testing has been done in dynamic testing.

Black box testing is a method where the tester does not need to know how the system works inside. The tester gives input to the system and watches the output. This helps to see how the system responds to different user actions and finds issues like response time, usability, and reliability problems.

#### **6.3.2 User Acceptance Testing**

User Acceptance Testing (UAT), also called application testing or end-user testing, is an important step in the software development process. In this phase, the software is tested by its intended users in a real-world setting. UAT usually happens at the end of the software testing process, just before the software is officially released to its target users. The main goal of UAT is to make sure the software can perform real-world tasks as expected according to the development criteria.

During UAT, users get a chance to use the software before it is officially released. This helps identify any features that might have been missed or any potential issues. UAT can be done in different ways, such as using volunteers from within the organization, involving paid testers, or offering a test version for free download. The

feedback from these users is then sent to the development team, who make any final changes needed before the software is launched.

UAT is effective in ensuring the software is of good quality and stays within budget and deadlines. It also improves transparency with end users. By allowing real interactions with real scenarios and data, UAT can confirm whether the software meets business needs if it is done successfully.

## 6.4 Test Design

The process of creating test cases to check a software system's functionality is called test design. It is a crucial step in the software testing process because it ensures that the tests are thorough and effective at finding flaws.

### 6.4.1 Test Description

The test description section is used to confirm that the system function produces the expected result. Each test description includes a unique identifier, a description, and the expected outcome of the system. The following table lists the test cases for each module. For this Campus Ride Application, the tests are conducted by the end user, Khalis Zakwan, and our team's Software Tester, Badrul Muhymin. The test is run in a testing environment where the testers are given time to test the data according to the test schedule.

#### 6.4.1.1 Test Description for User Authentication

Table 6.6 presents the test case for User Authentication, which includes the Module, Test Case ID, Test Case, and the Expected Result.

Module	Test Case ID	Description	Expected Result
--------	--------------	-------------	-----------------

<b>Login</b>	UAL001	To check the login functionality, if the user logs into the system using the correct username and password.	Login page redirected to Home Page without any issue occurs.
	UAL002	To check the login functionality, if the user logs into the system using the incorrect username and password	An alert dialog will appear, indicating that the user credentials used are incorrect.
	UAL003	To check the functionality, if the user does not fill the username field.	An alert dialog will appear, indicating that the user credentials used are incorrect
	UAL004	To check the functionality, if the user does not fill the password field.	An alert dialog will appear, indicating that email formatted address is bad.
	UAR001	To check the functionality, if the account can be signed up using a matric number, name, password and confirm password.	A message will appear, indicating that registration is successful and directing the user to the login page.
	UAR002	To check the functionality if the	A message will appear for empty fields, reminding



<b>Register</b>		matric number field is empty	the user to fill them out before registering.
	UAR003	To check the functionality if the full name field is empty.	A message will appear for empty fields, reminding the user to fill them out before registering.
	UAR004	To check the functionality if the password is empty	A message will appear for empty fields, reminding the user to fill them out before registering.
	UAR005	To check the functionality if the confirm password is empty	A message will appear for empty fields, reminding the user to fill them out before registering.
	UAR006	To check the functionality if the password and confirm password do not match.	A message will appear indicating that the password and confirm password do not match.

**Table 6.6 Test Case for User Authentication**

#### 6.4.1.2 Test Description for Wallet Module

Table 6.7 presents the test case for Wallet Module, which includes the Module, Test Case ID, Test Case, and the Expected Result.

Module	Test Case ID	Description	Expected Result
Wallet	WM001	To check the display of e-wallet balance when the user accesses the wallet page.	The wallet page should show the current e-wallet balance correctly without any issues.
	WM002	To check the functionality of the reload button when pressed by the user.	The reload button should update the e-wallet balance to reflect any changes and the updated balance should be displayed.
	WM003	To check the display of transaction history.	The transaction history section should list all previous transactions accurately.
	WM004	To check the behavior when no transaction history exists.	The transaction history section should display a message indicating that there are no transactions available.

**Table 6.7 Test Case for Wallet Module**

#### 6.4.1.3 Test Description for Ride History Module

Table 6.8 presents the test case for Ride History, which includes the Module, Test Case ID, Test Case, and the Expected Result.

Module	Test Case ID	Description	Expected Result
--------	--------------	-------------	-----------------

<b>Ride History</b>	RH001	To check the display of ride history with successful rides.	The ride history should list all successful rides with details including date, time, pickup location, and drop-off location accurately displayed.
	RH002	To check the display of ride history with cancelled rides.	The ride history should list all cancelled rides with details including date, time, pickup location, and drop-off location accurately displayed, and indicate that the ride was cancelled.
	RH003	To check the behavior when no ride history exists.	The ride history section should display a message indicating that no rides are available or no ride history exists.

**Table 6.8 Test Case for Ride History Module**

#### 6.4.1.4 Test Description for Search Module

Table 6.9 presents the test case for Search Module, which includes the Module, Test Case ID, Test Case, and the Expected Result.

Module	Test Case ID	Description	Expected Result
Search Module	SEM001	To check the auto-filled place suggestions when the user starts typing the drop-off location.	The system should display relevant place suggestions based on the user's input in a dropdown list.
	SEM002	To test the selection of a place from the suggestion list.	When the user clicks on a location from the suggestion list, the system should populate the drop-off location field with the selected place and redirect or update the page to reflect the chosen location.
	SEM003	To verify that the suggestion list is updated dynamically as the user types more characters.	The suggestion list should update in real-time based on the input, showing relevant suggestions as the user types.

**Table 6.9 Test Case for Search Module**

### 6.4.1.5 Test Description for Sharing Module

Table 6.10 presents the test case for Sharing Module, which includes the Module, Test Case ID, Test Case, and the Expected Result.

Module	Test Case ID	Description	Expected Result
Sharing Module	SHM001	To check if the system lists all matching orders where pickup and drop-off locations are the same.	The system should display a list of matching orders with the same pickup and drop-off locations for both parties if they have enabled ride-sharing permission.
	SHM002	To verify the functionality of selecting a person to share a ride with	The first person should be able to select a matching order from the list and choose with whom they want to share the ride.
	SHM003	To check if the second person receives a ride-sharing request.	The second person should receive a notification or prompt for the ride-sharing request from the first person, detailing the shared ride information.

	SHM004	To test the approval process for the ride-sharing request.	The second person should have the option to approve or decline the ride-sharing request. If they approve, the system should proceed with the ride-sharing process.
	SHM005	To ensure the price is correctly divided between both parties if the ride-sharing request is approved.	If the ride-sharing request is approved, the total price of the ride should be divided by two, with each person being charged half the original price.
	SHM006	To verify the behavior when either party has not enabled ride-sharing permissions in their settings.	If either party has not enabled the ride-sharing permission, the ride-sharing option should not be available, and no matching orders should be listed.
	SHM007	To test the scenario where no matching orders are available for ride-sharing.	If there are no orders with the same pickup and drop-off locations, the system should

			inform the user that no matching rides are available for sharing.
--	--	--	---

**Table 6.10 Test Case for Sharing Module**

#### 6.4.1.6 Test Description for Live Tracking Module

Table 6.11 presents the test case for Live Tracking Module, which includes the Module, Test Case ID, Test Case, and the Expected Result.

Module	Test Case ID	Description	Expected Result
<b>Live Tracking Module</b>	LTM001	To verify that the user's location is tracked in real-time from the start to the drop-off location.	To verify that the user's location is tracked in real-time from the start to the drop-off location.
	LTM002	To check the accuracy of the location tracking during the trip.	The displayed location on the live tracking map should be accurate and reflect the user's actual position in real-time.
	LTM003	To test the functionality of the emergency button on the live tracking page.	Pressing the emergency button should initiate a call to the person saved in the database as the emergency contact for the user.

**Table 6.11 Test Case for Live Tracking Module****6.4.1.7 Test Description for Google APIs**

Table 6.12 presents the test case for Google APIs, which includes the Module, Test Case ID, Test Case, and the Expected Result.

<b>Module</b>	<b>Test Case ID</b>	<b>Description</b>	<b>Expected Result</b>
	GA001	To verify that the Google Map is displayed correctly on the page.	The map should load and display correctly on the page, centered on the specified location or default location if not provided.
Google APIs	GA002	To check the functionality of the auto-complete location feature.	As the user types into the location input field, the auto-complete suggestions should appear, offering relevant locations based on the input.

**Table 6.12 Test Case for Google APIs**



## 6.4.2 Test Data for Dynamic Testing

Test data is the information given to a software program during testing. This data either affects how the software runs or is affected by it. Test data has two main uses: first, in positive testing, it checks if the software gives the right results with certain inputs; second, in negative testing, it tests how the software handles unusual or unexpected inputs.

### 6.4.2.1 Test Data for User Authentication

System : Campus Ride Application Version : v1  
Module/Unit : User Authentication Revision : -  
Processed by : Khalis Zakwan Date : 6/8/2024

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Results
UAL001	Login to the system using valid credentials	<ol style="list-style-type: none"><li>1. Enter matric number and password.</li><li>2. Login to the system by pressing login button.</li></ol>	Matric number: b032210001 Password: 111111	Login successfully and directed to the home page of the application.

