

**HOUSEMATEHEAVEN : HOUSEMATE
FINDER MOBILE APPLICATION FOR
UNIVERSITY STUDENT**

**DANISH FARHAN BIN MOHD RUSDI
52213120519**

**UNIVERSITY KUALA LUMPUR
FEBRUARY 2023**

**HOUSEMATEHEAVEN : HOUSEMATE
FINDER MOBILEAPPLICATION FOR
UNIVERSITY STUDENT**

**DANISH FARHAN BIN MOHD RUSDI
52213120519**

**Report Submitted to Fulfil the Partial Requirements
for the Bachelor of Information Technology (Hons) In
Software Engineering
University Kuala Lumpur**

FEBRUARY 2023

DECLARATION

I declared this report is my original work and all references have been cited accordingly as required by the University.

Signature : 
Name : Danish Farhan Bin Mohd Rusdi
Student ID : 52213120519
Date : 11/6/2023

APPROVAL PAGE

We have supervised and examined this report and verify that it meets the program and University's requirements for the Bachelor of Software Engineering

Signature : 
Supervisor : Ts. Dr. Husna Sarirah Husin
Date : 16 June 2023
Official Stamp : **Ts. Dr. HUSNA SARIRAH HUSIN**
Head of Research & Innovation Section
Universiti Kuala Lumpur
Malaysian Institute of Information Technology (UniKL MIIT)

Signature :
Assessor : Dr. Norhaidah Abu Haris
Date :
Official Stamp

COPYRIGHT PAGE

Declaration of Copyright and Affirmation of Fair Use of
Unpublished Research Work as stated below:

Copyright@ 11/6/2023 by DANISH FARHAN BIN MOHD
RUSDI(52213120519)

All rights reserved for HOUSEMATEHEAVEN :
HOUSEMATE FINDERMOBILE
APPLICATION FOR UNIVERSITY
STUDENT

No part of this unpublished research may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise without the prior written permission of the copyright holder except as provided below:

- i. Any material contained in or derived from this unpublished research may only be used by other in their writing with due acknowledgement.
- ii. MIIT UniKL or its library will have the right to make and transmit copies (print or electronic) for institutional and academic purposes.
- iii. The MIIT UniKL's library will have the right to make, store in a retrieval system and supply copies of this unpublished research if requested by other universities and research libraries.

DEDICATION

I would like to dedicate this Final Year Project report to everyone in the field of Software Engineering who want to start the project in developing mobile application in HousemateHeaven : Housemate Finder Mobile Application For University Student. Dedication also goes to my family members and supervisor, Ts. Dr. Husna Sarirah Husin. Last but not least, thank you to my friends and everyone that involved and contribute throughout completing this project in both directly and indirectly. Thank you so much for helping me to finish my final year project.

ACKNOWLEDGEMENT

All thanks are due to Almighty Allah, the compassionate and merciful, who knows about everything in the universe, hidden and unhidden, and has grant me a little of knowledge from all knowledge that available in the universe.

I have try my best in completing this final year project. However, it would not have been possible without the various support and help of many individuals. I would like to thanks all of them. I'm truly highly indebted to Husna Sarirah Husin for the guidance and constant supervision throughtout my final year project journey.

I would like to express my gratitude towards my parents and friends fortheir encouragement which help me mentally in completing this project. My thanks and appreciations also go to the people who have willingly helped me out with their abilities.

TABLE OF CONTENTS

DECLARATION	II
APPROVAL PAGE	III
COPYRIGHT PAGE	IV
DEDICATION	V
ACKNOWLEDGEMENT	VI
LIST OF ABBREVIATION	X
ABSTRACT	XI
CHAPTER 1 : INTRODUCTION	1
1.1 PROJECT INTRODUCTION	1
1.2 PROJECT OBJECTIVE	1
1.3 PROJECT PROBLEM STATEMENT	2
1.4 PROJECT SCOPE	2
1.5 PROJECT EXPECTED OUTCOME	4
CHAPTER 2 : LITERATURE REVIEW	5
2.1 INTRODUCTION	5
2.2 FACTORS IN CHOOSING HOUSEMATE	5
2.3 EFFECTS ON INTERACTIONS AND RELATIONSHIPS BETWEEN HOUSEMATES	6
2.4 USABILITY FACTORS IN MOBILE APPLICATION	7
2.5 DEVELOPMENT OF MOBILE APPLICATION	9
2.6 OVERVIEW SIMILAR APPLICATION	9
2.8 CONCLUSION	11
CHAPTER 3 : RESEARCH METHODOLOGY	12
3.1 PROJECT COST	12
3.2 INTRODUCTION	12
3.3 ITERATIVE AND INCREMENTAL	12
3.4 WORD BREAKDOWN SYSTEM(WBS)	14
3.5 HARDWARE AND SOFTWARE REQUIREMENT	14
3.6 GANTT CHART	16
3.7 FUNCTIONAL REQUIREMENTS	16
3.8 NON-FUNCTIONAL REQUIREMENT	24
CHAPTER 4 : PROTOTYPE DEVELOPMENT	26
4.1 INTRODUCTION	26
4.2 PROTOYPE DEVELOPMENT	26
4.3 APPLICATION INTERFACE	30
4.4 CONCLUSION	40
CHAPTER 5 : TESTING AND RESULTS	41
5.1 INTRODUCTION	41
5.2 TESTING APPROACH	41
5.3 TESTING PREPARATION	41
5.4 TESTING SCHEDULE	42
5.5 TEST CASES	43
5.6 TEST RESULTS	68
5.7 TEST SUMMARY	69
5.8 CONCLUSION	69
CHAPTER 6 : CONCLUSION	70
6.1 CONCLUSION	70
6.3 LIMITATIONS	71
6.2 RECOMMENDATIONS	71
REFERENCE	73
APPENDIX A : GANTT CHART	75
APPENDIX B : FIREBASE-FIRESTORE	76
APPENDIX C : CODING	77

LIST OF TABLES

DECLARATION	II
APPROVAL PAGE	III
COPYRIGHT PAGE	IV
DEDICATION	V
LIST OF ABBREVIATION	X
ABSTRACT	XI
CHAPTER 1 : INTRODUCTION	1
CHAPTER 2 : LITERATURE REVIEW	5
Table 1 : Project Comparison	11
CHAPTER 3 : RESEARCH METHODOLOGY	12
Table 2 : Software Requirement	14
Table 3 : Hardware Requirement	15
Table 4 : Project Cost	15
Table 5 : Functional Requirement	23
Table 6 : Non-Functional Requirement	24
CHAPTER 4 : PROTOTYPE DEVELOPMENT	26
Table 7 : Personal Laptop Specification	27
Table 8 : Personal Mobile Phone Specification	28
CHAPTER 5 : TESTING AND RESULTS	41
Table 9 : Hardware Preparation	42
Table 10 : Testing Schedule	42
Table 11 : Test Case Statement - Login	43
Table 12 : Test Case Statement - Search Profile	44
Table 13: Test Case Statement - Search Form	45
Table 14: Test Case Statement - Edit Profile	46
Table 15: Test Case Statement - Delete Profile	47
Table 16: Test Case Statement - Edit Form	48
Table 17: Test Case Statement - Delete Form	49
Table 18 : Test Case Statement - Logout	50
Table 19 : Test Case Statement - Register	52
Table 20 : Test Case Statement - Login	53
Table 21 : Test Case Statement - Choose Role	54
Table 22 : Test Case Statement - View Tutorial	55
Table 23 : Test Case Statement - Create Form	57
Table 24 : Test Case Statement - Edit Form	57
Table 25 : Test Case Statement - Delete Form	58
Table 26 : Test Case Statement - View form	59
Table 27 : Test Case Statement - Bookmark Form	60
Table 28: Test Case Statement - Logout	61
Table 29 : Test Case Statement - Search Form	62
Table 30 : Test Case Statement - View Bookmark	63
Table 31 : Test Case Statement - View Profile	64
Table 32 : Test Case Statement - Edit Profile	65
Table 33: Test Case Statement - Logout	66
Table 34: Test Case Statement - Forgot Password	67
Table 35 : Test Results	68
Table 36 : Test Summary	69
CHAPTER 6 : CONCLUSION	70
REFERENCE	73
APPENDIX A : GANTT CHART	75
APPENDIX B : FIREBASE-FIRESTORE	76
APPENDIX C : CODING	77

LIST OF FIGURES

DECLARATION	II
APPROVAL PAGE	III
COPYRIGHT PAGE	IV
DEDICATION	V
LIST OF ABBREVIATION	X
ABSTRACT	XI
CHAPTER 1 : INTRODUCTION	1
CHAPTER 2 : LITERATURE REVIEW	5
Figure 1 : Flatmates	9
Figure 2 : Roomster	10
CHAPTER 3 : RESEARCH METHODOLOGY	12
Figure 3 : Iterative and Incremental Model	13
Figure 4 : WBS	14
CHAPTER 4 : PROTOTYPE DEVELOPMENT	26
Figure 5 : Iterative and Incremental Model	26
Figure 6 : Use Case Diagram	29
Figure 7 : ERD Diagram	30
Figure 8 : Login Page	30
Figure 9 : Forgot Password Page	31
Figure 10 : Register Page	31
Figure 11 : Role Page	32
Figure 12 : Homepage	32
Figure 13 : Shootout Page	33
Figure 14 : Shootout Page Version 2	33
Figure 15 : Rental Form	34
Figure 16 : Edit Post Page	34
Figure 17 : Pop-up Notification Version 1	35
Figure 18 ; Favourite Page	35
Figure 19 : Profile Page	36
Figure 20 : Search Page	36
Figure 21 : Adjust Page - Post	37
Figure 22 : Adjust Page - User	37
Figure 23 : Adjust Page - User Result	38
Figure 24 - Adjust Page - Post Result	38
Figure 25 : Edit Post Page for Admin	39
Figure 26 : Edit User Page for Admin	39
CHAPTER 5 : TESTING AND RESULTS	41
CHAPTER 6 : CONCLUSION	70
REFERENCE	73
APPENDIX A : GANTT CHART	75
APPENDIX B : FIREBASE-FIRESTORE	76
APPENDIX C : CODING	77

LIST OF ABBREVIATION

University Kuala Lumpur	UNIKL
Malaysian Institute of Information Technology	MIIT
Word Breakdown System	WBS
Entity Relationship Diagram	ERD
Test Case	TC

ABSTRACT

The study is about to design a android-based mobile application. The mobile application will be called HousemateHeaven. HousemateHeaven mobile application main function is to help university student in finding housemate during their university years. The application will be beneficial to university student in finding housemate easily by using a user-friendly mobile application.

CHAPTER 1 : INTRODUCTION

1.1 PROJECT INTRODUCTION

Housemate definition is a person who lives in the same house with another (Housemate Definition & Meaning, 2022). Every year, there are always a complaint from university student about how hard is it to find housemate to rent a house around their university area. That is why HousemateHeaven been developed.

HousemateHeaven is intended to be a mobile application that will be an ease to university student in finding housemate or roommate during their university years. This application will be using Flutter as a backbone of the project. Student can expect that this application will be a good application for them in solving their finding in searching for person to share house with.

1.2 PROJECT OBJECTIVE

- I. To study the requirements in developing a mobile application for findinghousemate, which is named as HousemateHeaven mobile application.
- II.To design a good functional application with a nice looking interface.
- III.To develop the HousemateHeaven application using iterative andincremental model in SDLC methodology

1.3 PROJECT PROBLEM STATEMENT

1. Problem in Finding Housemate

- A lot of student faced a same problem in finding housemate nowadays. Although they have been sharing their search in social media, it still been ignored because the posting did not reach the target audience. Using the old way of sharing flyer also did not work anymore.

2. Difficulty to find similar application that focus in Malaysia

- After researching for a while, a housemate finder application that focusing in Malaysia is quite hard to find. The application mostly focus in am Europe Country like France, England and Germany. Student that studying in Malaysia will have a hard time to use this application because most user in other application mostly originate from outside of Malaysia.

3. Rental is high in certain place

- In certain places, rental of house can be unexpectedly high due to how develop the town is. For example, range of rental around Kuala Lumpur is around RM 1 300 - RM 1 800. This amount is too high if student want to rent the house.

1.4 PROJECT SCOPE

Module Admin :

I. Module Login

- Admin need to login into HousemateHeaven application by entering email and password.

II. Module Adjust

- Admin can edit or delete any student that register in the HousemateHeaven application.
- Admin can edit or delete any shootout form that violate term and condition of the HousemateHeaven application.

IV. Module Search

- Admin can search any student profile
- Admin can search any shootout form

V. Module Logout

- Admin can logout from the application by clicking logout button.

Module Student :

I. Module Register

- Student must sign up to access the system as a new user. The student needs to enter information like full name, username, email, password, university name and student id.

I. Module Login

- Student need to login into application by entering email and password.

II. Module Shootout

- Student can create new form
- Student can edit their form
- Student can view others form
- Student can favourite form

III. Module Profile

- Student can view their profile
- Student can edit their profile.

V. Module Search

- Student can search for a form by using search button.

IV. Module Favourite

- Student can view their favourite form

VII. Module Logout

- Student can logout from the application by clicking on the logout button

VII Module Forgot Password

- Student can reset their password by entering their email

1.5 PROJECT EXPECTED OUTCOME

I. Student and Admin able to register in the application by entering their information.

II. Student and Admin able to login in application by entering their username and password.

III. Admin are able to edit and delete any recruitment posting.

IV. Student are able to post their recruitment in housemate requirement page.

V. Student are able to rate their housemate in the application.

VI. Student are able to edit their profile.

VII. Admin and Student are able to use search button by filter the category.

VIII. Student are able to view other student rating.

CHAPTER 2 : LITERATURE REVIEW

2.1 INTRODUCTION

In this chapter, literature review about this project will be discussed. All information that will be collected will be used as references to do the project based on research that will be conduct. In this literature review, all information that been got by researching journal of the research paper from researchers and information taken from website will be discussed and hopefully it will help to clear all the problems that have been occur in the research before the development of the project started.

2.2 FACTORS IN CHOOSING HOUSEMATE

In choosing housemate, there have been many studies done to investigate the factors or criteria in choosing housemate. According to Clark and Keith Tuffin (2015), the factos of choosing a housemate are based on age, gender and ethnicity. This is supported by research that had been done by Ahrentzen (2003), that said that one of the factor in choosing housemate is related to age range. The atmosphere in the house will be different for each group of age. Sometimes, younger people tends to be a housemate with a group of people around their age then with older people because of things like conversation topic of younger generation mostly about current trends in social media. The same thing can be said for older generation because that generation will likely talk about something related to current issue in the

world like politic and economic.

The next criteria for choosing a housemate may be due to an economic factor, as stated by Kenyon and Health (2001). It is because there is big differences in rental around rural areas and city areas. The amount of rental fees in city area tends to be higher than rural area because of how development progress in city like public transportation, tall buildings and shopping malls growing faster.

Furthermore, in the research by Despres (1994), the need of housemate to fill the family role to other housemate is also been included among the factors. Housemate needs to be treated like a family because housemate live under same roof like how a family does. A person who learning at university that has long distance from their house will feel lonely because of absence of their family around them. The feeling will be overcome when they have an interaction among their housemate like what they used to with their family.

2.3 EFFECTS ON INTERACTIONS AND RELATIONSHIPS BETWEEN HOUSEMATES

Based on previous studies, there are many effects on interactions and relationships between housemates. One of it is research from Ozturk and Mutlu (2010), social interactions are importance for humans and their happiness. When a person interact with each other, they will feel happy because it fills the void of loneliness in them because there is someone they

can share things with. This research is supported by a research by Lizzy Onvlee (2020) that said that social interaction between housemates has an impact on the happiness of student. Student is also a human, so they need glimpse of happiness in their life too. When we are in a happy mood, everything will seems run so smoothly. Other than that, a research by Walsh (2015) said that social interaction can be seen as important source of emotional and practical support to a person. If a person has a problem with something, they need another person to support them emotionally by having conversation with them. This situation can happen between housemates when one of them is having problem with something. That is why according to research by Chatman, Broaddus and Spevack (2019), social interactions are more important than housing characteristics because of how social interactions can have many effects on someone. In this case, it it important for housemates to always have social interactions between each one of them.