UAL002	Login to the system using invalid credentials	<ol style="list-style-type: none"> <li>1. Enter matric number with wrong password or wrong matric number with valid password</li> <li>2. Login to the system by pressing login button</li> </ol>	Matric number: b03221001 Password: 111112	Alert dialog showing up indicating the login credentials is invalid.
UAL003	Login to the system without fill in the matric number	<ol style="list-style-type: none"> <li>1. Enter password only</li> <li>2. Login to the system by pressing login button.</li> </ol>	Matric number: Password: 111111	Alert dialog showing up indication the login credentials is invalid.
UAL004	Login to the system without fill in the password	<ol style="list-style-type: none"> <li>1. Enter matric number only</li> <li>2. Login to the system by pressing login button.</li> </ol>	Matric number: b032210001 Password:	Alert dialog showing up indication the login credentials is invalid

UAR001	Register account with valid information	<ol style="list-style-type: none"> <li>1. Enter all the information needed.</li> <li>2. Register the account by pressing register button.</li> </ol>	Matric number: b032210002 Full name: Husna Password: 111111 Confirm Password: 111111	A register successful message will shown up and redirect to the login page.
UAR002	Register account with empty matric number	<ol style="list-style-type: none"> <li>1. Enter all the information needed except matric number</li> <li>2. Register the account by pressing register button</li> </ol>	Matric number: Fullname: Husna Password: 111111 Confirm Password: 111111	A message appeared for empty field to remind user that the field is still empty.
UAR003	Register account with empty full name	<ol style="list-style-type: none"> <li>1. Enter all the information needed except full name</li> </ol>	Matric number: b032210002 Fullname: Password: 111111	A message appeared for empty field to remind user that the field is still empty.

		2. Register the account by pressing register button	Confirm Password: 111111	
UAR004	Register account with empty password	<ol style="list-style-type: none"> <li>1. Enter all the information needed except password.</li> <li>2. Register the account by pressing register button</li> </ol>	Matric number: b032210002 Fullname: Husna Password: Confirm Password: 111111	A message appeared for empty field to remind user that the field is still empty.
UAR005	Register account with empty confirm password	<ol style="list-style-type: none"> <li>1. Enter all the information needed except confirm password.</li> <li>2. Register the account by</li> </ol>	Matric number: b032210002 Fullname: Husna Password: 111111 Confirm Password:	A message appeared for empty field to remind user that the field is still empty.

		pressing register button		
UAR006	Register account but unmatched password and confirm password.	<ol style="list-style-type: none"> <li>1. Enter all the information needed except matric number</li> <li>2. Register the account by pressing register button</li> </ol>	Matric number: b032210002 Fullname: Husna Password: 111111 Confirm Password: 111112	A message will appear indicating that the password and confirm password do not match.

### 6.4.2.2 Test Data for Wallet Module

System : Campus Ride Application

Version : v1

Module/Unit : User Authentication

Revision : -

Processed by : Khalis Zakwan

Date : 8/8/2024

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Results
WM001	Checking the e-wallet balance.	1. Open drawer on home page and click on wallet	Balance: RM0.00	E-Wallet balance should be 0 align with the test data.
WM002	Reloading the e-wallet	1. Open drawer on home page and click on wallet 2. Click on the Reload button. 3. Insert amount to reload e-wallet	Amount: RM50.00	Balance should immediately update to the total of the amount reloaded and last balance.
WM003	Inspecting the transaction history	1. Open drawer on home page and click on wallet	-	All the transaction must be recorded in the history.

		2. Slide down the transaction history under the reload button to view it.		
WM004	Inspecting the transaction history even does not reload any amount	<ol style="list-style-type: none"> <li>1. Open drawer on home page and click on wallet</li> <li>2. Slide down to see the transaction history</li> </ol>	-	A message indicated that no transaction has been made must be shown there.

### 6.4.2.3 Test Data for Ride History Module

System : Campus Ride Application

Version : v1

Module/Unit : User Authentication

Revision : -

Processed by : Khalis Zakwan

Date : 10/8/2024

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Results
RH001	Inspecting the record of the ride	1. Open drawer on the homepage and click on ride history	-	All the ride history must be shown there
RH002	Inspecting the record of the cancelled ride	1. Open drawer on the homepage and click on ride history	-	All the cancelled ride history must be shown there
RH003	Inspecting the ride history even there is no ride has been made	1. Open drawer on the homepage and click on ride history	-	A message indicated that no ride has been made must be shown there.



#### 6.4.2.4 Test Data for Searching Module

System : Campus Ride Application

Version : v1

Module/Unit : User Authentication

Revision : -

Processed by : Khalis Zakwan

Date : 12/8/2024

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Results
SEM001	Checking the auto-filled suggestion of the search function	<ol style="list-style-type: none"> <li>1. Click on search bar at the homepage</li> <li>2. Insert the place to be the drop off location</li> </ol>	Place: UTeM Satria Key insert: Satria	Auto filled system will suggest “UTeM Satria” on the suggestion list even user just insert Satria on the search bar
SEM002	Select one of the locations on the suggestion list as drop off location	<ol style="list-style-type: none"> <li>1. Click on search bar at the homepage</li> <li>2. Insert the place to be the drop off location</li> <li>3. Click on the location in the suggestion list</li> </ol>	Place: UTeM Satria	System will make the location selected as the drop off location and proceed to the searching driver.


SEM003	Ensure the suggestion list update dynamically with the user input	<ol style="list-style-type: none"> <li>1. Click on search bar at the homepage</li> <li>2. Insert the place to be the drop off location</li> </ol>	First key insert: U Second key insert: T	The system should immediately suggest a place that starts with "UT" as soon as the user types the letter "T".
--------	---	---	---	---

#### 6.4.2.5 Test Data for Sharing Module

System : Campus Ride Application      Version : v1  
 Module/Unit : User Authentication      Revision : -  
 Processed by : Khalis Zakwan      Date : 14/8/2024

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Results
SHM001	List out all the matching order available	<ol style="list-style-type: none"> <li>1. Search for the drop off location</li> <li>2. Click the location on the suggestion list</li> </ol>	Drop off location: UTeM Satria Pickup Location: FTMK Account holder: Syakirin Matching Order: Hayati	Accountholder can see all the matching order that can be share the ride.

		3. System will list the matching order		
SHM002	Selecting the person to share with	<ol style="list-style-type: none"> <li>1. Search for the drop off location</li> <li>2. Click the location on the suggestion list</li> <li>3. System will list the matching order</li> <li>4. Click on the selected user to share with</li> </ol>	<p>Drop off location: UTeM Satria</p> <p>Pickup Location: FTMK</p> <p>Account holder: Syakirin</p> <p>Matching Order: Hayati</p>	The accountholder has chosen the person to share the ride with, and the system should record the request.
SHM003	Sharing request notification should appear on the other person's device.	<ol style="list-style-type: none"> <li>1. Wait for the notification to appear on the selected person's device.</li> </ol>	-	The selected person's device must display a notification informing them that there is a sharing request for their ride.

SHM004	<p>Approve or reject the approval of the sharing request</p> 	<ol style="list-style-type: none"> <li>1. Wait for the notification to appear on the selected person's device.</li> <li>2. Click on the notification to make the approval appear</li> <li>3. Click Approve or Reject button for the sharing request.</li> </ol>	<p>Requester: Syakirin Approval Given by: Hayati Approval status: Approve</p>	<p>Both will share the ride if the approval status is approved and does not share if the approval status is declined.</p>
SHM005	<p>Ensure the fare of the rides is deducted when sharing the ride</p>	<ol style="list-style-type: none"> <li>1. Wait for the notification to appear on the selected person's device.</li> <li>2. Click on the notification to</li> </ol>	<p>Requester: Syakirin Approval Given by: Hayati Approval status: Approve</p>	<p>The initial fare will be split in half, with each account paying an equal share.</p>

		<p>make the approval appear</p> <ol style="list-style-type: none"> <li>3. Click Approve button to share the ride</li> </ol>		
SHM006	Verify sharing settings if the sharing permission is turned off.	<ol style="list-style-type: none"> <li>1. Turn off sharing permission of the user setting</li> <li>2. Find a ride</li> </ol>	<p>Sharing permission: False</p> <p>Drop off location: UTeM Satria</p>	The system will not display any matching orders because the sharing permission is turned off.
SHM007	No matching order are available for ridesharing	<ol style="list-style-type: none"> <li>1. Search and select for a drop-off location</li> </ol>	<p>Drop off location: UTeM Satria</p> <p>Pickup location: FTMK</p>	The system will not display any matching orders if none are available.

### 6.4.2.6 Test Data for Live Tracking Module

System : Campus Ride Application

Version : v1


Module/Unit : User Authentication

Revision : -

Processed by : Khalis Zakwan

Date : 16/8/2024

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Results
LTM001	Verifying the user's location is tracked in real-time from the start to the drop-off location.	<ol style="list-style-type: none"> <li>1. Request for a ride</li> <li>2. Confirm ride with the driver</li> <li>3. Wait for the driver arrival</li> <li>4. Trip start</li> </ol>	Drop off location: UTeM Satria Pickup Location: FTMK	The user can track their location on the live tracking page.
LTM002	Checking the accuracy of the live tracking feature	<ol style="list-style-type: none"> <li>1. Request for a ride</li> <li>2. Confirm ride with the driver</li> <li>3. Wait for the driver arrival</li> <li>4. Trip start</li> </ol>	Drop off location: UTeM Satria Pickup Location: FTMK	Live tracking will refresh every second to ensure accuracy.

LTM003	Testing the emergency button 	<ol style="list-style-type: none"> <li>1. Request for a ride</li> <li>2. Confirm ride with the driver</li> <li>3. Wait for the driver arrival</li> <li>4. Trip start</li> <li>5. Click the emergency button</li> </ol>	Drop off location: UTeM Satria Pickup Location: FTMK	The user's device will immediately open the default phone app, with the number already displayed, ready for the call button to be clicked to make the call.
--------	---	--	---	---

### 6.4.2.7 Test Data for Google APIs

System : Campus Ride Application

Version : v1

Module/Unit : User Authentication

Revision : -

Processed by : Khalis Zakwan

Date : 18/8/2024

Test Case ID	Test Scenario	Test Steps	Test Data	Expected Results
GA001	Verifying the Google Map is displayed correctly on the home page	<ol style="list-style-type: none"> <li>1. Login to the system</li> <li>2. Google Map is on the homepage</li> </ol>	-	The map should load and display accurately without any error
GA002	Checking the auto complete location feature	<ol style="list-style-type: none"> <li>1. Click on the search bar at the home page</li> <li>2. Enter the drop-off location without typing it in full.</li> </ol>	Key search: UTeM	The auto-complete feature should list all suggestions for places that start with "UTeM."



## 6.5 User Acceptance Testing

User Acceptance Testing (UAT), also known as acceptance testing, is a crucial phase in software development. It is the final testing stage before the software is released to users or customers. The primary goal of UAT is to ensure that the software is ready for real-world use and performs as expected. For this UAT, feedback was gathered from 32 participants through questionnaires distributed via Google Forms. The testing took place over 5 days, from August 15, 2024, to August 20, 2024. The participants were primarily UTeM students, including friends and their acquaintances. The testing process included live sessions on Discord, where users interacted with the app in real-time, as well as watching a video that showcased the app's features. Responses and feedback were collected using Google Forms, enabling users to share their opinions and experiences with the application.

### 6.5.1 Questionnaires for User Acceptance Testing

The survey, created using Google Forms, consists of 24 questions divided into six sections. These sections cover respondent information, their evaluation of the system's ease of use and usefulness, the system's features, its reliability, and their overall attitude toward the system. The survey will be shared with people nearby through social media platforms. Table 6.13 displays the questionnaires provided to the end users.

No	Questions	Section
1	Gender	Respondent Information
2	Age	Respondent Information
3	The Campus Ride app is flexible to interact with.	Perceived Ease of Use (EU)
4	I find it easy to get the Campus Ride app to do what I want to do.	Perceived Ease of Use (EU)
5	It is easy to become skilled at using the Campus Ride app.	Perceived Ease of Use (EU)
6	I find the Campus Ride app easy to use.	Perceived Ease of Use (EU)

7	Interaction with the Campus Ride app is clear and understandable.	Perceived Ease of Use (EU)
8	Using the Campus Ride app enables me to easily find and share rides.	Perceived Usefulness (PU)
9	I find the Campus Ride app useful for my daily commute.	Perceived Usefulness (PU)
10	Using the Campus Ride app enhances my effectiveness in organizing my rides.	Perceived Usefulness (PU)
11	The Campus Ride app makes it easier to connect with other students for shared rides.	Perceived Usefulness (PU)
12	The Campus Ride app makes it easier to reduce travel costs and time.	Perceived Usefulness (PU)
13	The Campus Ride app provides clear instructions for find and offering rides.	Capability (CP)
14	Adding or searching for ride details is straightforward in the Campus Ride app.	Capability (CP)
15	The features of the Campus Ride app meet my ride-sharing needs.	Capability (CP)
16	I trust the Campus Ride app to keep my ride details secure.	Trustworthiness (TW)
17	The Campus Ride app provides security for my personal information.	Trustworthiness (TW)
18	I feel safe sharing my ride information using the Campus Ride app.	Trustworthiness (TW)
19	I enjoy using the Campus Ride app.	Attitude (ATT)
20	It is convenient for me to use the Campus Ride app.	Attitude (ATT)
21	I find it desirable to learn more about using the Campus Ride app.	Attitude (ATT)
22	I intend to use the Campus Ride app for my daily commute.	Intention to Use (IU)
23	I plan to continue using the Campus Ride app to share rides with others.	Intention to Use (IU)

24	I will recommend the Campus Ride app to my friends and classmates.	Intention to Use (IU)
----	--	-----------------------

**Table 6.13 User Acceptance Questionnaires**

## 6.6 Test Result and Analysis

The software testing process should include both test results and analysis. This involves evaluating the outcomes of the tests and interpreting the data collected to gain insights into the functionality and quality of the software. In summary, test results and analysis are essential in the software development lifecycle because they provide valuable information about the software's performance, reliability, and adherence to specifications. These efforts help create a more dependable product for end users while also improving the overall quality of the software over time.

### 6.6.1 Test Result for Dynamic Testing

All the test cases that were developed have passed successfully, with no failures reported.

#### 6.6.1.1 Test Result for User Authentication

Table 6.14 shows the Test Result and Analysis for User Authentication.

Test Case ID	Actual Result	Pass	Fail
UAL001	Login Successfully and screen directed to the home page.	✓	
UAL002	Alert dialog showing up indicating the login credentials is invalid.	✓	
UAL003	Alert dialog showing up indicating the login credentials is invalid.	✓	
UAL004	Alert dialog showing up indicating the login credentials is invalid.	✓	
UAR001	A register successful message will show up and redirect to the login page.	✓	
UAR002	A message appeared for empty field to remind user that the field is still empty	✓	

UAR003	A message appeared for empty field to remind user that the field is still empty	✓	
UAR004	A message appeared for empty field to remind user that the field is still empty	✓	
UAR005	A message appeared for empty field to remind user that the field is still empty	✓	
UAR006	A message will appear indicating that the password and confirm password do not match.	✓	

**Table 6.14 Test Result and Analysis for User Authentication**

### 6.6.1.2 Test Result for Wallet Module

Table 6.15 shows the Test Result and Analysis for Wallet Module.

Test Case ID	Actual Result	Pass	Fail
WM001	E-Wallet balance is RM0.00	✓	
WM002	Balance update immediately after reloading the e-wallet.	✓	
WM003	All transaction recorded in the transaction history.	✓	
WM004	A message appears in the transaction list showing that no transaction has been made.	✓	

**Table 6.15 Test Result and Analysis for Wallet Module**

### 6.6.1.3 Test Result for Ride History Module

Table 6.16 shows the Test Result and Analysis for Ride History.

Test Case ID	Actual Result	Pass	Fail
RH001	All ride history is listed	✓	
RH002	All cancelled ride history is listed	✓	

RH003	A message appears in the transaction list showing that no ride has been made.	✓	
-------	---	---	--

**Table 6.16 Test Result and Analysis for Ride History Module**

#### 6.6.1.4 Test Result for Searching Module

Table 6.17 shows the Test Result and Analysis for Searching Module.

Test Case ID	Actual Result	Pass	Fail
SEM001	Auto completed place prediction ran successfully.	✓	
SEM002	Selected location registered as drop off location.	✓	
SEM003	The place which does not same as the user input removed immediately in auto completed place prediction list.	✓	

**Table 6.17 Test Result and Analysis for Searching Module**

#### 6.6.1.5 Test Result for Sharing Module

Table 6.18 shows the Test Result and Analysis for Sharing Module.

Test Case ID	Actual Result	Pass	Fail
SHM001	All matching order listed in the list.	✓	
SHM002	The selected person to share with can be chosen.	✓	
SHM003	A notification appeared on the selected person devices.	✓	
SHM004	Approval request permission given and immediately inform the requester either the result is approved or rejected.	✓	

SHM005	The fare indeed split into half and equal share.	✓	
SHM006	Matching order list does not appear since the sharing permission is turned off.	✓	
SHM007	Matching order list does not appear since there is no matching order available.	✓	

**Table 6.18 Test Result and Analysis for Sharing Module**

### 6.6.1.6 Test Result for Live Tracking Module

Table 6.19 shows the Test Result and Analysis for Live Tracking Module.

Test Case ID	Actual Result	Pass	Fail
LTM001	Live tracking of the current user location tracked	✓	
LTM002	The accuracy is accurate since it refreshes every second	✓	
LTM003	Phone app open immediately after click the emergency button	✓	

**Table 6.19 Test Result and Analysis for Live Tracking Module**

### 6.6.1.7 Test Result for Google APIs

Table 6.20 shows the Test Result and Analysis for Google APIs.