2.4 USABILITY FACTORS IN MOBILE APPLICATION

When we talk about usability factors in mobile application, it related to how user feels about how well the application can be an user-friendly application. According to research by Harrison, Flood and Duce (2013), there are three factors that can affect usability of mobile application which are user, task and context of use. These factors are proven as a critical factors to the usability of an application as supported by a researched by Nielson (1994). The firstfactor is user. The end user need to be put as priority in the development of an application because in the end of the

day, they will use the system. In the view from mobile application development, user experience will be based on how well interface will be because of how small mobile application compared to the website. This is what based on research done by Hoehle and Venkatesh (2015) that said most of the time user interface is been sacrificed for unnecessary amount of information that want to be display. Next factor is task. It refers to goal of the user in the mobile application. For example, user might have download the mobile application for entertainment, functionality utility or communication based on research by Kang (2014). The user goals cannot be achieved if the application is to complex for them like there are too many additional features that been added to application.

Last but not least, context of use is one of the important factors of usability in mobile application. This factor also can be said as what environment does the user will use the application. According to research by Venkatesh, Morris, Davis and Davis (2003), user will likely adapt new technology when it offers expected outcome. This can means that mobile application will be used more when it offers what user want in the situation. For example, public transportation mobile application will be used more when user want to know the exact time the public bus arrived. Mobile application also can be accessed anytime like in the middle of walking to the office. This is supported of a research by Schildbach and Rukzio (2010) that stated, using mobile application while walking can slow down walker's average speed. The user walking's speed is affected because of how focus user on their mobile application during that time.

2.5 DEVELOPMENT OF MOBILE APPLICATION

A)Flutter

Flutter is open source UI software development kit that been used to develop cross-platform applications for Android, IOS, Linux, macOS, Windows, Google Fuchsia and the web from a single code base. It is created by Google and released in 2017. Flutter is widely used for mobile application development around the world.

B)Firebase

Firebase provides detailed documentation and cross-platform SDKs to help you build and ship apps on Android, iOS, the web, C++, and Unity. Firebase is a database that commonly used for mobile application and web development because it is a real time database.

2.6 OVERVIEW SIMILAR APPLICATION

A)Flatmates

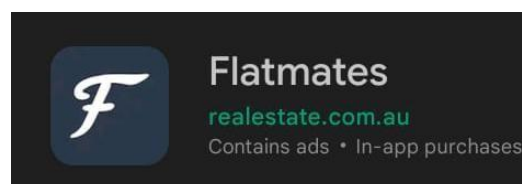


Figure 1 : Flatmates

Flatmates is a mobile application that can be used to search for a roommate. This application is focusing on people that lives in Australia. Every year, this application done a survey to see what is happening in the world of share housing.

B)Roomster

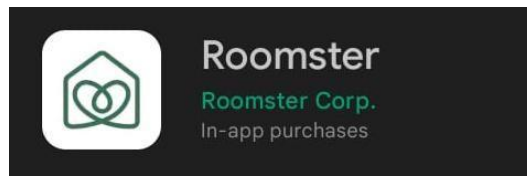


Figure 2 : Roomster

Roomster is a mobile application that can be used to search for a housemate. This application is available in 192 countries and has 18 languages option but mostly focus on famous city around the world like Paris and London. People can search for room with a pet friendly environment in the search filter function.

2.7 PROJECT COMPARISION

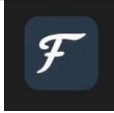
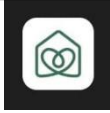

	Flatmates	Roomster	HousemateHeaven
Logo			
Platform	Mobile and Web	Mobile	Mobile
Function	Find roommate	Find roommate	Find housemate
Country	Australia	192 countries	Malaysia
Speciality	Have a survey every year	Able to search for a room that pet friendly	Focusing on Malaysia's University Student

Table 1 : Project Comparison

2.8 CONCLUSION

This chapter reviews the factors and criteria in choosing housemates which are the age, gender and economic factors. In addition, literature review was done on effects of interactions and relationships of housemates. On another note, usability factors in mobile application are discussed and descriptions of the list of software to develop mobile application are presented along with comparison between existing mobile application.

CHAPTER 3 : RESEARCH METHODOLOGY

3.1 PROJECT COST

The budget for this project is estimated to be RM 0. It is because all the hardware and software that being used is free.

3.2 INTRODUCTION

This chapter will consist of understanding of the project planning and requirement specification of the project. This is because of understanding new mechanism is difficult and it requires high critical thinking skills. Proper understanding about the application will lead to easy resolution to any problem occur.

3.3 ITERATIVE AND INCREMENTAL

Methodology Model that being used in HousemateHeaven application is Iterative and Incremental Model. It is a model that is an incremental model that is developed in multiple cycles of iterations (Iterative and Incremental Development (IID), 2020). This model is like waterfall model but unlike waterfall model, it will cycle through the phase several times.

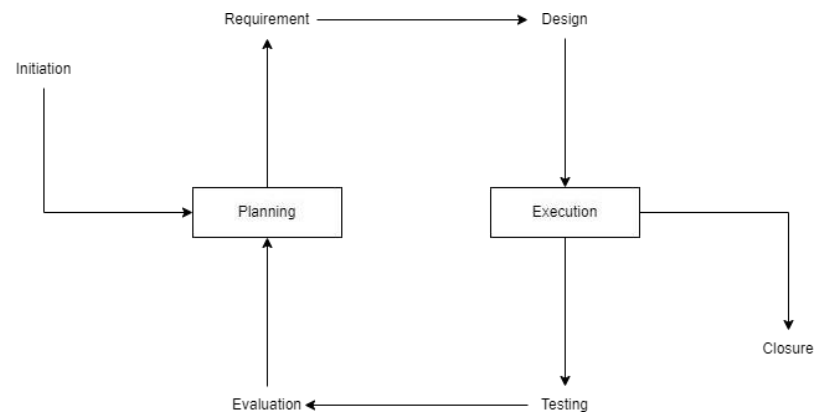


Figure 3 : Iterative and Incremental Model

The application will be using this model because requirement changes can be done through out the development stage. This will make it easier to change the functionality and improve the application. Customers can provide feedback in each increment during the development. It will make the final product will reach the expectation in customer's eyes

3.4 WORD BREAKDOWN SYSTEM(WBS)

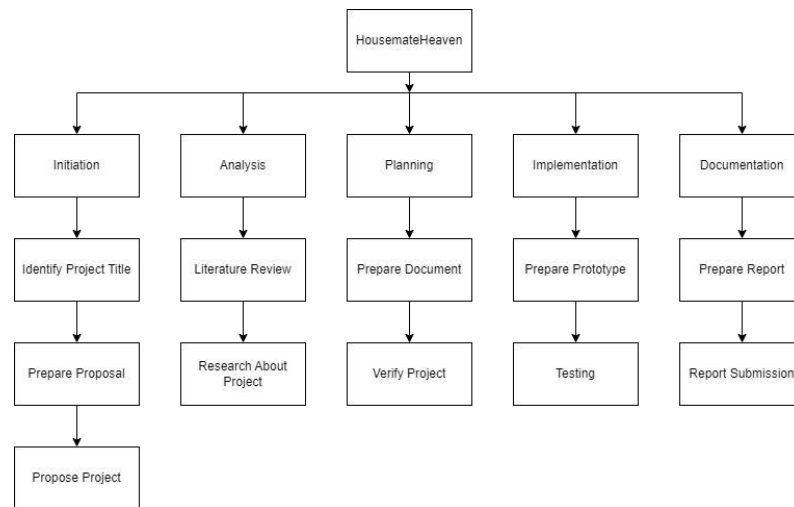


Figure 4 : WBS

3.5 HARDWARE AND SOFTWARE REQUIREMENT

A)Software Requirement

NO.	ITEMS
1	ANDROID STUDIO
2	FLUTTER
3	VISUAL STUDIO CODE
4	FIREBASE-FIRESTORE

Table 2 : Software Requirement

B)Hardware Requirement

NO.	ITEM
1	LAPTOP
2	MOBILE PHONE

Table 3 : Hardware Requirement

3.5 PROJECT COST

NO.	ITEMS	COST(RM)	TOTAL(RM)
1	HARWARE		
1.1	LAPTOP	SELF-OWNED	-
1.2	MOBILE PHONE	SELF-OWNED	-
2	SOFTWARE		
2.1	ANDROID STUDIO	-	-
2.3	VISUAL STUDIO CODE	-	-
2.4	FLUTTER	-	-
2.5	FIREBASE	-	-

Table 4 : Project Cost

3.6 GANTT CHART

Refer to Appendix A.

3.7 FUNCTIONAL REQUIREMENTS

Req. ID	Desc.	Test Case	Release	Priority	Status
RQ-1	HousemateHaven mobile application shall be able to allow student register to the application.	TC_009	1.0.0	High	Developed
RQ-2	HousemateHaven mobile application shall be able to allow student to login to application.	TC_010	1.0.0	High	Developed
RQ-3	Student that use HousemateHaven mobile application shall be able to pick between the roles given.	TC_011		High	Not Developed

RQ-4	Upon login to HousemateHaven mobile application, student shall be greeted with tutorial on how to use the application in homepage	TC_012	1.00	Medium	Developed
RQ-5	In shootout section of HousemateHaven mobile application, student shall be able to create form upon choosing "Want to Find Housemate" role.	TC_013	1.0.0	High	Developed
RQ-6	In shootout section of HousemateHaven mobile application, student shall be able to edit form upon choosing "Want to Find	TC_014	1.00	High	Not Developed

	Housemate “ role.				
RQ-7	In shootout section of HousemateHe aven mobile application, student shall be able to delete form upon choosing “ Want to Find Housemate “ role.	TC_015		High	Not Developed
RQ-8	In shootout section of HousemateHe aven mobile application, student shall be able to see the form that has been created form upon choosing “ Want to Find Housemate “ role.	TC_016		High	Not Developed

RQ-9	In shootout section of HousemateHeaven mobile application, student shall be able to see form that has been created by other students upon choosing “ Want to Find Rental House “ role.	TC_017		High	Not Developed
RQ-10	In shootout section of HousemateHeaven mobile application, student shall be able to bookmark any form upon choosing “ Want to Find Rental House “ role.	TC_018		High	Not Developed
RQ-	In search section of	TC_019		High	Not

11	HousemateHe aven mobile application, student shall be able to search form from other students upon choosing any role.				Developed
RQ- 12	In search section of HousemateHe aven mobile application, student shall be able to search other students' profile upon choosing any role.	TC_020		High	Not Developed
RQ- 13	In profile section of HousemateHe aven mobile application, student shall be able to view their profile	TC_021	1.00	High	Developed
RQ- 14	In profile section of HousemateHe	TC_022		High	Not Developed

	aven mobile application, student shall be able to edit their profile				
RQ-15	Student shall be able to logout from HousemateHeaven mobile application upon login to the application	TC_023	1.00	High	Developed
RQ-16	HousemateHeaven mobile application shall be able to allow admin to login to application by entering information that has been set.	TC_001		High	Not Developed
RQ-17	In search section of HousemateHeaven mobile application, admin shall be able to search form	TC_003		High	Not Developed

	that has been created by student				
RQ-18	In search section of HousemateHeaven mobile application, admin shall be able to search student's profile	TC_002		High	Not Developed
RQ-19	In search section of Housemate Heaven mobile application, admin shall be able to edit student profile	TC_004		High	Not Developed
RQ-20	In search section of HousemateHeaven mobile application, admin shall be able to delete student's profile upon	TC_005		High	Not Developed

	searching it.				
RQ-21	In search section of HousemateHeaven mobile application, admin shall be able to edit form t	TC_006		High	Not Developed
RQ-22	In search section of HousemateHeaven mobile application, admin shall be able to delete form	TC_007		High	Not Developed
RQ-23	Admin shall be able to logout from HousemateHeaven mobile application upon login to application	TC_008		High	Not Developed

Table 5 : Functional Requirement

3.8 NON-FUNCTIONAL REQUIREMENT

Number	Description
QR-1	The user interface of the HousemateHeaven Mobile Applicationshall be neat to be seen by user.
QR-2	The loading between section to another section inHousemateHeaven Mobile Application shall take 1 second at most.
QR-3	When the user updated their information, the HousemateHeavenMobile Appication database shall be updated.
QR-4	The user information like password shall be protected by the system in HousemateHeaven Mobile Application.
QR-5	HousemateHeaven mobile application shall be easy to use by newregister user
QR-6	HousemateHeaven mobile application server shall be stable most of the time
QR-7	All HousemateHeaven mobile application function shall be able to use without constraint
QR-8	Data that has been recorded in HousemateHeaven mobile application shall be consistent at anytime.

Table 6 : Non-Functional Requirement

3.9 CONSTRAINTS

- I. Other works might disturb project planning process.
- II. The project will be not submitted in time.
- III. There will be a chance when platform that be used will not work.
- IV. Available in Android only.
- V. Loading time between page will take a long time to switch.

3.10 CONCLUSION

In conclusion, this chapter summarized how the project will be implemented. All of the project model, platform, hardware and software have been listed and the budget has been shown.

CHAPTER 4 : PROTOTYPE DEVELOPMENT

4.1 INTRODUCTION

In this chapter, methodology that been used to refer in development of the project is examined. It also include a use case to show how the project will work. This chapter also will specify the software and hardware that been used for prototype development. In the last part of this chapter, system interface of the prototype will be shown.

4.2 PROTOTYPE DEVELOPMENT

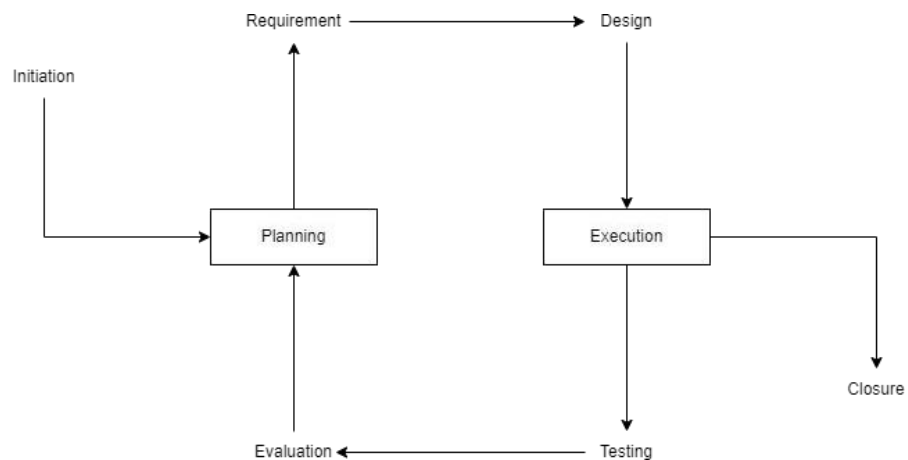


Figure 5 : Iterative and Incremental Model

4.2.1 Product Requirement

4.2.1.1 Hardware Requirement

A) Personal Laptop

Personal laptop being used as a tool to develop the prototype of application. It can easily be used in variety of location.

	Specification
Operating System	Windows 11
Browser	Google Chrome, Microsoft Edge
Random Access Memory (RAM)	16GB or above
Central Processing Unit (CPU)	Intel Core i5

Table 7 : Personal Laptop Specification

B) Personal Mobile Phone

Personal mobile phone is a portable device that can be easily used and carried because of its small size. It is a tool that has been used to test the prototype of application.

	Specification
Platform	Android 12
Random Access Memory (RAM)	8GB + 6GB

Network	5G
----------------	----

Table 8 : Personal Mobile Phone Specification

4.2.1.2 Software Requirement

- A) Android Studio**
- B) Visual Studio Code**
- C) Flutter**
- D) Dart Language**
- E) Firebase-Firestore**

4.2.2 Functional Requirement

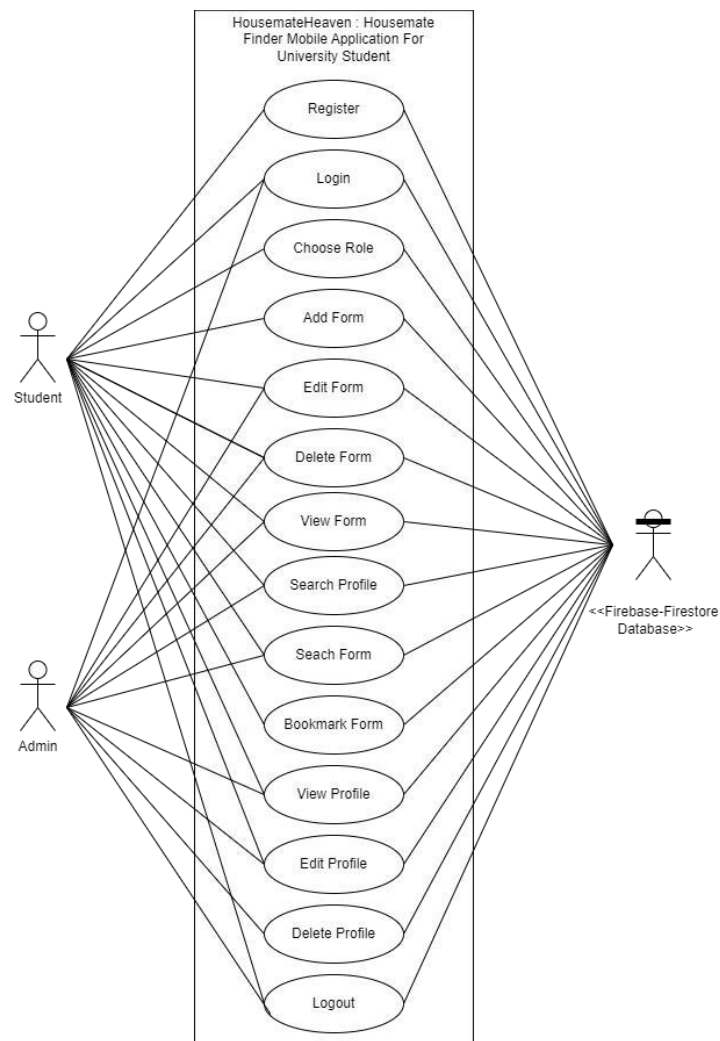


Figure 6 : Use Case Diagram

4.2.3 Database Design

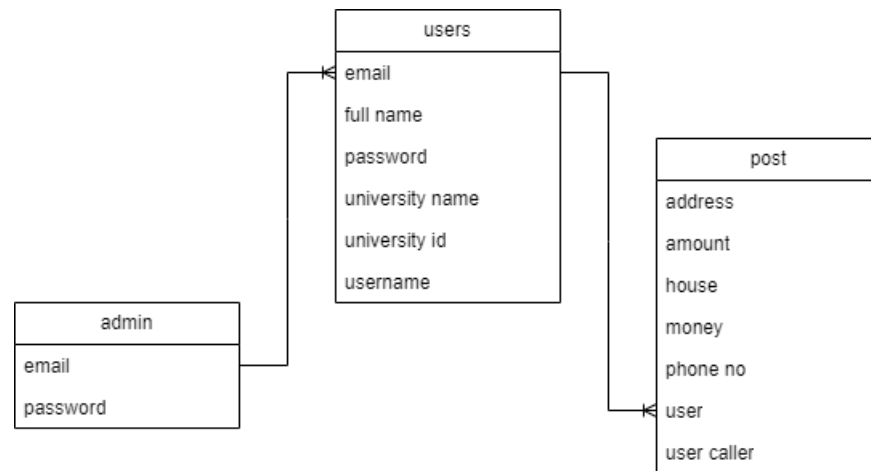


Figure 7 : ERD Diagram

4.3 APPLICATION INTERFACE

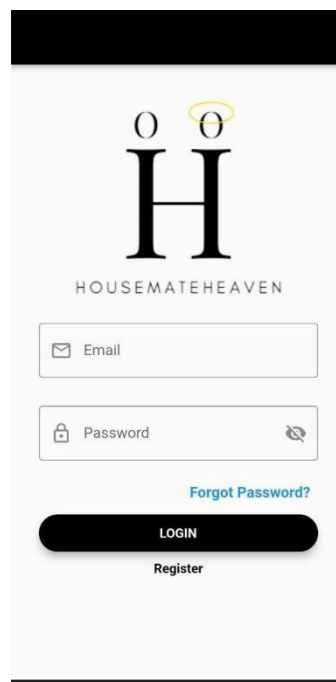


Figure 8 : Login Page

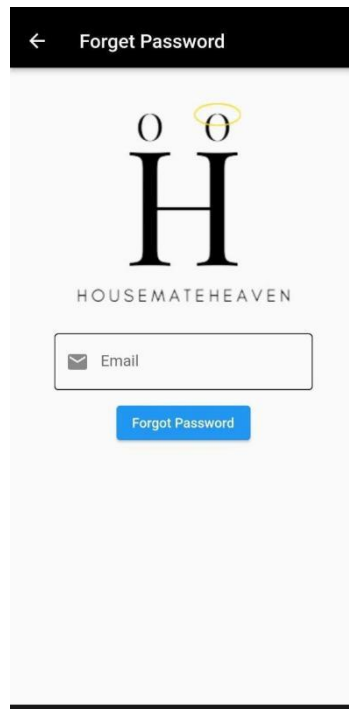


Figure 9 : Forget Password Page

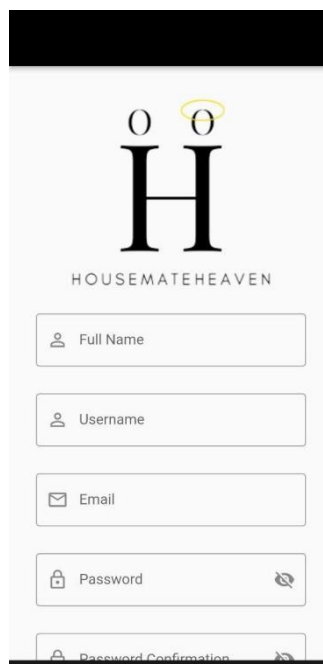


Figure 10 : Register Page



Figure 11 : Role Page



Figure 12 : Homepage

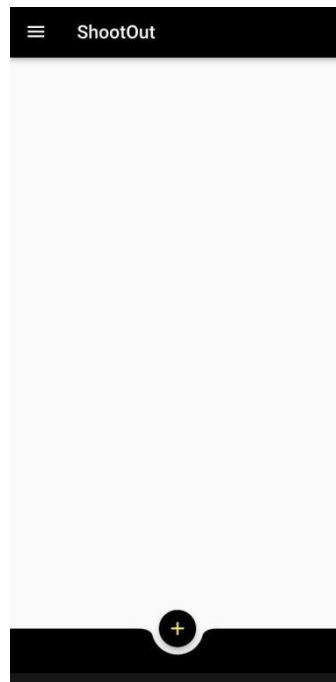


Figure 13 : Shootout Page

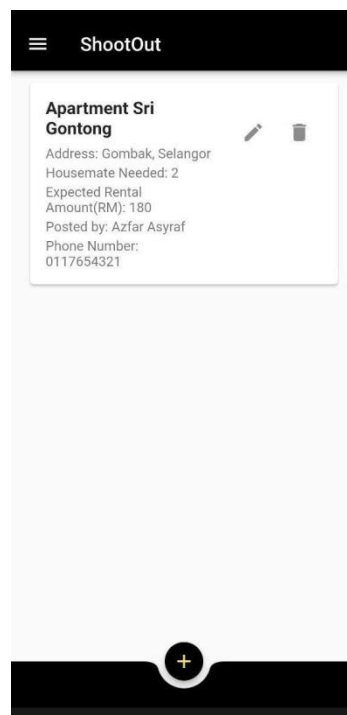
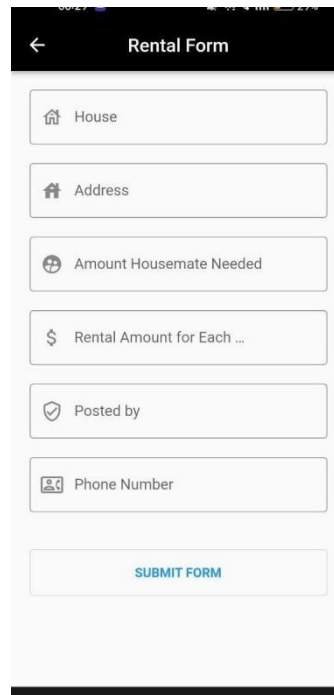
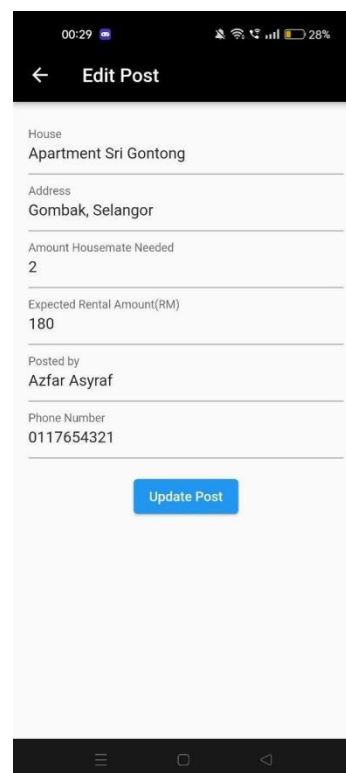


Figure 14 : Shootout Page Version 2



A screenshot of a mobile application's 'Rental Form' page. The page has a black header with a back arrow and the title 'Rental Form'. Below the header, there are six input fields, each with a small icon and a label: a house icon for 'House', a location pin for 'Address', a plus sign in a circle for 'Amount Housemate Needed', a dollar sign for 'Rental Amount for Each ...', a shield for 'Posted by', and a phone icon for 'Phone Number'. At the bottom of the form is a blue button labeled 'SUBMIT FORM'.

Figure 15 : Rental Form



A screenshot of a mobile application's 'Edit Post' page. The page has a black header with a back arrow and the title 'Edit Post'. Below the header, there are six input fields, each with a small icon and a label: a house icon for 'House', a location pin for 'Address', a plus sign in a circle for 'Amount Housemate Needed', a dollar sign for 'Expected Rental Amount(RM)', a shield for 'Posted by', and a phone icon for 'Phone Number'. The fields contain the following text: 'Apartment Sri Gontong', 'Gombak, Selangor', '2', '180', 'Azfar Asyraf', and '0117654321'. At the bottom of the form is a blue button labeled 'Update Post'.

Figure 16 : Edit Post Page

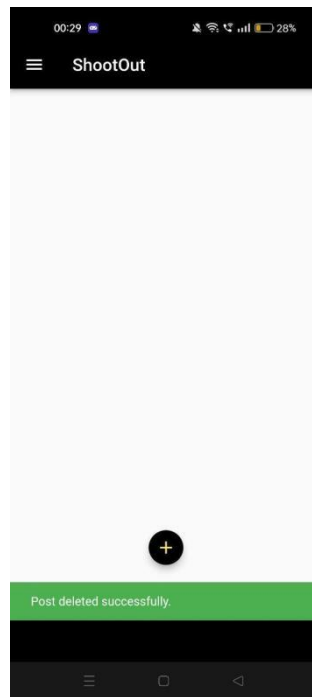


Figure 17 : Pop-up Notification Version 1

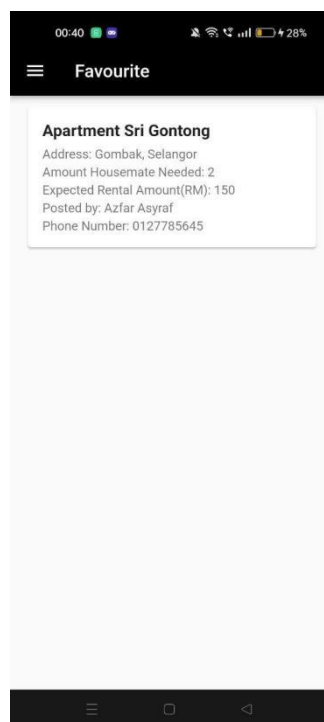


Figure 18 ; Favourite Page

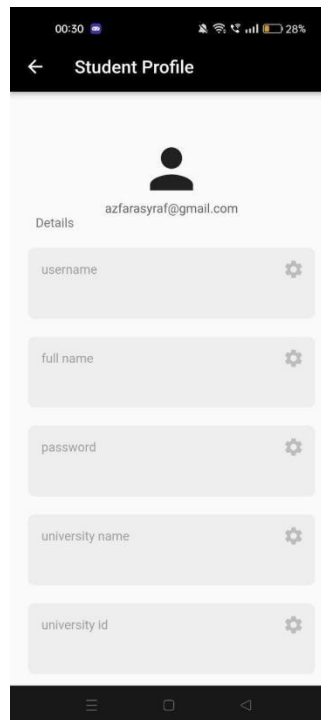


Figure 19 : Profile Page

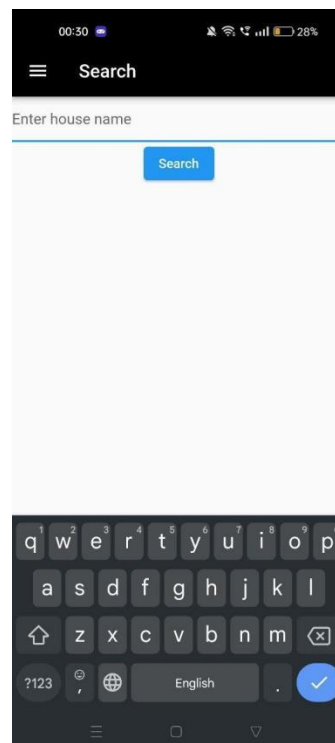


Figure 20 : Search Page

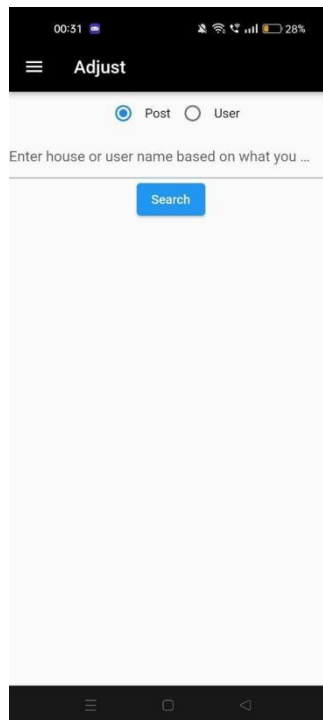


Figure 21 : Adjust Page - Post

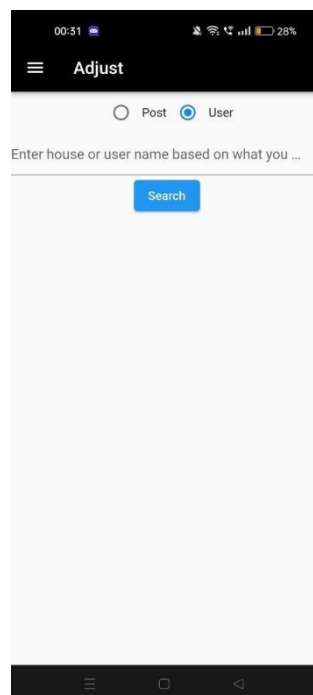


Figure 22 : Adjust Page - User

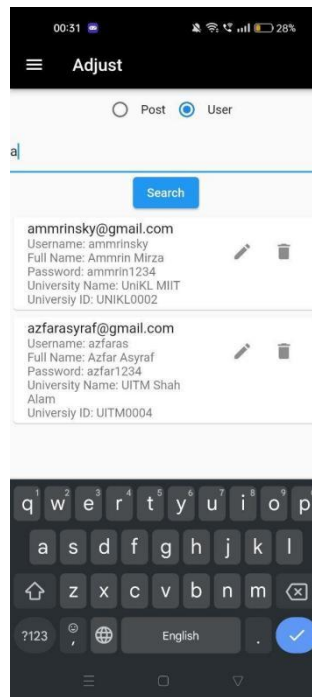


Figure 23 : Adjust Page - User Result

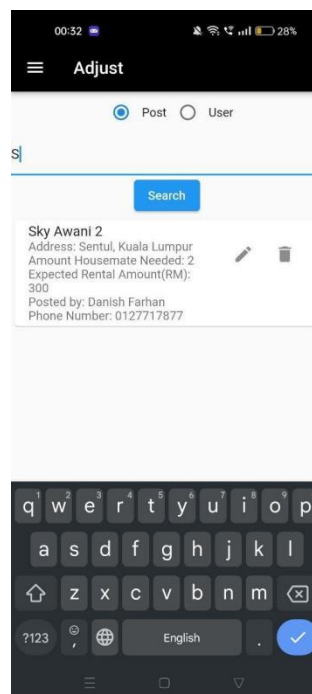


Figure 24 - Adjust Page - Post Result

00:32 28%

← Edit Post

House:
Sky Awani 2

Address:
Sentul, Kuala Lumpur

Amount Housemate Needed:
2

Posted by:
Danish Farhan

Expected Rental Amount(RM):
300

Save Changes

Figure 25 : Edit Post Page for Admin

00:32 28%

← Edit User

Email
ammrinsky@gmail.com

Username
ammrinsky

Full Name
Ammrin Mirza

Password
ammrin1234

University Name
UniKL MIIT

University ID
UNIKL0002

Update User

Figure 26 : Edit User Page for Admin

4.4 CONCLUSION

In conclusion, methodology that been used for development of HousemateHeaven prototype is iterative and incremental model. In the designphase of the model, use case and ERD diagram been showed to show the flow of the application. In the last part of the chapter, the limitations of the application are been talked. Limitations been highlighted during prototype development to make sure that the final product able to satisfy client requirement and is well-designed.