Test Case ID	Actual Result	Pass	Fail
GA001	Google Map load accurately	✓	
GA002	Auto complete place prediction function well.	✓	

**Table 6.20 Test Result and Analysis for Google APIs**

### 6.6.1.8 Summary of Recorded Test Case

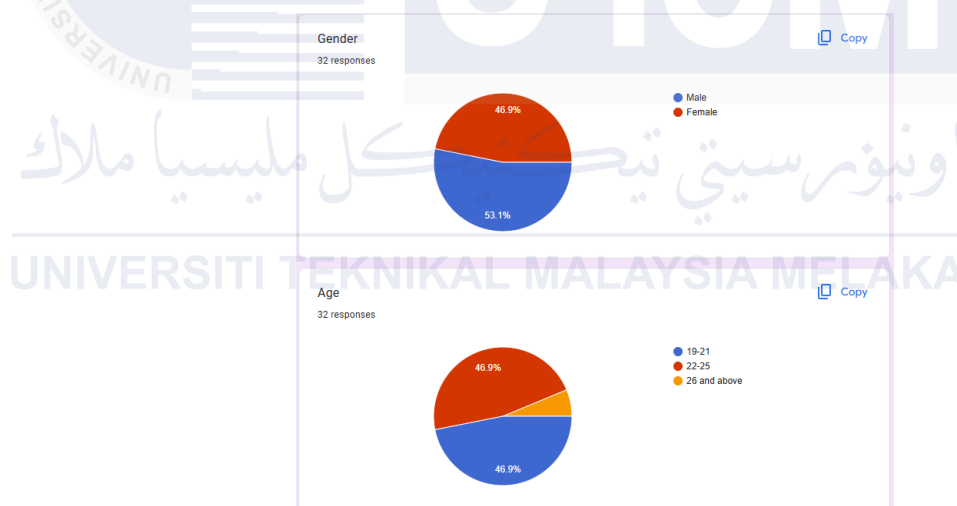
Test Case	Total Success
User Authentication	10

Wallet Module	4
Ride History Module	3
Searching Module	3
Sharing Module	7
Live Tracking Module	3
Google APIs	2
<b>Total</b>	<b>32</b>

**Table 6.21 Summary of Recorded Test Case**

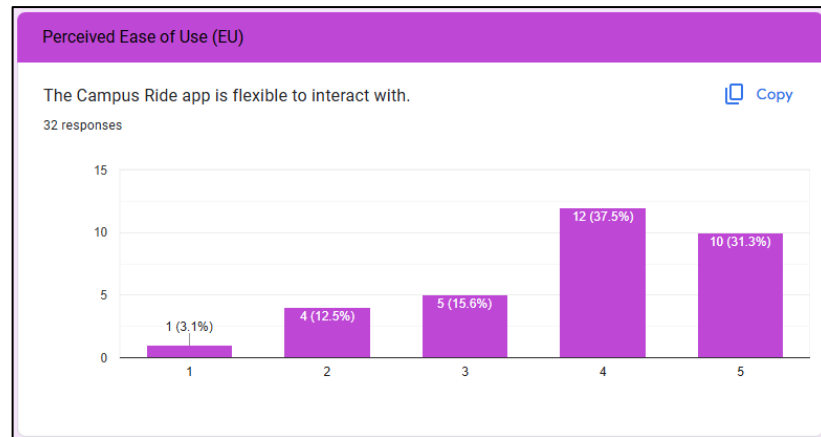
Table 6.21 shows the summary of recorded test case for Campus Ride Application Testing. There are a total of 7 test cases with a total of 32 total success of testing conducted

### 6.6.2 User Acceptance Testing Analysis and Result



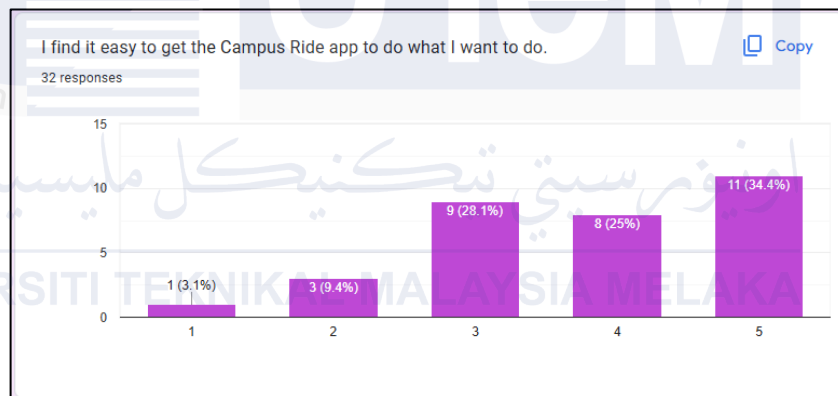
**Figure 6.1 Pie Chart of Questionnaire Question**

Figure 6.1 shows how a group of people is divided by gender and age. Among all the participants, 53.1% are male, and 46.9% are female. When looking at age, 46.9% of the group are between 19-21 years old, another 46.9% are between 22-25 years old, and 6.3% are 26 years old or older. This provides a clear view of the gender and age distribution within the group.



**Figure 6.2 Bar Chart of Questionnaire Question**

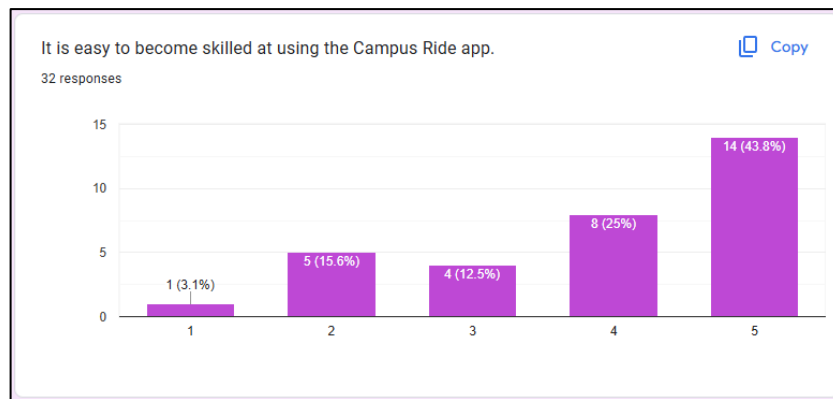
Figure 6.2 represents those 10 respondents strongly agree that the system is flexible to interact with while 12 respondents is agreed, 5 respondents is neutral, 4 respondents is disagreeing, and 1 respondent is totally disagreed.



**Figure 6.3 Bar Chart of Questionnaire Question**

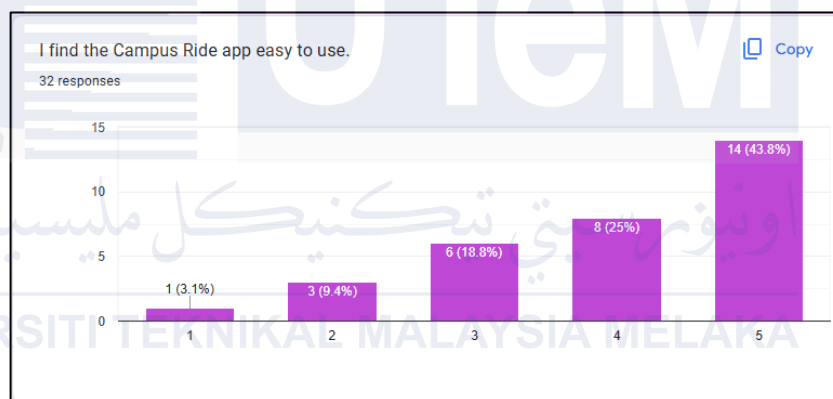
Figure 6.3 represents those 11 respondents strongly agree that it easy to get the system do what they want to do while 8 respondents is agreed, 9 respondents is neutral, 3 respondents is disagreeing, and 1 respondent is totally disagreed.





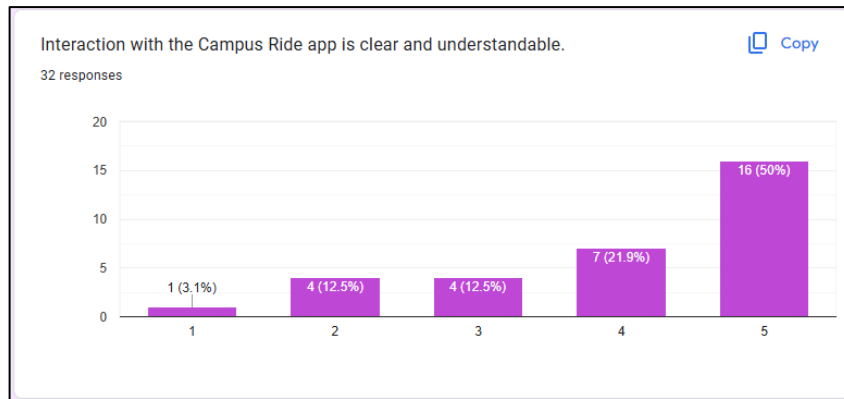
**Figure 6.4 Bar Chart of Questionnaire Question**

Figure 6.4 represents those 14 respondents strongly agree that it is easy to become skilled at using the system while 8 respondents is agreed, 4 respondents is neutral, 5 respondents is disagreeing, and 1 respondent is totally disagreed.



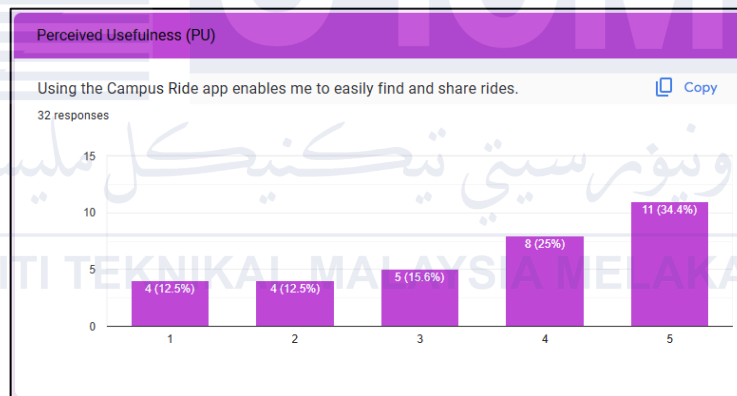
**Figure 6.5 Bar Chart of Questionnaire Question**

Figure 6.5 represents those 14 respondents strongly agree that the system is easy to use while 8 respondents is agreed, 6 respondents is neutral, 3 respondents is disagreeing, and 1 respondent is totally disagreed.



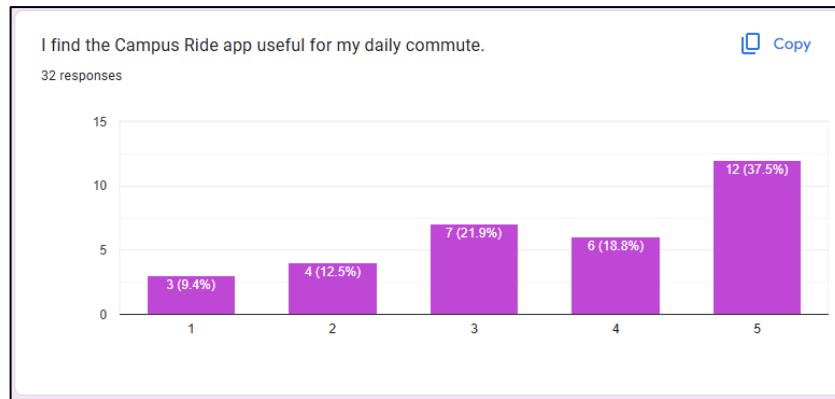
**Figure 6.6 Bar Chart of Questionnaire Question**

Figure 6.6 represents those 14 respondents strongly agree that the interaction with the system is clear and understandable while 7 respondents is agreed, 4 respondents is neutral, 4 respondents is disagreeing, and 1 respondent is totally disagreed.



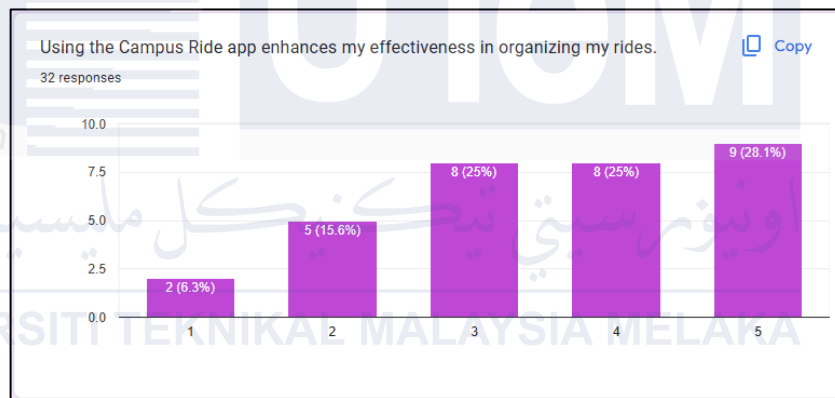
**Figure 6.7 Bar Chart of Questionnaire Question**

Figure 6.7 represents those 11 respondents strongly agree that the system help them to easily find and share ride while 8 respondents is agreed, 5 respondents is neutral, 4 respondents is disagreeing, and 4 respondent is totally disagreed.



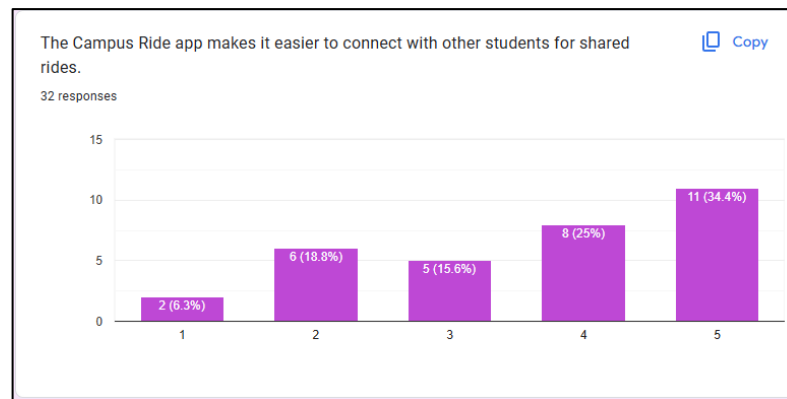
**Figure 6.8 Bar Chart of Questionnaire Question**

Figure 6.8 represents those 12 respondents strongly agree that the system is useful for their daily commute while 6 respondents is agreed, 7 respondents is neutral, 4 respondents is disagreeing, and 3 respondent is totally disagreed.



**Figure 6.9 Bar Chart of Questionnaire Question**

Figure 6.9 represents those 9 respondents strongly agree that the system can enhance their effectiveness in organizing their rides while 8 respondents is agreed, 8 respondents is neutral, 5 respondents is disagreeing, and 2 respondent is totally disagreed.



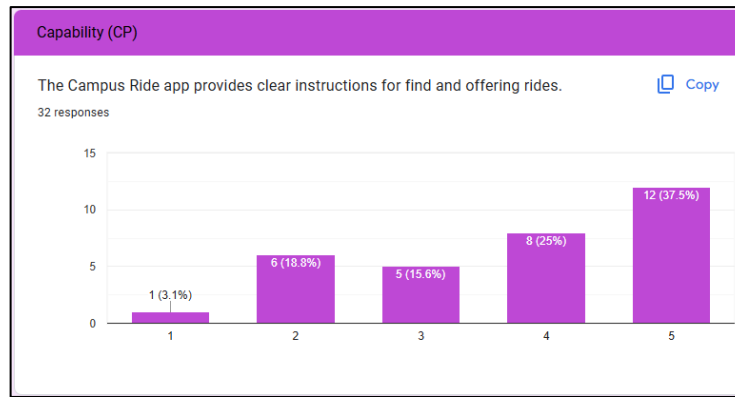
**Figure 6.10 Bar Chart of Questionnaire Question**

Figure 6.10 represents those 11 respondents strongly agree that the system can makes it easier to connect with other student for shared rides while 8 respondents is agreed, 5 respondents is neutral, 6 respondents is disagreeing, and 2 respondent is totally disagreed.



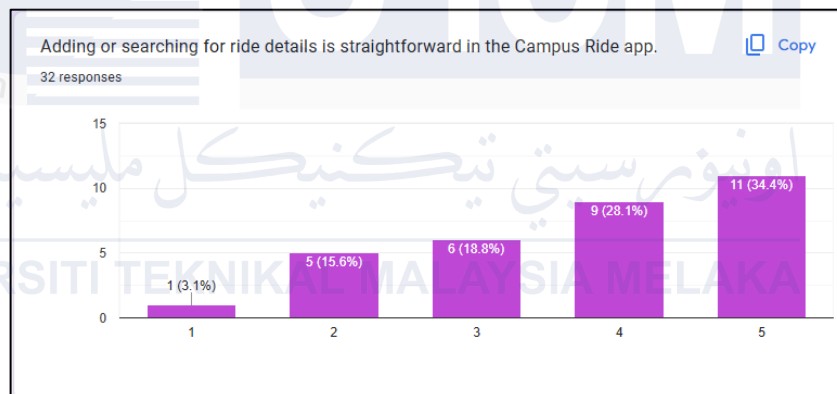
**Figure 6.11 Bar Chart of Questionnaire Question**

Figure 6.11 represents those 12 respondents strongly agree that the system can easier their ride to reduce travel costs and time while 10 respondents is agreed, 4 respondents is neutral, 3 respondents is disagreeing, and 3 respondent is totally disagreed.



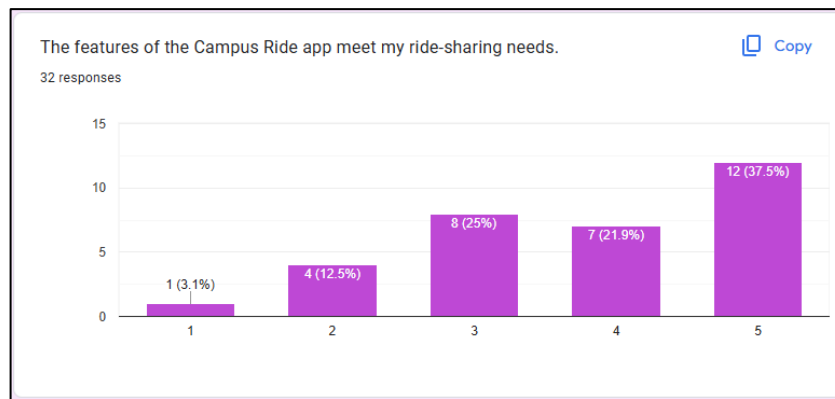
**Figure 6.12 Bar Chart of Questionnaire Question**

Figure 6.12 represents those 12 respondents strongly agree that the provides clear instructions for find and offering rides while 8 respondents is agreed, 5 respondents is neutral, 6 respondents is disagreeing, and 1 respondent is totally disagreed.