CHAPTER 5 : TESTING AND RESULTS

5.1 INTRODUCTION

In this part, testing phase of HousemateHeaven mobile application will be explained. Testing is needed in developing software because of to improve the standard of the system which includes functionality and capabilities of the system. It also important to avoid any errors or defects that may happen during development of the system.

5.2 TESTING APPROACH

The test approach that will be used for HousemateHeaven will be Regression Testing. The system will be tested by developer. The testing will be done at Universiti Kuala Lumpur - Malaysian Institute of Information Technology (UniKL MIIT) during a gap between lecture session.

5.3 TESTING PREPARATION

5.3.1 Software Preparation

- A) Visual Studio Code**
- B) Firebase-Firestore**

5.3.2 Hardware Preparation

No.	Items	Description
1	Laptop	Asus Nitro 5
2	RAM	16 GB
3	Operating System	Windows 11
4	CPU	Intel® Core(TM) i5-10300H, 2.50 GHz
5	Hard Disk4	475 GB

Table 9 : Hardware Preparation

5.4 TESTING SCHEDULE

Task	Milestone	Date
Planning	Completion of Section 1 and 2	26/12/2022
Analysis and Design	Completion of Section 3	6/1/2023
Test Environment Set-Up	Completion of Section 4	5/4/2023
Test Implementation and Execution	Completion of Section 5	12/5/2023
Test Monitoring and Controlling	Submission of Software Test Plan	3/6/2023

Table 10 : Testing Schedule

5.5 TEST CASES

5.5.1 Test Case : Module Admin Login

Tested By:	Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level	High		
Test Design Technique	Use Case Testing		
Test Type	Functional Testing		
Test Case Number	TC_001		
Test Case Name	Login		
Test Case Description	To test login to application as admin		
Item(s) to be tested			
1	Email		
2	Password		
Pre-Condition:	None		
Post Condition:	Able to login as admin		
Specifications			
Input		Expected Output	
Email: adminhh@gmail.com Password: adminhh1234		Login successful	
Procedural Steps			
1	Enter email and password		
2	Click Login		

Table 11 : Test Case Statement - Login

5.5.2 Test Case : Module Admin Search

Tested By:	Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level	High		
Test Design Technique	Use Case Testing		
Test Type	Functional Testing		
Test Case Number	TC_002		
Test Case Name	Search Profile		
Test Case Description	To search for any student profile as admin		
Item(s) to be tested			
1	Search profile function		
Pre-Condition:	Login as admin and one or more student data exist		
Post Condition:	Able to search for student profile		
Specifications			
Input		Expected Output	
Keyword : Ammrin Mirza		Display Ammrin Mirza profile	
Procedural Steps			
1	Click search by user		
2	Enter keyword		
3	Click search button		

Table 12 : Test Case Statement - Search Profile

Tested By:	Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level	High		
Test Design Technique	Use Case Testing		
Test Type	Functional Testing		
Test Case Number	TC_003		
Test Case Name	Search Form		
Test Case Description	To search any form that created by student by admin		
Item(s) to be tested			
1	Search form function		
Pre-Condition:	Login as admin and one or more student data exist		
Post Condition:	Able to search for student profile		
Specifications			
Input		Expected Output	
Keyword : Sky Awani		Display Sky Awani related form	
Procedural Steps			
1	Click search by form		
2	Enter keyword		
3	Click search button		

Table 13: Test Case Statement - Search Form

5.5.3 Test Case : Module Admin Adjust

Tested By:	Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level	High		
Test Design	Use Case Testing		
Technique			
Test Type	Functional Testing		
Test Case Number	TC_004		
Test Case Name	Edit Profile		
Test Case Description	To edit student profile as admin		
Item(s) to be tested			
1	Edit student profile function		
Pre-Condition:	Already search for student profile and it exist		
Post Condition:	Able to edit the student profile		
Specifications			
Input		Expected Output	
Click edit button		Edit can be done	
Procedural Steps			
1	Search for student profile		
2	Click edit button on student profile		

Table 14: Test Case Statement - Edit Profile

Tested By:	Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level	High		
Test Design Technique	Use Case Testing		
Test Type	Functional Testing		
Test Case Number	TC_005		
Test Case Name	Delete Profile		
Test Case Description	To test delete any student profile function as admin		
Item(s) to be tested			
1	Delete student profile function		
Pre-Condition:	Already search for student profile and it exist		
Post Condition:	Able to delete the student profile		
Specifications			
Input		Expected Output	
Click delete button		Delete successfully	
Procedural Steps			
1	Search student profile		
2	Click delete button on student profile		

Table 15: Test Case Statement - Delete Profile

Tested By:	Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level	High		
Test Design Technique	Use Case Testing		
Test Type	Functional Testing		

Test Case Number		TC_006	
Test Case Name		Edit Form	
Test Case Description		To test edit form function as admin	
Item(s) to be tested			
1	Edit form function		
Pre-Condition:		Search any form that created by student	
Post Condition:		Able to edit the form	
Specifications			
Input		Expected Output	
Click edit form button		Form can be edit	
Procedural Steps			
1	Search form created by student		
2	Click edit button		

Table 16: Test Case Statement - Edit Form

Tested By:	Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level	High		
Test Design Technique	Use Case Testing		
Test Type	Functional Testing		
Test Case Number	TC_007		
Test Case Name	Delete Form		
Test Case Description	To test delete form function as admin		
Item(s) to be tested			
1	Delete function		
Pre-Condition:	Search any form created by student		
Post Condition:	Able to delete the form		
Specifications			
Input		Expected Output	
Click delete button		Form successfully deleted	
Procedural Steps			
1	Search any form created by student		
2	Click delete button		

Table 17: Test Case Statement - Delete Form

5.5.4 Test Case: Module Admin Logout

Tested By:	Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level	High		
Test Design Technique	Use Case Testing		
Test Type	Functional Testing		
Test Case Number	TC_008		
Test Case Name	Logout		
Test Case Description	To test logout from the application as admin		
Item(s) to be tested			
1	Logout function		
Pre-Condition:	Login to application as admin		
Post Condition:	Logout from application		
Specifications			
Input		Expected Output	
Click logout button		Logout successfully	
Procedural Steps			
1	Login to application		
2	Click logout button		

Table 18 : Test Case Statement - Logout

5.5.5 Test Case : Module Student Register

Tested By:		Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level		High		
Test Design Technique		Use Case Testing		
Test Type		Functional Testing		
Test Case Number		TC_009		
Test Case Name		Register		
Test Case Description		To test register function as student		
Item(s) to be tested				
1	Full name			
2	Username			
3	Email			
4	Password			
5	Password confirmation			
6	University name			
7	University ID			
Pre-Condition:		Student does not have an account yet		
Post Condition:		Student can register to application		
Specifications				
Input			Expected Output	
Full name : Username : Email : Password : Password confirmation : University name : University id :			All information register into database	

Procedural Steps	
1	Click register option
2	Enter full name, username, email, password, password confirmation, university name and university id
3	Click register button

Table 19 : Test Case Statement - Register

5.5.6 Test Case : Module Student Login

Tested By:		Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level		High		
Test Design Technique		Use Case Testing		
Test Type		Functional Testing		
Test Case Number		TC_010		
Test Case Name		Login		
Test Case Description		To test login function as student		
Item(s) to be tested				
1	Email			
2	Password			
Pre-Condition:		Student already have an account		
Post Condition:		Student can login to application		
Specifications				
Input			Expected Output	
Email :			Login successful	
Password :				
Procedural Steps				
1	Enter email and password			
2	Click login button			

Table 20 : Test Case Statement - Login

5.5.7 Test Case : Module Student Role

Tested By:	Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level	High		
Test Design Technique	Use Case Testing		
Test Type	Functional Testing		
Test Case Number	TC_011		
Test Case Name	Choose Role		
Test Case Description	To test choose role function as student		
Item(s) to be tested			
1	Choose role function		
Pre-Condition:	Student already login to application		
Post Condition:	Student can choose role		
Specifications			
Input		Expected Output	
Click any role		Login as the picked role	
Procedural Steps			
1	Login to application as student		
2	Click any role		

Table 21 : Test Case Statement - Choose Role

5.5.8 Test Case : Module Student Homepage

Tested By:	Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level	High		
Test Design Technique	Use Case Testing		
Test Type	Functional Testing		
Test Case Number	TC_011		
Test Case Name	View Tutorial		
Test Case Description	To test view tutorial function as student		
Item(s) to be tested			
1	View tutorial function		
Pre-Condition:	Student already choose role		
Post Condition:	Student can view tutorial in homepage		
Specifications			
Input		Expected Output	
Click homepage section		Tutorial can be seen	
Procedural Steps			
1	Login to application as student		
2	Choose any roles		
3	Click homepage section		

Table 22 : Test Case Statement - View Tutorial

5.5.9 Test Case : Module Student Shootout

Tested By:		Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level		High		
Test Design Technique		Use Case Testing		
Test Type		Functional Testing		
Test Case Number		TC_012		
Test Case Name		Create Form		
Test Case Description		To test create form function as student		
Item(s) to be tested				
1	Create form function			
Pre-Condition:		Student choose “ Want to Search For Housemate “ role		
Post Condition:		Student can create a form		
Specifications				
Input			Expected Output	
House name : House address : Housemate needed : Rental Amount : Posted By: Phone Number:			Form can be created	
Procedural Steps				
1	Choose “ Want to Search for Housemate “ role			
2	Click shootout section			
3	Click add button			
4	Enter house name, house address, housemate needed, rental amount, posted by and phone number information			
5	Click submit button			

Table 23 : Test Case Statement - Create Form

Tested By:		Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level		High		
Test Design Technique		Use Case Testing		
Test Type		Functional Testing		
Test Case Number		TC_013		
Test Case Name		Edit Form		
Item(s) to be tested				
1	Edit form function			
Pre-Condition:		Student already created one or more form		
Post Condition:		Student able to edit the form		
Specifications				
Input			Expected Output	
Click edit form button			Form can be edit	
Procedural Steps				
1	Click form			
2	Click edit button			
3	Change any information			
4	Click confirm			
Test Case Description		To edit form function as student		

Table 24 : Test Case Statement - Edit Form

Tested By:	Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level	High		
Test Design Technique	Use Case Testing		
Test Type	Functional Testing		
Test Case Number	TC_014		
Test Case Name	Delete Form		
Test Case Description	To test delete form function as student		
Item(s) to be tested			
1	Delete form function		
Pre-Condition:	Student already created one or more form		
Post Condition:	Student can delete form		
Specifications			
Input		Expected Output	
Click delete form button		Form can be deleted	
Procedural Steps			
1	Click form		
2	Click delete button		

Table 25 : Test Case Statement - Delete Form

Tested By:	Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level	High		
Test Design Technique	Use Case Testing		
Test Type	Functional Testing		
Test Case	TC_015		

Number		
Test Case Name		View Form
Test Case Description		To test view form function as student
Item(s) to be tested		
1	View form function	
Pre-Condition:		One or more form data exists
Post Condition:		Student can view form
Specifications		
Input		Expected Output
Click shootout section		List of form can be viewed
Procedural Steps		
1	Click shootout section	
2	Student can view form	

Table 26 : Test Case Statement - View form

Tested By:	Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level	High		
Test Design Technique	Use Case Testing		
Test Type	Functional Testing		
Test Case Number	TC_016		
Test Case Name	Bookmark Form		
Test Case Description	To test bookmark form function as student		
Item(s) to be tested			
1	Bookmark form function		
Pre-Condition:	One or more form data exists		
Post Condition:	Student can bookmark form		
Specifications			
Input		Expected Output	
Click love button		Form successfully bookmarked	
Procedural Steps			
1	Click any form		
2	Click bookmark button		

Table 27 : Test Case Statement - Bookmark Form

Tested By:	Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level	High		
Test Design Technique	Use Case Testing		
Test Type	Functional Testing		
Test Case	TC_008		

Number		
Test Case Name		Logout
Test Case Description		To test logout from the application as admin
Item(s) to be tested		
1	Logout function	
Pre-Condition:		Login to application as admin
Post Condition:		Logout from application
Specifications		
Input		Expected Output
Click logout button		Logout successfully
Procedural Steps		
1	Login to application	
2	Click logout button	

Table 28: Test Case Statement - Logout

5.5.10 Test Case : Module Student Search

Tested By:		Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level		High		
Test Design Technique		Use Case Testing		
Test Type		Functional Testing		
Test Case Number		TC_017		
Test Case Name		Search Form		
Test Case Description		To test search form function as student		
Item(s) to be tested				
1	Search form function			
2	Keyword			
Pre-Condition:		One or more form data exists.		
Post Condition:		Able to search any form		
Specifications				
Input			Expected Output	
Keyword : “ Sky Awani 2 “			Sky Awani 2 related form can be searched	
Procedural Steps				
1	Click search section			
2	Enter form detail			
3	Click search button			

Table 29 : Test Case Statement - Search Form

5.5.11 Test Case : Module Student Favourite

Tested By:	Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level	High		
Test Design Technique	Use Case Testing		
Test Type	Functional Testing		
Test Case Number	TC_018		
Test Case Name	View Bookmark		
Test Case Description	To test view bookmark function as student		
Item(s) to be tested			
1	View bookmark function		
Pre-Condition:	Student already login or already bookmark any form		
Post Condition:	Student can view bookmark		
Specifications			
Input		Expected Output	
Click bookmark section		Bookmark form can be viewed	
Procedural Steps			
1	Click bookmark section		
2	Able to view bookmark form		

Table 30 : Test Case Statement - View Bookmark

5.5.12 Test Case : Module Student Profile

Tested By:	Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level	High		
Test Design	Use Case Testing		

Technique	
Test Type	Functional Testing
Test Case Number	TC_019
Test Case Name	View Profile
Test Case Description	To test view profile function as student
Item(s) to be tested	
1	View profile function
Pre-Condition:	Student already register
Post Condition:	Able to view profile
Specifications	
Input	Expected Output
Click profile section	Profile can be viewed
Procedural Steps	
1	Click profile section
2	Profile can viewed

Table 31 : Test Case Statement - View Profile

Tested By:	Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level	High		
Test Design Technique	Use Case Testing		
Test Type	Functional Testing		
Test Case Number	TC_020		
Test Case Name	Edit Profile		
Test Case Description	To test edit profile function as student		
Item(s) to be tested			
1	Edit profile function		
Pre-Condition:	Student already register		
Post Condition:	Able to edit profile		
Specifications			
Input		Expected Output	
Click edit button		Edit can be done	
Procedural Steps			
1	Click edit button		
2	Change any information		
3	Click confirm button		

Table 32 : Test Case Statement - Edit Profile

5.5.13 Test Case : Module Student Logout

Tested By:	Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level	High		
Test Design	Use Case Testing		

Technique		
Test Type		Functional Testing
Test Case Number		TC_021
Test Case Name		Logout
Test Case Description		To test logoute function as student
Item(s) to be tested		
1	Logout function	
Pre-Condition:		Login to application as student
Post Condition:		Able to logout from the application
Specifications		
Input		Expected Output
Click Logout Button		Logout successful
Procedural Steps		
1	Login to application as student	
2	Click logout button	

Table 33: Test Case Statement - Logout

5.5.14 Test Case : Module Student Forgot Password

Tested By:	Ts. Dr. Husna Sarirah Husin	Test Level:	User Acceptance Testing
Risk Level	High		
Test Design Technique	Use Case Testing		
Test Type	Functional Testing		
Test Case Number	TC_022		
Test Case Name	Forgot Password		
Test Case Description	To test forgot password function as student		
Item(s) to be tested			
1	Forgot password function		
Pre-Condition:	Already register as a student in application		
Post Condition:	Able to login using new password		
Specifications			
Input		Expected Output	
Email:		Able to login using new password	
Procedural Steps			
1	Click forgot password button in login screen		
2	Enter email		
3	Click send email		
4	Check email for the reset password link		
5	Change to new password		
6	Login to application using new password		

Table 34: Test Case Statement - Forgot Password

5.6 TEST RESULTS

Test Sequence	Remarks (PASSED/ FAILED)
Register as Student	PASSED
Login as Student	PASSED
Pick role	PASSED
View Tutorial	PASSED
Create Form	PASSED
Edit Form	PASSED
Delete Form	PASSED
Search Form	PASSED
Search User	PASSED
Search Form	PASSED
Bookmark Form	PASSED
View Bookmark	PASSED
View Profile	FAILED
Edit Profile	FAILED
Logout	PASSED
Login as Admin	PASSED
Search User	PASSED
Search Form	PASSED
Edit Form	PASSED
Delete Form	PASSED
Logout	PASSED
Forgot Password	PASSED

Table 35 : Test Results

5.7 TEST SUMMARY

Total No. OfTest Cases Planned	Total No. OfTest Cases Executed	Total No. OfPassed Test Cases	Total No. Of. Failed Test Cases
22	22	20	2

Table 36 : Test Summary

5.8 CONCLUSION

Testing is an important part in development of the application. That is why in this chapter, testing was done to ensure detect any defects or erros in the development of HousemateHeaven application. Testing approach that been used to develop the application is user acceptance testing.

CHAPTER 6 : CONCLUSION

6.1 CONCLUSION

In this chapter, all finding of previous chapters will be summarized that will prove that HousemateHeaven mobile application is indeed build and developed correctly. Most of the requirements and modules successfully followed. The application also able to manage end user expectation. The HousemateHeaven mobile application focus on two main users which are admin and university student. Admin will be responsible in managing the data of the system while university student will be the main user that will use the main function of the application which is to find housemate. Research that been conducted during the development of the system also will be important in improving the system. Methodology that been used which is Iterative and Incremental model is a model that been referred to conduct the development of HousemateHeaven mobile application. Furthermore, most of the challenges that faced by developer In developing the system successfully been overcome to complete the modules of requirement of the mobile application. The development of the application follow the timeline, budget and requirement that proposed during the proposal to develop it. Testing of HousemateHeaven mobile application is done by using User Acceptance testing method. Testing of it has been done by tester after the development of the system. By doing the testing, the application can be improve and develop into more stable application.

6.3 LIMITATIONS

Limitation of development refer to operation that the system unable to operate successfully. This limitation will be improved in future. The first limitation of HousemateHeaven for user which in this case is student, cannot know the distance between the rental house and their university automatically. The second limitation is student might not be able to see or edit their profile information because of bug happened in the development of application. Next, student need to logout and login again to change their role do different functional in the application. Lastly, student might need to press the same button repeatedly before the application accept the action because of bug in the application.

6.2 RECOMMENDATIONS

The successes of HousemateHeaven mobile application will be more possible by considering recommendations that has been provided by some parties. The first recommendation is to improve the interface of the application to make it more user friendly. This will improve the end user experience in using this application. Security of the system need to be improved more to provide more safety validation to the users. For example, a process to check the background of the user before approving them to register to application and an obligation to make sure user use an unique password. Next, the application need to have customer service operation in it. It is important because If some of the users are having problem in using the application, they

can contact customer support to help them with it. The responsive and reliability of the customer service will be important aspect. Furthermore, HousemateHeaven mobile application need to follow regulation and laws to keep operate and make sure that users of the application will have more confidence in using the application. The application need to provide End-user License Agreement (EULA) in the system. This will make sure that HousemateHeaven will not break the rules in managing user data and not involved in any cases that will need interfere from the court in future. Finally, the application must fix the bug or any errors in the development to make surethat HousemateHeaven mobile application will be more smooth to use by the user without any constraint that will effect performance of the system.

REFERENCE

- Ahrentzen, S. (2003). Double indemnity of double delight? The health consequences of shared housing and 'doubling up'. *Journal of Social Issues*, 59, 547–568. doi:10.1111/1540-4560.00077
- Chatman, D., Broaddus, A., Spevack, A. (2019). Are movers irrational? On travel patterns, housing characteristics, social interactions, and happiness before and after a move. *Travel behavior and society*. 262- 271
- Despres, C. (1994). The meaning and experience of shared housing: Companionship, security and a home. In *Power by design: Proceedings of the Twenty-Fourth Annual Conference of the Environmental Design Research Association* (pp. 119–127). Oklahoma City, OK: EDRA.
- Kenyon, E., & Heath, S. (2001). Choosing this life: Narratives of choice among house sharers. *Housing Studies*, 16, 619–635. doi:10.1080/02673030120080080
- Nielsen, J. (1994). *Usability engineering*. Morgan Kaufmann Pub.
- Ozturk, A., Mutlu, T. (2010). The relationship between attachment style, subjective well-being, happiness and social anxiety among university students. *Procedia social and behavioral sciences*. 9.1772-

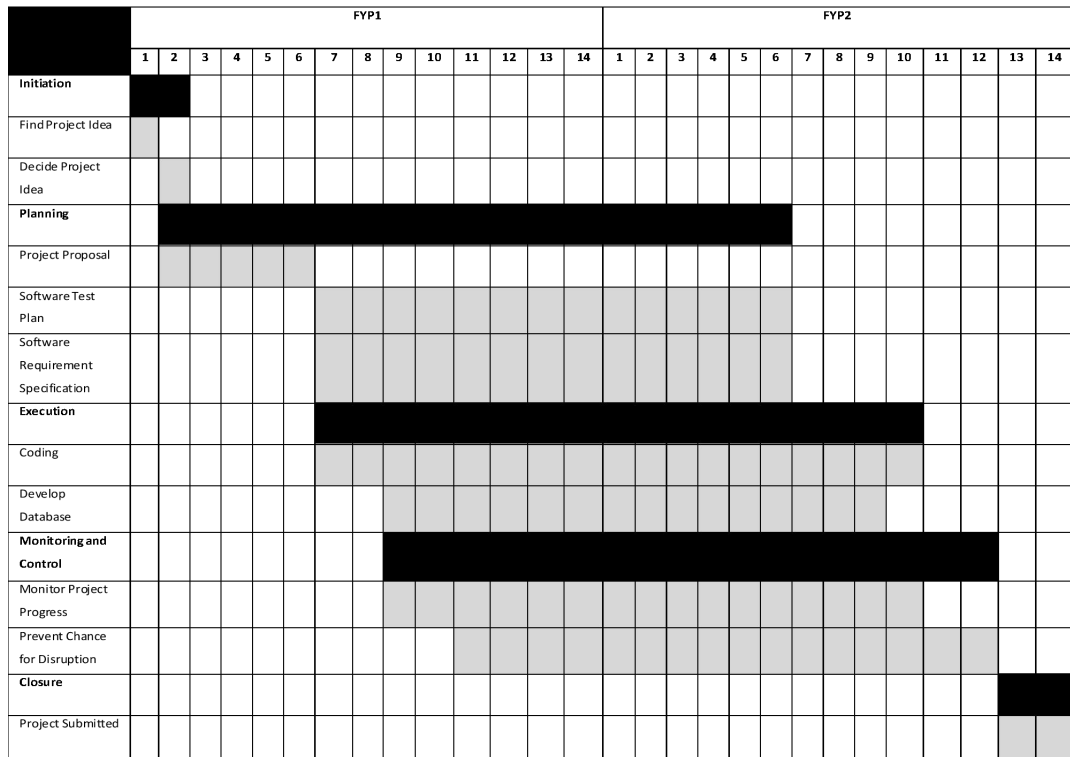
1776.

Schildbach, B, & Rukzio, E. (2010). Investigating selection and reading performance on a mobile phone while walking (Proceedings of the 12th international conference on human computer interaction with mobile devices and services). Lisbon, Portugal: ACM. 2010.

Venkatesh, V., Morris, M.G., Davis, G.B. and Davis, F.D. (2003) 'User acceptance of information technology: toward a unified view', MIS Quarterly, Vol. 27, No. 3, pp.425–478.

Walsh, A. (2015). The effect of social interaction on mental health nurse student learning. Nurse education in practice. 15. 7-12.

APPENDIX A : GANTT CHART



APPENDIX B : FIREBASE-FIRESTORE

🏠 > users > 1WPoU5cIAQ1tDh7pURdc More in Google Cloud		
housemateheaven-67791	users	1WPoU5cIAQ1tDh7pURdc
+ Start collection admin post users	+ Add document 1WPoU5cIAQ1tDh7pURdc 1ycabPCc86RpY7iXfIyt E81zPNpv1bx0hCsZ9xLi EzCr0h4vt26w312nXgXP J2EYI3z2xne0Bp1kq0M5 NRYfB2Agk13TyG1jCaxZ e7M1s99r1xkv4jJKPkAK k8ZI8MaG2G1a15Pj9ZIC 1Vv63q1Yw265Xbz3nV71	+ Start collection + Add field email: "syakirsya@gmail.com" full name: "Syakir Amir" password: "syakir1234" university id: "UITM0001" university name: "UITM Shah Alam" username: "syakir Amir"

🏠 > post > 0LdK0tgiaFHfw0aVCSHU More in Google Cloud		
housemateheaven-67791	post	0LdK0tgiaFHfw0aVCSHU
+ Start collection admin post users	+ Add document 0LdK0tgiaFHfw0aVCSHU 5AGD2caWGEsMSKw5FVVb UjI7hsQfjpMjpiAFnYMC 1ZS7oNxxYe93L3qTCb65 smQZeSWG1XFGScaN6160	+ Start collection + Add field address: "Gombak, Selangor" amount: "2" house: "Apartment Sri Gontong" money: "150" phone no: "0127785645" user: "mT680FrYjfMkRM5DYHrqR3FID5T2" user caller: "Azfar Asyraf"

🏠 > admin > admin More in Google Cloud		
housemateheaven-67791	admin	admin
+ Start collection admin post users	+ Add document admin	+ Start collection + Add field email: "adminhh@gmail.com" password: "adminhh1234"

APPENDIX C : CODING

Main.dart

```
import 'package:flutter/material.dart';
import 'package:housemateheaven/auth.dart';
import 'package:housemateheaven/pages_student/homepage.dart';
import 'package:housemateheaven/pages_studentrent/homepage2.dart';
import 'package:housemateheaven/post_model.dart';
import 'package:housemateheaven/role.dart';
import 'login.dart';
import 'onboarding.dart';
import 'pages_studentrent/favourite_provider.dart';
import 'register.dart';
import 'package:provider/provider.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:cloud_firestore/cloud_firestore.dart';
```

```
void main() async {
  WidgetsFlutterBinding.ensureInitialized();
  await Firebase.initializeApp();
  runApp(
    MultiProvider(
      providers: [
        ChangeNotifierProvider(create: (_) => FavoritePostsProvider()),
        StreamProvider<List<PostModel>>.value(
          value: FirebaseFirestore.instance
            .collection('post')
            .snapshots()
            .map((QuerySnapshot snapshot) {
              return snapshot.docs.map((DocumentSnapshot doc) {
                return PostModel(
                  id: doc.id,
                  house: doc['house'],
```

```

        address: doc['address'],
        amount: doc['amount'],
        money: doc['money'],
        phoneno: doc['phone no'],
        usercaller: doc['user caller'],
        user: doc['user'],
    );
  }).toList();
}),
initialData: [], // Provide an empty list as initial data
),
],
child: HousemateHeavenApp(),
),
);
}

```

```

class HousemateHeavenApp extends StatelessWidget {
  HousemateHeavenApp({Key? key}) : super(key: key);

```

```

  @override

```

```

  Widget build(BuildContext context) {
    return StreamBuilder<User?>(
      stream: FirebaseAuth.instance.authStateChanges(),
      builder: (context, snapshot) {
        if (snapshot.hasData) {
          final currentUser = snapshot.data;
          return MaterialApp(
            debugShowCheckedModeBanner: false,
            home: Scaffold(
              appBar: AppBar(
                backgroundColor: Colors.black,
              ),

```

```

        body: Role(),
      ),
    );
  } else {
    return MaterialApp(
      debugShowCheckedModeBanner: false,
      home: Scaffold(
        appBar: AppBar(
          backgroundColor: Colors.black,
        ),
        body: Auth(),
      ),
    );
  }
},
);
}
}

```

Register.dart

```

import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:flutter/material.dart';
import 'package:housemateheaven/login.dart';

```

```

class RegisterPage extends StatefulWidget {
  final VoidCallback showLoginPage;
  const RegisterPage({
    Key? key,
    required this.showLoginPage,
  }) : super(key: key);

```

```

@override
State<RegisterPage> createState() => _RegisterPageState();
}

class _RegisterPageState extends State<RegisterPage> {

  final TextEditingController _fullnameController = TextEditingController();
  final TextEditingController _usernameController = TextEditingController();
  final TextEditingController _emailController = TextEditingController();
  final TextEditingController _passwordController = TextEditingController();
  final      TextEditingController      _confirmpasswordController      =
TextEditingController();
  final TextEditingController _uninameController = TextEditingController();
  final TextEditingController _uniidController = TextEditingController();
  bool _isPasswordVisible = false;

  @override
  void dispose(){
    _fullnameController.dispose();
    _usernameController.dispose();
    _emailController.dispose();
    _passwordController.dispose();
    _confirmpasswordController.dispose();
    _uninameController.dispose();
    _uniidController.dispose();

    super.dispose();
  }

  Future<void> signUp() async {
    try {
      if (passwordConfirmed()) {
        await FirebaseAuth.instance.createUserWithEmailAndPassword(

```

```

        email: _emailController.text.trim(),
        password: _passwordController.text.trim(),
    );

    await addUserDetails(
        _fullNameController.text.trim(),
        _usernameController.text.trim(),
        _emailController.text.trim(),
        _passwordController.text.trim(),
        _uninameController.text.trim(),
        _uniidController.text.trim(),
    );
}
} catch (e, stackTrace) {
    print('Error during sign up: $e');
    print('Stack trace: $stackTrace');
}
}

```

```

bool passwordConfirmed(){
    if(_passwordController.text.trim() ==
        _confirmpasswordController.text.trim()) {
        return true;
    } else {
        return false;
    }
}

```

```

Future addUserDetails(String fullName, String username, String email,
String password, String uniName, String uniID) async {
    await FirebaseFirestore.instance.collection('users').add({
        'full name': fullName,
        'username': username,

```

```

    'email': email,
    'password': password,
    'university name': uniName,
    'university id': uniID,
  });
}

```

```

void togglePasswordVisibility() {
  setState(() {
    _isPasswordVisible = !_isPasswordVisible;
  });
}

```

```

@override
Widget build(BuildContext context) {
  return SafeArea(
    child : Scaffold(
      body : SingleChildScrollView(
        child: Container(
          padding: const EdgeInsets.all(30.0),
          child: Column(
            mainAxisAlignment: MainAxisAlignment.spaceEvenly,
            children: [
              Image(image: AssetImage('assests/image/logowhite.jpeg')),

              Form(
                child: Container(
                  padding: const EdgeInsets.symmetric(vertical: 20.0),

                  child: Column(
                    crossAxisAlignment: CrossAxisAlignment.start,
                    children: <Widget>[
                      TextField(

```

```

        controller: _fullnameController,
        decoration: InputDecoration(
          prefixIcon: Icon(Icons.person_outline_outlined),
          labelText: "Full Name",
          hintText: "Full Name",
          border: OutlineInputBorder(),
        ),
      ),
      const SizedBox(height: 30.0),
      TextField(
        controller: _usernameController,
        decoration: InputDecoration(
          prefixIcon: Icon(Icons.person_outline_outlined),
          labelText: "Username",
          hintText: "Username",
          border: OutlineInputBorder(),
        ),
      ),
      const SizedBox(height: 30.0),
      TextField(
        controller: _emailController,
        decoration: InputDecoration(
          prefixIcon: Icon(Icons.email_outlined),
          labelText: "Email",
          hintText: "Email",
          border: OutlineInputBorder(),
        ),
      ),
      const SizedBox(height: 30.0),
      TextField(
        controller: _passwordController,
        obscureText: !_isPasswordVisible,
        decoration: InputDecoration(