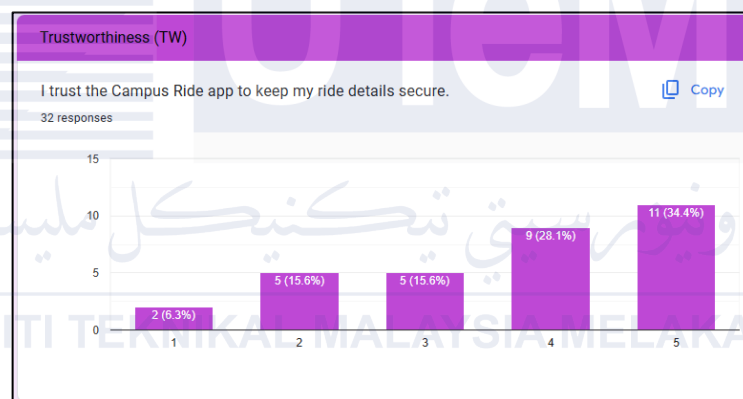
**Figure 6.13 Bar Chart of Questionnaire Question**

Figure 6.13 represents those 11 respondents strongly agree that searching for ride details is straightforward in the system while 9 respondents is agreed, 6 respondents is neutral, 5 respondents is disagreeing, and 1 respondent is totally disagreed.



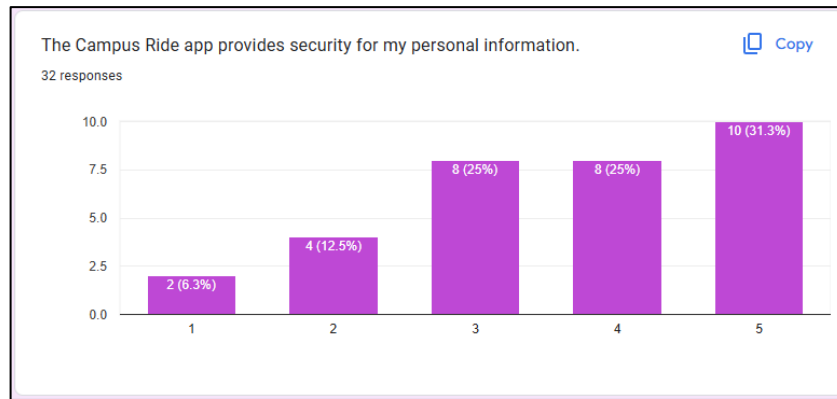
**Figure 6.14 Bar Chart of Questionnaire Question**

Figure 6.14 represents those 12 respondents strongly agree that the system meet their ride sharing needs while 7 respondents is agreed, 8 respondents is neutral, 4 respondents is disagreeing, and 1 respondent is totally disagreed.



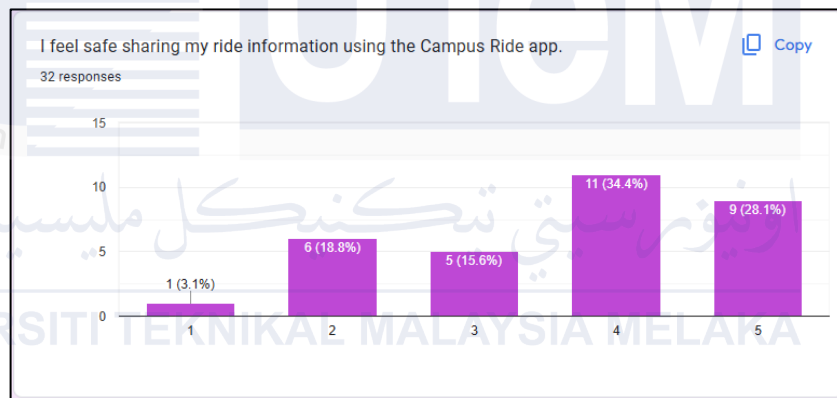
**Figure 6.15 Bar Chart of Questionnaire Question**

Figure 6.15 represents those 11 respondents strongly agree that the system keep their ride details secure while 9 respondents is agreed, 5 respondents is neutral, 5 respondents is disagreeing, and 2 respondent is totally disagreed.



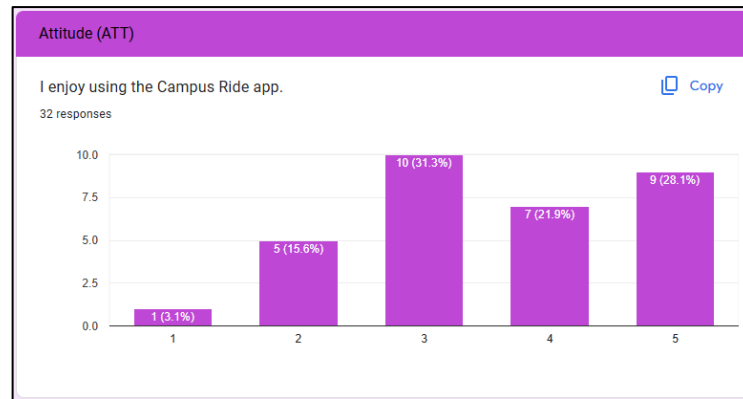
**Figure 6.16 Bar Chart of Questionnaire Question**

Figure 6.16 represents those 10 respondents strongly agree that the provides security for their personal information while 8 respondents is agreed, 8 respondents is neutral, 4 respondents is disagreeing, and 2 respondent is totally disagreed.



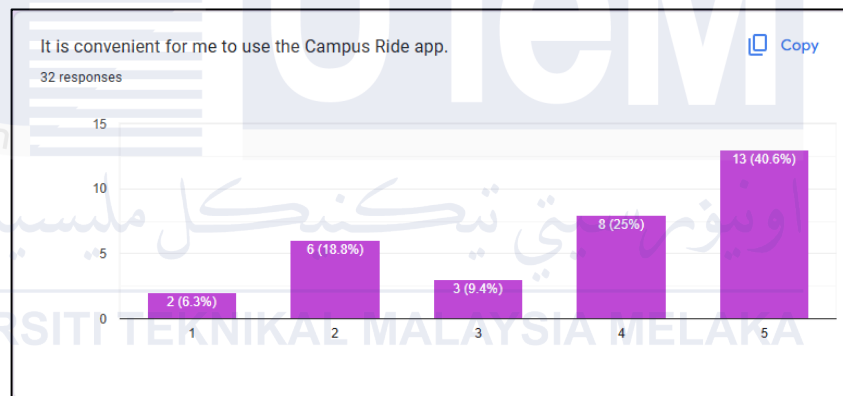
**Figure 6.17 Bar Chart of Questionnaire Question**

Figure 6.17 represents those 9 respondents strongly agree that fell safe sharing their ride information while using the system while 11 respondents is agreed, 5 respondents is neutral, 6 respondents is disagreeing, and 1 respondent is totally disagreed.



**Figure 6.18 Bar Chart of Questionnaire Question**

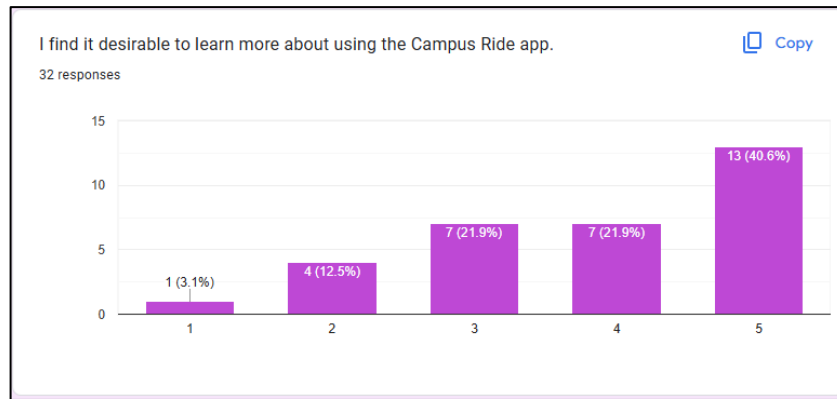
Figure 6.18 represents those 9 respondents strongly agree that they enjoy using the system while 7 respondents is agreed, 10 respondents is neutral, 5 respondents is disagreeing, and 1 respondent is totally disagreed.