```

```

        prefixIcon: Icon(Icons.lock_outline),
        labelText: "Password",
        hintText: "password",
        border: OutlineInputBorder(),
        suffixIcon: IconButton(
          onPressed: togglePasswordVisibility,
          icon: Icon(_isPasswordVisible
            ? Icons.visibility
            : Icons.visibility_off),
        ),
      ),
    ),
  ),
  const SizedBox(height: 30.0),
  TextField(
    controller: _confirmpasswordController,
    obscureText: !_isPasswordVisible,
    decoration: InputDecoration(
      prefixIcon: Icon(Icons.lock_outline),
      labelText: "Password Confirmation",
      hintText: "Password Confirmation",
      border: OutlineInputBorder(),
      suffixIcon: IconButton(
        onPressed: togglePasswordVisibility,
        icon: Icon(_isPasswordVisible
          ? Icons.visibility
          : Icons.visibility_off),
      ),
    ),
  ),
  const SizedBox(height: 30.0),
  TextField(
    controller: _uninameController,
    decoration: InputDecoration(

```



```

        prefixIcon: Icon(Icons.school_outlined),
        labelText: "University Name",
        hintText: "University Name",
        border: OutlineInputBorder(),
      ),
    ),
    const SizedBox(height: 30.0),
    TextField(
      controller: _uniidController,
      decoration: InputDecoration(
        prefixIcon: Icon(Icons.school),
        labelText: "University ID",
        hintText: "University ID",
        border: OutlineInputBorder(),
      ),
    ),
    const SizedBox(height: 30.0),
    Container(
      height: 40,
      child: Material(
        borderRadius: BorderRadius.circular(20),
        shadowColor: Colors.black,
        color: Colors.black,
        elevation: 7,
        child: GestureDetector(
          onTap: signUp,
          child: Center(
            child: Text(
              'SIGN UP',
              style: TextStyle(
                color: Colors.white,
                fontWeight: FontWeight.bold,
                fontFamily: 'Montserrat'

```

```

        ),
      ),
    ),
  )
),
),
const SizedBox(height: 10.0),
Row(
  mainAxisAlignment: MainAxisAlignment.center,
  children: <Widget>[
    InkWell(
      onTap: widget.showLoginPage,
      child: Text(
        'Return to Login Page',
        style: TextStyle(
          color: Colors.black,
          fontFamily: 'Montserrat',
          fontWeight: FontWeight.bold
        ),
      ),
    ),
  ],
),
],
),
),
)
),
);

```

```
}  
}
```

Login.dart

```
import 'package:firebase_auth/firebase_auth.dart';  
import 'package:flutter/material.dart';  
import 'package:get/get.dart';  
import 'package:housemateheaven/forgetpassword.dart';  
import 'package:housemateheaven/pages_student/homepage.dart';  
import 'package:housemateheaven/register.dart';  
import 'package:housemateheaven/role.dart';  
import 'package:firebase_core/firebase_core.dart';  
import 'package:firebase_auth/firebase_auth.dart';  
import 'package:cloud_firestore/cloud_firestore.dart';  
  
import 'pages_admin/admin_page.dart';  
  
class LoginPage extends StatefulWidget {  
  final VoidCallback showRegisterPage;  
  const LoginPage({Key? key, required this.showRegisterPage}) : super(key:  
key);  
  
  @override  
  State<LoginPage> createState() => _LoginPageState();  
}  
  
class _LoginPageState extends State<LoginPage> {  
  final _emailController = TextEditingController();  
  final _passwordController = TextEditingController();  
  bool _isPasswordVisible = false;  
  bool _isAdmin = false;  
  static const String defaultAdminEmail = "adminhh@gmail.com";
```

```
static const String defaultAdminPassword = "adminhh1234";
```

```
@override
```

```
void dispose() {  
  _emailController.dispose();  
  _passwordController.dispose();  
  super.dispose();  
}
```

```
void togglePasswordVisibility() {  
  setState(() {  
    _isPasswordVisible = !_isPasswordVisible;  
  });  
}
```

```
Future<void> checkIfUserIsAdmin(User user) async {  
  try {  
    DocumentSnapshot adminSnapshot = await FirebaseFirestore.instance  
      .collection('admin')  
      .doc('admin')  
      .get();  
  
    if (adminSnapshot.exists) {  
      String adminEmail = adminSnapshot['email'];  
  
      setState(() {  
        _isAdmin = user.email == adminEmail;  
      });  
    }  
  } catch (e) {  
    print('Error checking if user is admin: $e');  
  }  
}
```

```

Future<void> signIn() async {
  try {
    final email = _emailController.text.trim();
    final password = _passwordController.text.trim();

    if (email == defaultAdminEmail && password == defaultAdminPassword)
    {
      setState(() {
        _isAdmin = true;
      });
    } else {
      UserCredential userCredential =
        await FirebaseAuth.instance.signInWithEmailAndPassword(
          email: email,
          password: password,
        );

      checkIfUserIsAdmin(userCredential.user!);
    }

    if (_isAdmin) {
      // Redirect the admin user to a specific page
      Navigator.pushReplacement(
        context,
        MaterialPageRoute(builder: (context) => Admin()),
      );
    } else {
      // Redirect regular users to a different page
      Navigator.pushReplacement(
        context,
        MaterialPageRoute(builder: (context) => Role()),
      );
    }
  }
}

```

```

    }
  } catch (e) {
    // Handle login errors
    print('Error logging in: $e');
  }
}

void navigateToRegisterPage() {
  // Call the callback function to navigate to the register page
  widget.showRegisterPage();
}

```

```

@override
Widget build(BuildContext context) {
  return SafeArea(
    child: Scaffold(
      body: SingleChildScrollView(
        child: Container(
          padding: const EdgeInsets.all(30.0),
          child: Column(
            mainAxisAlignment: MainAxisAlignment.spaceEvenly,
            children: [
              Image(image: AssetImage('assets/image/logowhite.jpeg')),
              Form(
                child: Container(
                  padding: const EdgeInsets.symmetric(vertical: 20.0),
                  child: Column(
                    crossAxisAlignment: CrossAxisAlignment.start,
                    children: [
                      TextField(
                        controller: _emailController,
                        decoration: InputDecoration(
                          prefixIcon: Icon(Icons.email_outlined),

```

```

        labelText: "Email",
        hintText: "Email",
        border: OutlineInputBorder(),
    ),
),
const SizedBox(height: 30.0),
TextField(
    controller: _passwordController,
    obscureText: !_isPasswordVisible,
    decoration: InputDecoration(
        prefixIcon: Icon(Icons.lock_outline),
        labelText: "Password",
        hintText: "password",
        border: OutlineInputBorder(),
        suffixIcon: IconButton(
            onPressed: togglePasswordVisibility,
            icon: Icon(_isPasswordVisible
                ? Icons.visibility
                : Icons.visibility_off),
        ),
    ),
),
const SizedBox(height: 10.0),
Align(
    alignment: Alignment.centerRight,
    child: TextButton(
        onPressed: () {
            Navigator.push(
                context,
                MaterialPageRoute(
                    builder: (context) => const ForgetPassword(),
                ),
            );
        },
    );

```

```

    },
    child: Text(
      'Forgot Password?',
      style: TextStyle(
        fontSize: 16,
        fontWeight: FontWeight.bold,
      ),
    ),
  ),
),
Container(
  height: 40,
  child: Material(
    borderRadius: BorderRadius.circular(20),
    shadowColor: Colors.black,
    color: Colors.black,
    elevation: 7,
    child: GestureDetector(
      onTap: signIn,
      child: Center(
        child: Text(
          'LOGIN',
          style: TextStyle(
            color: Colors.white,
            fontWeight: FontWeight.bold,
            fontFamily: 'Montserrat'),
        ),
      ),
    ),
  ),
),
const SizedBox(height: 10.0),
Row(

```



```

class Role extends StatefulWidget {
  const Role({Key? key}) : super(key: key);

  @override
  State <Role> createState() => RoleState();
}

class RoleState extends State <Role> {
  @override
  Widget build(BuildContext context) {
    var height = MediaQuery.of(context).size.height;
    return Scaffold(
      body: Container(
        padding: EdgeInsets.all(30.0),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.spaceEvenly,
          children: [
            Image(image: AssetImage('assets/image/logowhite.jpeg'), height:
height * 0.3),
            Column(
              children: [
                Text(
                  "Welcome to HousemateHeaven Application",
                  style: Theme.of(context).textTheme.headline4,
                ),
                Text(
                  "Please Choose Your Role for Us to Determine Which Functionlity
You Want To Use",
                  style: Theme.of(context).textTheme.bodyText1,
                ),
              ],
            ),
          ],
        ),
      ),
    );
  }
}

```

```

Row(
  children: [
    Expanded(
      child: ElevatedButton(
        onPressed: (){
          Navigator.push(
            context,
            MaterialPageRoute(builder: (context) => const HomePage()),);
        },
        child: Text('Find Housemate')
      ),
    ),
    const SizedBox(width: 10.0,),
    Expanded(
      child: ElevatedButton(
        onPressed: (){
          Navigator.push(
            context,
            MaterialPageRoute(builder: (context) => const HomePage2()),);
        },
        child: Text('Looking For A House To Share')
      ),
    ),
  ],
),
);
}
}

```

User.dart

```
import 'package:flutter/material.dart';
import 'package:housemateheaven/pages_student/homepage.dart';
import 'package:housemateheaven/pages_studentrent/homepage2.dart';
import 'package:housemateheaven/widget/navdrawer2.dart';

import 'widget/navdrawer.dart';

class Role extends StatefulWidget {
  const Role({Key? key}) : super(key: key);

  @override
  State <Role> createState() => RoleState();
}

class RoleState extends State <Role> {
  @override
  Widget build(BuildContext context) {
    var height = MediaQuery.of(context).size.height;
    return Scaffold(
      body: Container(
        padding: EdgeInsets.all(30.0),
        child: Column(
          mainAxisAlignment: MainAxisAlignment.spaceEvenly,
          children: [
            Image(image: AssetImage('assests/image/logowhite.jpeg'), height:
height * 0.3),
            Column(
              children: [
                Text(
                  "Welcome to HousemateHeaven Application",
                  style: Theme.of(context).textTheme.headline4,
                ),

```

```

Text(
  "Please Choose Your Role for Us to Determine Which Functionlity
You Want To Use",
  style: Theme.of(context).textTheme.bodyText1,
),
],
),
Row(
  children: [
    Expanded(
      child: ElevatedButton(
        onPressed: (){
          Navigator.push(
            context,
            MaterialPageRoute(builder: (context) => const HomePage()),);
        },
        child: Text('Find Housemate')
      ),
    ),
    const SizedBox(width: 10.0,),
    Expanded(
      child: ElevatedButton(
        onPressed: (){
          Navigator.push(
            context,
            MaterialPageRoute(builder: (context) => const HomePage2()),);
        },
        child: Text('Looking For A House To Share')
      ),
    ),
  ],
),
],

```

```

    ),

    ),
  );
}
}

```

User_model.dart

```
import 'package:cloud_firestore/cloud_firestore.dart';
```

```

class UserModel {
  final String? id;
  final String email;
  final String fullName;
  final String password;
  final String uniName;
  final String uniID;
  final String username;

  const UserModel({
    this.id,
    required this.email,
    required this.fullName,
    required this.password,
    required this.uniName,
    required this.uniID,
    required this.username,
  });

  toJson() {
    return {"email": email, "full name": fullName, "password": password,
"university name": uniName, "university id": uniID, "username": username};
  }
}

```

```

//map user fetch from firebase to userModel
UserModel? _userFromFirebaseSnapshot(DocumentSnapshot? snapshot) {
  if (snapshot == null || !snapshot.exists) {
    return null;
  }

  final data = snapshot.data() as Map<String, dynamic>;

  return UserModel(
    id: snapshot.id,
    email: data['email'] ?? "",
    fullName: data['full name'] ?? "",
    password: data['password'] ?? "",
    uniName: data['university name'] ?? "",
    uniID: data['university id'] ?? "",
    username: data['username'] ?? "",
  );
}
}

```

Text_box.dart

```

import 'package:flutter/material.dart';

class MyTextBox extends StatelessWidget {
  final String text;
  final String sectionName;
  final void Function()? onPressed;

  const MyTextBox({
    required this.text,
    required this.sectionName,

```

```

        required this.onPressed,
    });

    @override
    Widget build(BuildContext context) {
    return Container(
      decoration: BoxDecoration(
        color: Colors.grey[200],
        borderRadius: BorderRadius.circular(8),
      ),
      padding: const EdgeInsets.only(left: 15, bottom: 15),
      margin: const EdgeInsets.only(left: 20, right: 20, top: 20),
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Row(
            mainAxisAlignment: MainAxisAlignment.spaceBetween,
            children: [
              //sectionname
              Text(
                sectionName,
                style: TextStyle(color: Colors.grey[500]),
              ),

              IconButton(
                onPressed: onPressed,
                icon: Icon(
                  Icons.settings,
                  color: Colors.grey[400],
                ),
              ),
            ],
          ),
        ],
      ),
    );
  }
}

```



```

        //text
        Text(text),
      ],
    ),
  );
}
}

```

Post_model.dart

```

import 'pages_student/post.dart';
import 'package:flutter/material.dart';

```

```

class PostModel {
  final String id;
  final String house;
  final String address;
  final String amount;
  final String money;
  final String phoneno;
  final String usercaller;
  final String user;

  PostModel({
    required this.id,
    required this.house,
    required this.address,
    required this.amount,
    required this.money,
    required this.phoneno,
    required this.usercaller,
    required this.user,
  });
}

```

Generated_plugin_registered.dart

```
import 'package:cloud_firestore_web/cloud_firestore_web.dart';
import 'package:firebase_auth_web/firebase_auth_web.dart';
import 'package:firebase_core_web/firebase_core_web.dart';

import 'package:flutter_web_plugins/flutter_web_plugins.dart';

// ignore: public_member_api_docs
void registerPlugins(Registrar registrar) {
  FirebaseFirestoreWeb.registerWith(registrar);
  FirebaseAuthWeb.registerWith(registrar);
  FirebaseCoreWeb.registerWith(registrar);
  registrar.registerMessageHandler();
}
```

Forgetpassword.dart

```
import 'package:flutter/material.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:housemateheaven/login.dart';
import 'package:get/get.dart';

class ForgetPassword extends StatefulWidget {
  const ForgetPassword({super.key});

  @override
  State<ForgetPassword> createState() => _ForgetPasswordState();
}

class _ForgetPasswordState extends State<ForgetPassword> {
```

```
TextEditingController forgetPasswordController = TextEditingController();
```

```
@override
```

```
Widget build(BuildContext context) {  
  return Scaffold(  
    appBar: AppBar(  
      backgroundColor: Colors.black,  
      title: Text('Forget Password'),  
    ),  
    body: SingleChildScrollView(  
      padding: EdgeInsets.all(16),  
      child: Column(  
        children: [  
          Container(  
            alignment: Alignment.center,  
            height: 250.0,  
            child: Image.asset('assets/image/logowhite.jpeg'),  
          ),  
          SizedBox(height: 10.0),  
          Container(  
            margin: EdgeInsets.symmetric(horizontal: 30.0),  
            child: TextFormField(  
              controller: forgetPasswordController,  
              decoration: InputDecoration(  
                prefixIcon: Icon(Icons.email),  
                hintText: 'Email',  
                enabledBorder: OutlineInputBorder(),  
              ),  
            ),  
          ),  
          SizedBox(height: 10.0),  
          ElevatedButton(  
            onPressed: () async {
```

```

var forgotEmail = forgotPasswordController.text.trim();

try {
  FirebaseAuth.instance
    .sendPasswordResetEmail(email: forgotEmail)
    .then((value) => {
      print("Email Sent"),
      Navigator.push(
        context,
        MaterialPageRoute(
          builder: (context) => LoginPage(showRegisterPage: ()
{ },),
        ),
      ),
    });
} on FirebaseAuthException catch (e) {
  print("Error $e");
}
},
child: Text("Forgot Password"),
),
],
),
),
);
}
}

```

Authentication_repository.dart

```

import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:flutter/material.dart';

```

```

import 'package:housemateheaven/auth.dart';
import 'package:housemateheaven/login.dart';
import 'package:get/get.dart';
import 'package:housemateheaven/role.dart';

class AuthenticationRepository extends GetxController {

  static AuthenticationRepository get instance => Get.find();

  //variables
  final _auth = FirebaseAuth.instance;
  late final Rx<User?> firebaseUser;

  @override
  void onReady(){
    firebaseUser = Rx<User?>(_auth.currentUser);
    firebaseUser.bindStream(_auth.userChanges());
    ever(firebaseUser, _setInitialScreen);
  }

  _setInitialScreen(User? user) {
    user == null ? Get.offAll(() => const Auth()) : Get.offAll(() => const Role());
  }

  Future<void>  createUserWithEmailAndPassword(String email, String
password) async {
    try{
      await _auth.createUserWithEmailAndPassword(email: email, password:
password);
    } on FirebaseAuthException catch(e){

    } catch (_){}
  }
}

```

```
}
```

```
Future<void> loginWithEmailAndPassword(String email, String password)
async {
  try{
    await _auth.signInWithEmailAndPassword(email: email, password:
password);
  } on FirebaseAuthException catch(e){
    } catch (_){}
  }
}
```

```
Future<void> logout() async => await _auth.signOut();
}
```

Auth.dart

```
import 'package:flutter/material.dart';
import 'package:housemateheaven/login.dart';
import 'package:housemateheaven/register.dart';
```

```
class Auth extends StatefulWidget {
  const Auth({Key? key}) : super(key: key);
```

```
  @override
  State<Auth> createState() => _AuthState();
}
```

```
class _AuthState extends State<Auth> {
```

```
  bool showLoginPage = true;
```

```
  void toggleScreens(){
    setState(() {
      showLoginPage =! showLoginPage;
```

```

    });
  }

  @override
  Widget build(BuildContext context) {
    if (showLoginPage){
      return LoginPage(showRegisterPage: toggleScreens);
    } else {
      return RegisterPage(showLoginPage: toggleScreens);
    }
  }
}

```

Navdrawer_admin.dart

```

import 'package:flutter/material.dart';
import 'package:housemateheaven/login.dart';
import 'package:housemateheaven/main.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:cloud_firestore/cloud_firestore.dart';

```

```

import '../pages_admin/admin_page.dart';

```

```

class NavDrawerAdmin extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Drawer(
      child: ListView(
        padding: EdgeInsets.zero,
        children: <Widget>[
          DrawerHeader(
            child: Text(

```

```

        'Menu',
        style: TextStyle(color: Colors.white, fontSize: 25),
      ),
      decoration: BoxDecoration(
        color: Colors.black,
      ),
    ),
    ListTile(
      leading: Icon(Icons.home),
      title: Text('Adjust'),
      onTap: () {
        // Open home page
        Navigator.push(
          context,
          MaterialPageRoute(
            builder: (context) => const Admin(),
          ),
        );
      },
    ),
    ListTile(
      leading: Icon(Icons.exit_to_app),
      title: Text('Logout'),
      onTap: () {
        FirebaseAuth.instance.signOut().then((value) {
          Navigator.pushAndRemoveUntil(
            context,
            MaterialPageRoute(builder: (context) =>
LoginPage(showRegisterPage: () { },)),
            (Route<dynamic> route) => false,
          );
        }).catchError((error) {
          print('Logout error: $error');
        });
      },
    ),
  ),
);

```



```

        });
    },
),
],
),
);
}
}

```

Navdrawer.dart

```

import 'package:flutter/material.dart';
import 'package:housemateheaven/login.dart';
import 'package:housemateheaven/main.dart';
import 'package:housemateheaven/pages_student/homepage.dart';
import 'package:housemateheaven/pages_student/shootout.dart';
import 'package:housemateheaven/pages_student/studentprofile.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:cloud_firestore/cloud_firestore.dart';

```

```

class NavDrawer extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Drawer(
      child: ListView(
        padding: EdgeInsets.zero,
        children: <Widget>[
          DrawerHeader(
            child: Text(
              'Menu',
              style: TextStyle(color: Colors.white, fontSize: 25),
            ),

```

```

        decoration: BoxDecoration(
          color: Colors.black,
        ),
      ),
    ListTile(
      leading: Icon(Icons.home),
      title: Text('Homepage'),
      onTap: () {
        //Open home page
        Navigator.push(
          context,
          MaterialPageRoute(
            builder: (context) => const HomePage(),
          ),
        );
      },
    ),
    ListTile(
      leading: Icon(Icons.campaign),
      title: Text('ShootOut'),
      onTap: () {
        //Open shootout
        Navigator.push(
          context,
          MaterialPageRoute(
            builder: (context) => ShootOut(),
          ),
        );
      },
    ),
    ListTile(
      leading: Icon(Icons.verified_user),
      title: Text('Profile'),

```

```

onTap: () {
  //Open profile page
  Navigator.push(
    context,
    MaterialPageRoute(
      builder: (context) => const StudentProfile(),
    ),
  );
},
),
ListTile(
  leading: Icon(Icons.exit_to_app),
  title: Text('Logout'),
  onTap: () {
    FirebaseAuth.instance.signOut().then((value) {
      Navigator.pushAndRemoveUntil(
        context,
        MaterialPageRoute(builder: (context) =>
LoginPage(showRegisterPage: () { },)),
        (Route<dynamic> route) => false,
      );
    }).catchError((error) {
      print('Logout error: $error');
    });
  },
),
],
),
);
}
}

```

Navdrawer2.dart

```

import 'package:flutter/material.dart';
import 'package:housemateheaven/login.dart';
import 'package:housemateheaven/main.dart';
import 'package:housemateheaven/pages_studentrent/favourite.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:housemateheaven/pages_studentrent/homepage2.dart';
import 'package:housemateheaven/pages_studentrent/search2.dart';
import 'package:housemateheaven/pages_studentrent/shootout2.dart';
import 'package:housemateheaven/pages_studentrent/studentprofile2.dart';

```

```

class NavDrawer2 extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Drawer(
      child: ListView(
        padding: EdgeInsets.zero,
        children: <Widget>[
          DrawerHeader(
            child: Text(
              'Menu',
              style: TextStyle(color: Colors.white, fontSize: 25),
            ),
            decoration: BoxDecoration(
              color: Colors.black,
            ),
          ),
          ListTile(
            leading: Icon(Icons.home),
            title: Text('Homepage'),
            onTap: () {

```

```

        //Open home page
        Navigator.push(
          context,
          MaterialPageRoute(
            builder: (context) => const HomePage2(),
          ),
        );
      },
    ),
    ListTile(
      leading: Icon(Icons.campaign),
      title: Text('ShootOut'),
      onTap: () {
        //Open shootout
        Navigator.push(
          context,
          MaterialPageRoute(
            builder: (context) => ShootOut2(),
          ),
        );
      },
    ),
    ListTile(
      leading: Icon(Icons.search),
      title: Text('Search'),
      onTap: () {
        //Open search page
        Navigator.push(
          context,
          MaterialPageRoute(
            builder: (context) => const Search2(),
          ),
        );
      },
    ),
  ),
);

```

```

        },
    ),
    ListTile(
      leading: Icon(Icons.bookmark),
      title: Text('Favourite'),
      onTap: () {
        //Open bookmark
        Navigator.push(
          context,
          MaterialPageRoute(
            builder: (context) => const Favourite(favoritePosts: []),
          ),
        );
      },
    ),
    ListTile(
      leading: Icon(Icons.verified_user),
      title: Text('Profile'),
      onTap: () {
        //Open profile page
        Navigator.push(
          context,
          MaterialPageRoute(
            builder: (context) => const StudentProfile2(),
          ),
        );
      },
    ),
    ListTile(
      leading: Icon(Icons.exit_to_app),
      title: Text('Logout'),
      onTap: () {
        FirebaseAuth.instance.signOut().then((value) {

```

```

        Navigator.pushAndRemoveUntil(
            context,
            MaterialPageRoute(builder: (context) =>
LoginPage(showRegisterPage: () { },)),
            (Route<dynamic> route) => false,
        );
    }).catchError((error) {
        print('Logout error: $error');
    });
},
),
],
),
);
}
}

```

Admin_editpost.dart

```

import 'package:flutter/material.dart';
import 'package:cloud_firestore/cloud_firestore.dart';

class EditPostPage extends StatefulWidget {
    final DocumentSnapshot postSnapshot;

    const EditPostPage({Key? key, required this.postSnapshot}) : super(key:
key);

    @override
    _EditPostPageState createState() => _EditPostPageState();
}

class _EditPostPageState extends State<EditPostPage> {
    TextEditingController _houseController = TextEditingController();

```

```
TextEditingController _addressController = TextEditingController();
TextEditingController _amountController = TextEditingController();
TextEditingController _moneyController = TextEditingController();
TextEditingController _usercallerController = TextEditingController();
```

```
@override
```

```
void initState() {
```

```
    super.initState();
```

```
    // Initialize the text controllers with the existing post data
```

```
    _houseController.text = widget.postSnapshot['house'];
```

```
    _addressController.text = widget.postSnapshot['address'];
```

```
    _amountController.text = widget.postSnapshot['amount'];
```

```
    _moneyController.text = widget.postSnapshot['money'];
```

```
    _usercallerController.text = widget.postSnapshot['user caller'];
```

```
}
```

```
@override
```

```
void dispose() {
```

```
    // Dispose the text controllers
```

```
    _houseController.dispose();
```

```
    _addressController.dispose();
```

```
    _amountController.dispose();
```

```
    _moneyController.dispose();
```

```
    _usercallerController.dispose();
```

```
    super.dispose();
```

```
}
```

```
void _saveChanges() {
```

```
    // Get the updated values from the text controllers
```

```
    String updatedHouse = _houseController.text;
```

```
    String updatedAddress = _addressController.text;
```

```
    String updatedAmount = _amountController.text;
```



```

String updatedMoney = _moneyController.text;
String updatedusercaller = _usercallerController.text;

// Update the post data in the Firestore database
String postId = widget.postSnapshot.id;
FirebaseFirestore.instance
  .collection('post')
  .doc(postId)
  .update({
    'house': updatedHouse,
    'address': updatedAddress,
    'amount': updatedAmount,
    'money': updatedMoney,
    'usercaller': updatedusercaller,
  }).then((_) {
    print('Post updated successfully');
    // Show a success message or navigate back to the previous screen
  }).catchError((error) {
    print('Error updating post: $error');
    // Show an error message or handle the error appropriately
  });
}

```

```

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      backgroundColor: Colors.black,
      title: Text('Edit Post'),
    ),
    body: SingleChildScrollView(
      child: Container(
        padding: const EdgeInsets.all(16.0),

```

```

child: Column(
  crossAxisAlignment: CrossAxisAlignment.start,
  children: [
    Text('House:'),
    TextField(
      controller: _houseController,
    ),
    SizedBox(height: 16.0),
    Text('Address:'),
    TextField(
      controller: _addressController,
    ),
    SizedBox(height: 16.0),
    Text('Amount Housemate Needed:'),
    TextField(
      controller: _amountController,
    ),
    SizedBox(height: 16.0),
    Text('Posted by:'),
    TextField(
      controller: _usercallerController,
    ),
    SizedBox(height: 16.0),
    Text('Expected Rental Amount(RM):'),
    TextField(
      controller: _moneyController,
    ),
    SizedBox(height: 16.0),
    ElevatedButton(
      onPressed: _saveChanges,
      child: Text('Save Changes'),
    ),
  ],

```

```

        ),
      ),
    ),
  );
}
}

```

Admin_edituser.dart

```

import 'package:flutter/material.dart';
import 'package:cloud_firestore/cloud_firestore.dart';

class EditUserPage extends StatefulWidget {
  final DocumentSnapshot userSnapshot;

  const EditUserPage({Key? key, required this.userSnapshot}) : super(key:
key);

  @override
  _EditUserPageState createState() => _EditUserPageState();
}

class _EditUserPageState extends State<EditUserPage> {
  TextEditingController _emailController = TextEditingController();
  TextEditingController _usernameController = TextEditingController();
  TextEditingController _fullNameController = TextEditingController();
  TextEditingController _passwordController = TextEditingController();
  TextEditingController _uniNameController = TextEditingController();
  TextEditingController _uniIDController = TextEditingController();

  @override
  void initState() {
    super.initState();
    // Initialize the text controllers with the current user information

```

```

_emailController.text = widget.userSnapshot['email'];
_usernameController.text = widget.userSnapshot['username'];
_fullNameController.text = widget.userSnapshot['full name'];
_passwordController.text = widget.userSnapshot['password'];
_uniNameController.text = widget.userSnapshot['university name'];
_uniIDController.text = widget.userSnapshot['university id'];
}

```

@override

```

void dispose() {
_emailController.dispose();
_usernameController.dispose();
_fullNameController.dispose();
_passwordController.dispose();
_uniNameController.dispose();
_uniIDController.dispose();
super.dispose();
}

```

```

void _updateUser() {
// Get the updated values from the text controllers
String updatedEmail = _emailController.text.trim();
String updatedUsername = _usernameController.text.trim();
String updatedFullName = _fullNameController.text.trim();
String updatedPassword = _passwordController.text.trim();
String updatedUniName = _uniNameController.text.trim();
String updatedUniID = _uniIDController.text.trim();

// Update the user document in Firestore
FirebaseFirestore.instance
.collection('users')
.doc(widget.userSnapshot.id)
.update({

```

```

        'email': updatedEmail,
        'username': updatedUsername,
        'full name': updatedFullName,
        'password': updatedPassword,
        'university name': updatedUniName,
        'university id': updatedUniID,
    }).then((value) {
        // Show a success message or navigate back to the previous screen
        print('User updated successfully');
        Navigator.pop(context); // Navigate back to the previous screen
    }).catchError((error) {
        // Show an error message
        print('Error updating user: $error');
    });
}

```

```

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      backgroundColor: Colors.black,
      title: Text('Edit User'),
    ),
    body: Padding(
      padding: const EdgeInsets.all(16.0),
      child: SingleChildScrollView(
        child: Column(
          children: [
            TextField(
              controller: _emailController,
              decoration: InputDecoration(labelText: 'Email'),
            ),
            TextField(

```

```

        controller: _usernameController,
        decoration: InputDecoration(labelText: 'Username'),
    ),
    TextField(
        controller: _fullNameController,
        decoration: InputDecoration(labelText: 'Full Name'),
    ),
    TextField(
        controller: _passwordController,
        decoration: InputDecoration(labelText: 'Password'),
    ),
    TextField(
        controller: _uniNameController,
        decoration: InputDecoration(labelText: 'University Name'),
    ),
    TextField(
        controller: _uniIDController,
        decoration: InputDecoration(labelText: 'University ID'),
    ),
    SizedBox(height: 16.0),
    ElevatedButton(
        onPressed: _updateUser,
        child: Text('Update User'),
    ),
  ],
),
),
),
);
}
}

```

Admin_page.dart

```

import 'package:flutter/material.dart';
import 'package:housemateheaven/widget/navdrawer_admin.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:cloud_firestore/cloud_firestore.dart';

import 'admin_editpost.dart';
import 'admin_edituser.dart';

enum SearchCategory {
  Post,
  User,
}

class Admin extends StatefulWidget {
  const Admin({Key? key}) : super(key: key);

  @override
  State<Admin> createState() => _AdminState();
}

class _AdminState extends State<Admin> {
  final TextEditingController _searchController = TextEditingController();
  List<DocumentSnapshot> _searchResults = [];
  SearchCategory _selectedCategory = SearchCategory.Post;

  void _performSearch(String query) {
    String startRange = query;
    String endRange = query + '\uf8ff';

    CollectionReference collection;
    if (_selectedCategory == SearchCategory.Post) {
      collection = FirebaseFirestore.instance.collection('post');
    }
  }
}

```

```

collection
    .where('house', isGreaterThanOrEqualTo: startRange)
    .where('house', isLessThan: endRange)
    .get()
    .then((QuerySnapshot snapshot) {
    setState(() {
        _searchResults = snapshot.docs;
    });
    }).catchError((error) {
        print('Error performing search: $error');
    });

} else {
    collection = FirebaseFirestore.instance.collection('users');

    collection
        .where('email', isGreaterThanOrEqualTo: startRange)
        .where('email', isLessThan: endRange)
        .get()
        .then((QuerySnapshot snapshot) {
        setState(() {
            _searchResults = snapshot.docs;
        });
        }).catchError((error) {
            print('Error performing search: $error');
        });
    }

}

void _deletePost(DocumentSnapshot postSnapshot) {
    String postId = postSnapshot.id;