**Figure 6.19 Bar Chart of Questionnaire Question**

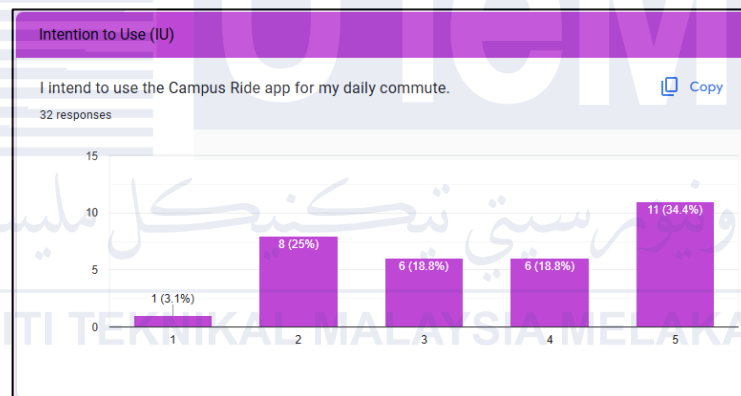
Figure 6.19 represents those 13 respondents strongly agree that it convenient for them to use the system while 8 respondents is agreed, 3 respondents is neutral, 6 respondents is disagreeing, and 2 respondent is totally disagreed.





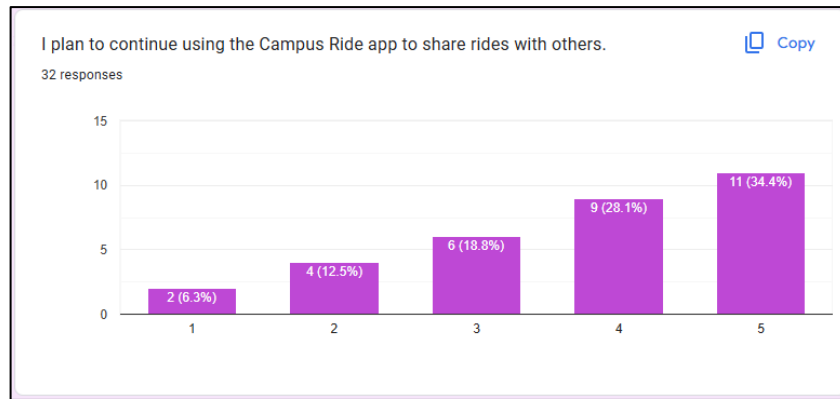
**Figure 6.20 Bar Chart of Questionnaire Question**

Figure 6.20 represents those 13 respondents strongly agree that it is desirable to learn more about using the system while 7 respondents is agreed, 7 respondents is neutral, 4 respondents is disagreeing, and 1 respondent is totally disagreed.



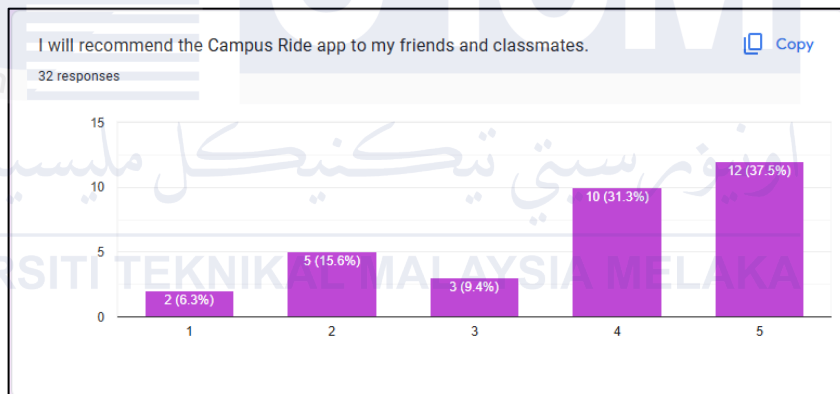
**Figure 6.21 Bar Chart of Questionnaire Question**

Figure 6.21 represents those 11 respondents strongly agree that they intend to use the system for their daily commute while 6 respondents is agreed, 6 respondents is neutral, 8 respondents is disagreeing, and 1 respondent is totally disagreed.



**Figure 6.22 Bar Chart of Questionnaire Question**

Figure 6.22 represents those 11 respondents strongly agree that they plan to continue using the system to share the rides with others while 9 respondents is agreed, 6 respondents is neutral, 4 respondents is disagreeing, and 2 respondent is totally disagreed.



**Figure 6.23 Bar Chart of Questionnaire Question**

Figure 6.23 represents those 12 respondents strongly agree that they will recommend the system with their friends and classmates while 10 respondents is agreed, 3 respondents is neutral, 5 respondents is disagreeing, and 2 respondent is totally disagreed.

## 6.7 Conclusion

In summary, this testing phase is important for making sure the software is of good quality and works well for users. It helps reduce the chance of defects reaching end-users and ensures a positive experience with the Campus Ride Application. This phase happens after the software is developed but before it is released. The main goal is to find and fix any problems or inconsistencies to make the software better and more reliable.



## **CHAPTER 7: PROJECT CONCLUSION**

### **7.1 Introduction**

This chapter wraps up the entire project by discussing its strengths, weaknesses, and areas for improvement. It also highlights opportunities for enhancement and explains how the project contributes to the target users.

### **7.2 Observation on Weakness and Strengths**

Observing the system's weaknesses and strengths is essential for understanding its capabilities. The details of these weaknesses and strengths will be further explained in the points below.

#### **7.2.1 System Strengths**

The strengths of Campus Ride include the convenience of having its own e-wallet, so users don't need to rely on FPX or debit cards, which saves time when making payments. The app also allows users to view the driver's details before confirming them as their driver. This means that if a driver has received bad reviews or feedback, users can choose to avoid them and find a new driver. Another benefit is that the app enables users to share rides with others, helping to reduce costs and save money. Additionally, before sharing a ride, users can get to know some details about the person they'll be riding with, which adds a layer of comfort. The system also offers live tracking, allowing users to monitor their location in real time.

### **7.2.2 System Weakness**

The Campus Ride application has several weaknesses. One issue is that the app needs to be connected to the university's database to make sure student data is accurate and to prevent misuse, like someone registering with another student's matric number. Another problem is that users cannot set their own drop-off location, as it is automatically set to their current location. Users also cannot make ride bookings in advance. The details about the current location are also not very user-friendly. Additionally, the ride-sharing feature only works if the drop-off and pick-up locations are the same. If the system detects that the locations are even slightly different, the feature cannot be used. Finally, there is no payment gateway in the app yet, so the e-wallet is not functional and can't be used for real payments.

### **7.3 Propositions for Improvement**

To improve the Campus Ride application, it should allow users to set their own drop-off locations instead of automatically using the current location. Adding an option to book rides in advance would make the app more convenient. The app should also show clearer location details that are easier for users to understand. The ride-sharing feature could be improved by allowing users to share rides even if the pick-up and drop-off locations are close but not the same. To prevent misuse, the app should connect better with the university's database to ensure accurate student data. Finally, a payment gateway should be added to make the e-wallet functional for real payments.

### **7.4 Project Contribution**

This system has great potential for students because it allows certified users to take advantage of cost-saving features. By using the app, students can share rides and reduce their travel expenses. This helps them save money and manage their budget more effectively. The app makes it easier for students to find affordable transportation options, making it a useful tool for cutting down on daily commuting costs.

## 7.5 Conclusion

To conclude the Campus Ride project, there are many ways to improve and refine the app in the future, which will require more time. Even so, the app has already been very beneficial for users by meeting the needs. The Software Development Life Cycle used during development was very helpful in completing the project. Campus Ride was developed over 14 weeks and met its main goals. Although it works well, its design could be improved. Making these changes will help make the system complete and more effective, but this will need additional time and effort.



## REFERENCES

*Grab Clone - Start your Business with Super App in 7 Days.* (n.d.). V3cube.  
<https://www.v3cube.com/grab-clone/>

Kaushik, V. (2024, March 31). Mastering Flutter UI: Tips for designing Intuitive user experiences. *Medium*. <https://medium.com/@kaushikvikas/mastering-flutter-ui-tips-for-designing-intuitive-user-experiences-5bc3ba0a8d3c>

Davisson, K. (2021, December 7). Adding Google Maps to Flutter - flutter - medium. *Medium*. <https://medium.com/flutter/google-maps-and-flutter-cfb330f9a245>

Mail, M. (2023, January 17). Grab Malaysia raises fare prices per minute and this is why. *Malay Mail*.  
<https://www.malaymail.com/news/malaysia/2023/01/17/grab-malaysia-raises-fare-prices-per-minute-and-this-is-why/50665#:~:text=For%20example%2C%20a%2010%20km,9am%20and%205pm%20to%208pm.>

Sabirov, R. (2023, December 28). *A guide to black box testing vs white box testing.* Qase Blog. <https://qase.io/blog/black-box-vs-white-box-testing/>

M,Fikri. (2024, August 20). *Campus Ride Demonstration* [Video]. YouTube.  
<https://youtu.be/hkrRN4gUSGQ>

## APPENDICES

## Appendix A: Demographic (User Information)

Campus Ride Application

This form was created to collect user acceptance for my application, which was constructed as a final year project. Please answer the questions below honestly and thank you!

fikrifadzil28@gmail.com [Switch account](#)

Not shared

\* Indicates required question

Gender \*

Male

Female

Age \*

19-21

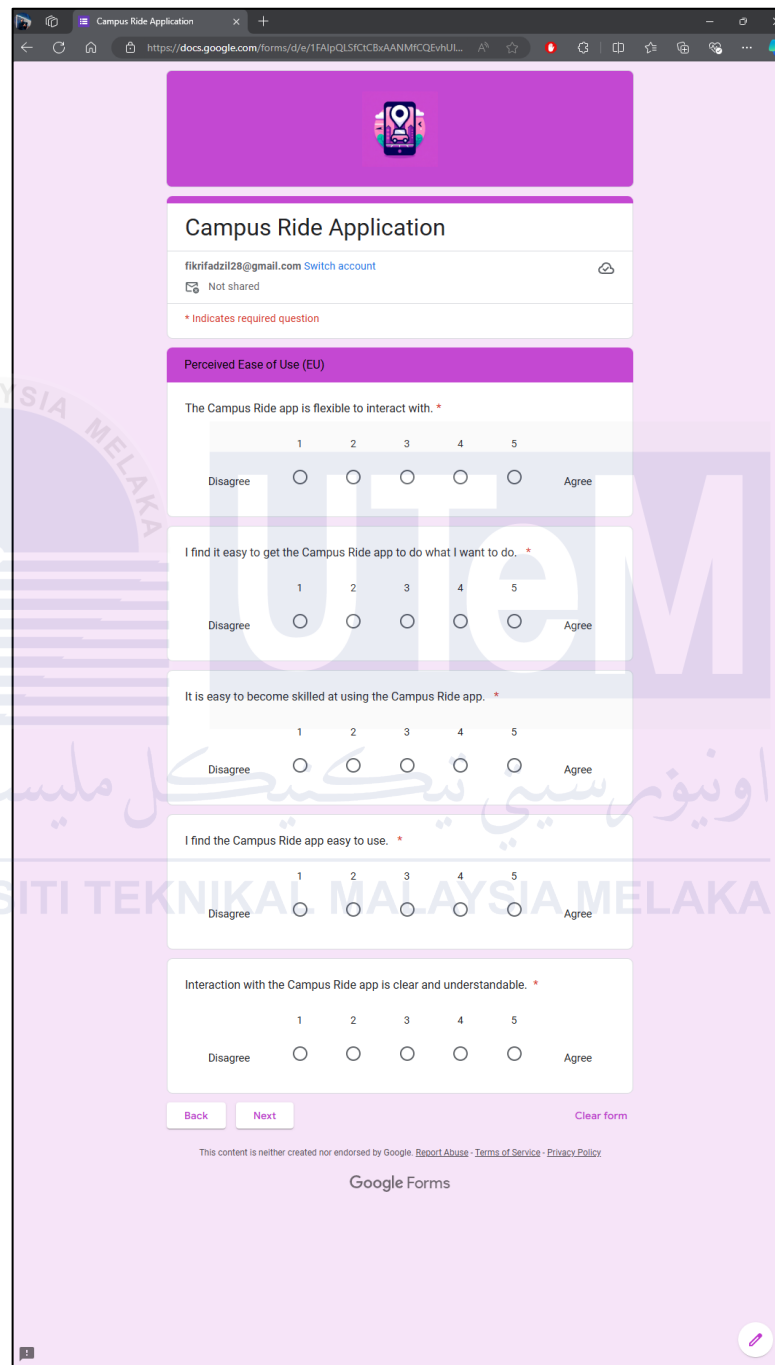
Next Clear form

This content is neither created nor endorsed by Google. [Report Abuse](#) - [Terms of Service](#) - [Privacy Policy](#)

Google Forms



## Appendix B: Demographic (Perceived Ease of Use)



Campus Ride Application

fikrifadzil28@gmail.com [Switch account](#)

Not shared

\* Indicates required question

### Perceived Ease of Use (EU)

The Campus Ride app is flexible to interact with. \*

1 2 3 4 5

Disagree      Agree

I find it easy to get the Campus Ride app to do what I want to do. \*

1 2 3 4 5

Disagree      Agree

It is easy to become skilled at using the Campus Ride app. \*

1 2 3 4 5

Disagree      Agree

I find the Campus Ride app easy to use. \*

1 2 3 4 5

Disagree      Agree

Interaction with the Campus Ride app is clear and understandable. \*

1 2 3 4 5

Disagree      Agree

[Back](#) [Next](#) [Clear form](#)

This content is neither created nor endorsed by Google. [Report Abuse](#) - [Terms of Service](#) - [Privacy Policy](#)

Google Forms

## Appendix C: Demographic (Perceived Usefulness)



The image shows a Google Form titled "Campus Ride Application" with a purple header. The form is displayed in a browser window. The form content includes a title, a header image of a smartphone with a location pin, and five Likert scale questions. Each question has a 5-point scale from "Disagree" to "Agree".

**Campus Ride Application**

fikrifadzil28@gmail.com [Switch account](#)  
Not shared

\* Indicates required question

**Perceived Usefulness (PU)**

Using the Campus Ride app enables me to easily find and share rides. \*

1 2 3 4 5  
Disagree      Agree

I find the Campus Ride app useful for my daily commute. \*

1 2 3 4 5  
Disagree      Agree

Using the Campus Ride app enhances my effectiveness in organizing my rides. \*

1 2 3 4 5  
Disagree      Agree

The Campus Ride app makes it easier to connect with other students for shared rides. \*

1 2 3 4 5  
Disagree      Agree

The Campus Ride app makes it easier to reduce travel costs and time. \*

1 2 3 4 5  
Disagree      Agree

[Back](#) [Next](#) [Clear form](#)

This content is neither created nor endorsed by Google. [Report Abuse](#) - [Terms of Service](#) - [Privacy Policy](#)

Google Forms

## Appendix D: Demographic (Capability)



The screenshot displays a Google Forms survey titled "Campus Ride Application" in a web browser. The form is set to "Not shared" and is created by fikrifadzil28@gmail.com. It contains three Likert scale questions under the heading "Capability (CP)". Each question has five response options: Disagree, 1, 2, 3, 4, 5, and Agree. The questions are:

- 1. The Campus Ride app provides clear instructions for find and offering rides. \*
- 2. Adding or searching for ride details is straightforward in the Campus Ride app. \*
- 3. The features of the Campus Ride app meet my ride-sharing needs. \*

At the bottom of the form, there are "Back" and "Next" buttons, a "Clear form" link, and a footer with the text "This content is neither created nor endorsed by Google. Report Abuse - Terms of Service - Privacy Policy" and the "Google Forms" logo. The background of the page features a watermark of the Universiti Teknikal Malaysia Melaka logo and name in both English and Malay.

## Appendix E: Demographic (Trustworthiness)

The screenshot shows a Google Form titled "Campus Ride Application" in a browser window. The form is set to "Not shared" and is created by "fikrifadzil28@gmail.com". It contains three Likert scale questions under the heading "Trustworthiness (TW)".

**Question 1:** I trust the Campus Ride app to keep my ride details secure. \*

1 2 3 4 5

Disagree      Agree

**Question 2:** The Campus Ride app provides security for my personal information. \*

1 2 3 4 5

Disagree      Agree

**Question 3:** I feel safe sharing my ride information using the Campus Ride app. \*

1 2 3 4 5

Disagree      Agree

Navigation buttons: Back, Next, Clear form.

Footer: This content is neither created nor endorsed by Google. Report Abuse - Terms of Service - Privacy Policy. Google Forms.

## Appendix F: Demographic (Attitude)

The screenshot displays a Google Forms survey titled "Campus Ride Application". The form is set to "Not shared" and is accessed via a browser. The survey is titled "Attitude (ATT)" and contains three Likert scale questions. Each question has five response options: 1 (Disagree), 2, 3, 4, 5 (Agree). The questions are:

- 1. I enjoy using the Campus Ride app. \*
- 2. It is convenient for me to use the Campus Ride app. \*
- 3. I find it desirable to learn more about using the Campus Ride app. \*

At the bottom of the form, there are "Back" and "Next" buttons, and a "Clear form" link. A footer note states: "This content is neither created nor endorsed by Google. [Report Abuse](#) - [Terms of Service](#) - [Privacy Policy](#)". The Google Forms logo is also visible.

UNIVERSITI TEKNIKAL MALAYSIA MELAKA

## Appendix G: Demographic (Intention to Use)



The image shows a screenshot of a Google Forms survey titled "Campus Ride Application". The form is displayed in a browser window with the URL <https://docs.google.com/forms/d/e/1FAIpQLSfCfC8xkAANMfCQEvHUL...>. The form header includes the title "Campus Ride Application", the creator's email "fikrifadzil28@gmail.com", and a "Switch account" link. Below the header, there is a section titled "Intention to Use (IU)" containing three Likert scale questions. Each question has five radio button options labeled "Disagree", "1", "2", "3", "4", "5", and "Agree". The questions are:

- "I intend to use the Campus Ride app for my daily commute. \*
- "I plan to continue using the Campus Ride app to share rides with others. \*
- "I will recommend the Campus Ride app to my friends and classmates. \*

At the bottom of the form, there are buttons for "Back", "Submit", and "Clear form". Below the buttons, there is a disclaimer: "This content is neither created nor endorsed by Google. [Report Abuse](#) - [Terms of Service](#) - [Privacy Policy](#)". The Google Forms logo is visible at the bottom center. The background of the form is a light purple color with a large, faint watermark of the Universiti Teknikal Malaysia Melaka logo and name in both English and Malay.



اونيورسيتي تيكنيكل مليسيا ملاك

UNIVERSITI TEKNIKAL MALAYSIA MELAKA