```



```

FirebaseFirestore.instance
  .collection('post')
  .doc(postId)
  .delete()
  .then((value) {
    print('Post deleted successfully');
    setState(() {
      _searchResults.remove(postSnapshot);
    });
  }).catchError((error) {
    print('Error deleting post: $error');
  });
}

void _deleteUser(DocumentSnapshot userSnapshot) {
  String userId = userSnapshot.id;

  FirebaseFirestore.instance
    .collection('users')
    .doc(userId)
    .delete()
    .then((value) {
      print('User deleted successfully');
      setState(() {
        _searchResults.remove(userSnapshot);
      });
    }).catchError((error) {
      print('Error deleting user: $error');
    });
}

void _editPost(DocumentSnapshot postSnapshot) {

```

```

Navigator.push(
  context,
  MaterialPageRoute(
    builder: (context) => EditPostPage(postSnapshot: postSnapshot),
  ),
);
}

```

```

void _editUser(DocumentSnapshot userSnapshot) {
  Navigator.push(
    context,
    MaterialPageRoute(
      builder: (context) => EditUserPage(userSnapshot: userSnapshot),
    ),
  );
}

```

```

@override
Widget build(BuildContext context) {
  return Scaffold(
    drawer: NavDrawerAdmin(),
    appBar: AppBar(
      backgroundColor: Colors.black,
      title: Text('Adjust'),
    ),
    body: Column(
      children: [
        Row(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Radio<SearchCategory>(
              value: SearchCategory.Post,
              groupValue: _selectedCategory,

```

```

        onChanged: (SearchCategory? value) {
          setState(() {
            _selectedCategory = value!;
          });
        },
      ),
      Text('Post'),
      Radio<SearchCategory>(
        value: SearchCategory.User,
        groupValue: _selectedCategory,
        onChanged: (SearchCategory? value) {
          setState(() {
            _selectedCategory = value!;
          });
        },
      ),
      Text('User'),
    ],
  ),
  TextField(
    controller: _searchController,
    decoration: InputDecoration(
      hintText: 'Enter house or user name based on what you choose
above',
    ),
  ),
  ElevatedButton(
    onPressed: () {
      String searchQuery = _searchController.text;
      _performSearch(searchQuery);
    },
    child: Text('Search'),
  ),

```

```

Expanded(
  child: ListView.builder(
    itemCount: _searchResults.length,
    itemBuilder: (context, index) {
      DocumentSnapshot snapshot = _searchResults[index];

      if (_selectedCategory == SearchCategory.Post) {
        // Display post data in a Card
        String house = snapshot['house'];
        String address = snapshot['address'];
        String amount = snapshot['amount'];
        String money = snapshot['money'];
        String phoneno = snapshot['phone no'];
        String usercaller = snapshot['user caller'];

        return Card(
          child: ListTile(
            title: Text(house),
            subtitle: Column(
              crossAxisAlignment: CrossAxisAlignment.start,
              children: [
                Text('Address: $address'),
                Text('Amount Housemate Needed: $amount'),
                Text('Expected Rental Amount(RM): $money'),
                Text('Posted by: $usercaller'),
                Text('Phone Number: $phoneno'),
              ],
            ),
            trailing: Row(
              mainAxisAlignment: MainAxisAlignment.min,
              children: [
                IconButton(
                  onPressed: () => _editPost(snapshot),

```

```

        icon: Icon(Icons.edit),
      ),
      IconButton(
        onPressed: () => _deletePost(snapshot),
        icon: Icon(Icons.delete),
      ),
    ],
  ),
),
);
} else {
  // Display user data in a Card
  String email = snapshot['email'];
  String username = snapshot['username'];
  String fullName = snapshot['full name'];
  String password = snapshot['password'];
  String uniName = snapshot['university name'];
  String uniID = snapshot['university id'];

  return Card(
    child: ListTile(
      title: Text(email),
      subtitle: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Text('Username: $username'),
          Text('Full Name: $fullName'),
          Text('Password: $password'),
          Text('University Name: $uniName'),
          Text('Universiy ID: $uniID'),
        ],
      ),
      trailing: Row(

```

```

        mainAxisAlignment: MainAxisAlignment.min,
        children: [
          IconButton(
            onPressed: () => _editUser(snapshot),
            icon: Icon(Icons.edit),
          ),
          IconButton(
            onPressed: () => _deleteUser(snapshot),
            icon: Icon(Icons.delete),
          ),
        ],
      ),
    ),
  );
}
},
),
),
],
),
);
}
}
}

```

Edit_post.dart

```

import 'package:flutter/material.dart';
import 'package:housemateheaven/pages_student/post.dart';
import 'package:housemateheaven/post_model.dart';
import 'package:provider/provider.dart';
import 'package:housemateheaven/pages_student/list.dart';

```

```

class EditPost extends StatefulWidget {
  final String postId;

```

```

EditPost({required this.postId});

@override
_EditPostState createState() => _EditPostState();
}

class _EditPostState extends State<EditPost> {
  PostService _postService = PostService();
  TextEditingController _houseController = TextEditingController();
  TextEditingController _addressController = TextEditingController();
  TextEditingController _amountController = TextEditingController();
  TextEditingController _moneyController = TextEditingController();
  TextEditingController _phonenoController = TextEditingController();
  TextEditingController _usercallerController = TextEditingController();

  @override
  void initState() {
    super.initState();
    // Fetch the post details using the postId
    _fetchPostDetails();
  }

  Future<void> _fetchPostDetails() async {
    try {
      PostModel post = await _postService.getPostById(widget.postId);
      if (post != null) {
        // Set the fetched post details to the respective text controllers
        _houseController.text = post.house;
        _addressController.text = post.address;
        _amountController.text = post.amount;
        _moneyController.text = post.money;
        _phonenoController.text = post.phoneno;
      }
    }
  }
}

```

```

        _usercallerController.text = post.usercaller;
    }
} catch (e) {
    print('Error fetching post details: $e');
}
}

```

```

Future<void> _updatePost() async {
    try {
        // Retrieve the updated values from the text controllers
        String updatedHouse = _houseController.text;
        String updatedAddress = _addressController.text;
        String updatedAmount = _amountController.text;
        String updatedMoney = _moneyController.text;
        String updatedphoneno = _phonenoController.text;
        String updatedusercaller = _usercallerController.text;

        // Call the post service to update the post
        await _postService.updatePost(widget.postId, updatedHouse,
        updatedAddress, updatedAmount, updatedMoney, updatedphoneno,
        updatedusercaller);

        // Show a success message or perform any necessary actions after
        updating the post
        ScaffoldMessenger.of(context).showSnackBar(
            SnackBar(
                content: Text('Post updated successfully.'),
                backgroundColor: Colors.green,
            ),
        );

        // Navigate back to the previous screen
        Navigator.pop(context);
    }
}

```



```

    } catch (e) {
        // Handle any error that occurred during the update process
        print('Error updating post: $e');
        // Show an error message or perform any necessary error handling
    }
}

```

```

@override
void dispose() {
    _houseController.dispose();
    _addressController.dispose();
    _amountController.dispose();
    _moneyController.dispose();
    _phonenoController.dispose();
    _usercallerController.dispose();
    super.dispose();
}

```

```

@override
Widget build(BuildContext context) {
    return Scaffold(
        appBar: AppBar(
            backgroundColor: Colors.black,
            title: Text('Edit Post'),
        ),
        body: Padding(
            padding: EdgeInsets.all(16),
            child: Column(
                children: [
                    TextField(
                        controller: _houseController,
                        decoration: InputDecoration(labelText: 'House'),
                    ),

```

```

        TextField(
          controller: _addressController,
          decoration: InputDecoration(labelText: 'Address'),
        ),
        TextField(
          controller: _amountController,
          decoration: InputDecoration(labelText: 'Amount Housemate
Needed'),
        ),
        TextField(
          controller: _moneyController,
          decoration: InputDecoration(labelText: 'Expected Rental
Amount(RM)'),
        ),
        TextField(
          controller: _usercallerController,
          decoration: InputDecoration(labelText: 'Posted by'),
        ),
        TextField(
          controller: _phonenoController,
          decoration: InputDecoration(labelText: 'Phone Number'),
        ),
        SizedBox(height: 16),
        ElevatedButton(
          onPressed: _updatePost,
          child: Text('Update Post'),
        ),
      ],
    ),
  ),
);
}
}

```

Homepage.dart

```
import 'package:flutter/material.dart';
import 'package:introduction_screen/introduction_screen.dart';
import '../widget/navdrawer.dart';
```

```
class HomePage extends StatefulWidget {
  const HomePage({
    Key? key,
  }) : super(key: key);

  @override
  _HomePageState createState() => _HomePageState();
}
```

```
class _HomePageState extends State<HomePage> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      drawer: NavDrawer(),
      appBar: AppBar(
        backgroundColor: Colors.black,
        title: Text('HousemateHeaven'),
      ),
      body: PageView(
        children: [
          Center(
            child: Container(
              child: Column(
                mainAxisAlignment: MainAxisAlignment.center,
                children: [
                  Image.asset('assests/image/logowhite.jpeg'),
                  SizedBox(height: 40),
                ],
              ),
            ),
          ],
        ),
      ),
    );
  }
}
```

```

Text(
  'Welcome to HousemateHeaven',
  style: TextStyle(
    color: Colors.black,
    fontSize: 20,
    fontWeight: FontWeight.bold,
  ),
  textAlign: TextAlign.center,
),
Text(
  'This is a tutorial if you selected Find Housemate role.',
  style: TextStyle(
    color: Colors.black,
    fontSize: 16,
  ),
  textAlign: TextAlign.center,
),
 SizedBox(height: 10),
Text(
  'Swipe right for more.',
  style: TextStyle(
    color: Colors.black,
    fontSize: 16,
    fontWeight: FontWeight.bold,
  ),
  textAlign: TextAlign.center,
),
],
),
),
),
Center(
  child: Container(

```

```

child: Column(
  mainAxisAlignment: MainAxisAlignment.center,
  children: [
    Image.asset('assets/image/sections.jpeg'),
    SizedBox(height: 20),
    Text(
      'There are four sections and you need to press section that you
want to use.',
      style: TextStyle(
        color: Colors.black,
        fontSize: 16,
      ),
      textAlign: TextAlign.center,
    ),
    SizedBox(height: 10),
    Text(
      'Swipe right for more.',
      style: TextStyle(
        color: Colors.black,
        fontSize: 16,
        fontWeight: FontWeight.bold,
      ),
      textAlign: TextAlign.center,
    ),
  ],
),
),
),
Center(
  child: Container(
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [

```

```

Text(
  'ShootOut:',
  style: TextStyle(
    color: Colors.black,
    fontSize: 16,
    fontWeight: FontWeight.bold,
  ),
  textAlign: TextAlign.center,
),
 SizedBox(height: 40),
Image.asset('assests/image/shootout.jpeg'),
SizedBox(height: 40),

```

In the shootout section, you can create a form that can be viewed by other users. You can create a form by clicking the add button located in the middle of the page. After creating a form, you can view, edit or delete it in this section',

```

  style: TextStyle(
    color: Colors.black,
    fontSize: 16,
  ),
  textAlign: TextAlign.center,
),
SizedBox(height: 10),
Text(
  'Swipe right for more.',
  style: TextStyle(
    color: Colors.black,
    fontSize: 16,
    fontWeight: FontWeight.bold,
  ),
  textAlign: TextAlign.center,
),

```

```

    ],
  ),
),
),
Center(
  child: Container(
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Text(
          'Profile:',
          style: TextStyle(
            color: Colors.black,
            fontSize: 16,
            fontWeight: FontWeight.bold,
          ),
          textAlign: TextAlign.center,
        ),
        SizedBox(height: 40),
        Image.asset('assets/image/profile.jpeg'),
        SizedBox(height: 40),
        Text(
          'In the profile section, you can view details that you registered
earlier. You can also edit any detail that you want',
          style: TextStyle(
            color: Colors.black,
            fontSize: 16,
          ),
          textAlign: TextAlign.center,
        ),
        SizedBox(height: 10),
        Text(
          'Swipe right for more.',

```

```

        style: TextStyle(
          color: Colors.black,
          fontSize: 16,
          fontWeight: FontWeight.bold,
        ),
        textAlign: TextAlign.center,
      ),
    ],
  ),
),
Center(
  child: Container(
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Text(
          'Logout:',
          style: TextStyle(
            color: Colors.black,
            fontSize: 16,
            fontWeight: FontWeight.bold,
          ),
          textAlign: TextAlign.center,
        ),
        SizedBox(height: 40),
        Image.asset('assets/image/logout.jpeg'),
        SizedBox(height: 40),
        Text(
          'In the logout section, you can logout from the application by pressing it.',
          style: TextStyle(
            color: Colors.black,

```



```

        fontSize: 16,
      ),
      textAlign: TextAlign.center,
    ),
    SizedBox(height: 10),
    Text(
      'That is all for tutorial on how to use the application. Hope you
will have a great time with HousemateHeaven!!!',
      style: TextStyle(
        color: Colors.black,
        fontSize: 16,
        fontWeight: FontWeight.bold,
      ),
      textAlign: TextAlign.center,
    ),
  ],
),
),
),
],
),
);
}
}

```

List.dart

```

import 'package:flutter/material.dart';
import 'package:housemateheaven/pages_student/edit_post.dart';
import 'package:housemateheaven/pages_student/post.dart';
import 'package:housemateheaven/post_model.dart';
import 'package:provider/provider.dart';

import '../user.dart';

```

```

import '../user_model.dart';

class ListPost extends StatefulWidget {
  const ListPost({Key? key}) : super(key: key);

  @override
  State<ListPost> createState() => _ListPostState();
}

class _ListPostState extends State<ListPost> {
  PostService _postService = PostService();
  UserService _userService = UserService();

  Future<void> _deletePost(String postId) async {
    try {
      await _postService.deletePost(postId);
      ScaffoldMessenger.of(context).showSnackBar(
        SnackBar(
          content: Text('Post deleted successfully.'),
          backgroundColor: Colors.green,
        ),
      );

      // Refresh the list of posts
      setState(() {});
    } catch (e) {
      print('Error deleting post: $e');
      ScaffoldMessenger.of(context).showSnackBar(
        SnackBar(
          content: Text('Error deleting post. Please try again.'),
          backgroundColor: Colors.red,
        ),
      );
    }
  }
}

```

```

    }
}

```

```

void _editPost(String postId) {
  Navigator.push(
    context,
    MaterialPageRoute(
      builder: (context) => EditPost(postId: postId),
    ),
  );
}

```

```

@override
Widget build(BuildContext context) {
  return Consumer<List<PostModel>>(
    builder: (context, posts, _) {
      return ListView.builder(
        itemCount: posts.length,
        itemBuilder: (context, index) {
          final post = posts[index];
          return StreamBuilder<UserModel>(
            stream: _userService.getUserInfo(post.user),
            builder: (BuildContext context, AsyncSnapshot<UserModel>
snapshot) {
              if (!snapshot.hasData) {
                return Center(child: CircularProgressIndicator());
              }
              final userModel = snapshot.data!;
              return Padding(
                padding: EdgeInsets.symmetric(horizontal: 16, vertical: 8),
                child: Card(
                  elevation: 2,
                  child: ListTile(

```

16),
contentPadding: EdgeInsets.symmetric(vertical: 12, horizontal:

```
title: Text(
  post.house,
  style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),
),
subtitle: Column(
  crossAxisAlignment: CrossAxisAlignment.start,
  children: [
    SizedBox(height: 8),
    Text(
      'Address: ${post.address}',
      style: TextStyle(fontSize: 14),
    ),
    SizedBox(height: 4),
    Text(
      'Housemate Needed: ${post.amount}',
      style: TextStyle(fontSize: 14),
    ),
    SizedBox(height: 4),
    Text(
      'Expected Rental Amount(RM): ${post.money}',
      style: TextStyle(fontSize: 14),
    ),
    SizedBox(height: 4),
    Text(
      'Posted by: ${post.usercaller}',
      style: TextStyle(fontSize: 14),
    ),
    SizedBox(height: 4),
    Text(
      'Phone Number: ${post.phoneno}',
      style: TextStyle(fontSize: 14),
```

```

        ),
      ],
    ),
    trailing: Row(
      mainAxisAlignment: MainAxisAlignment.min,
      children: [
        IconButton(
          icon: Icon(Icons.edit),
          onPressed: () {
            _editPost(post.id);
          },
        ),
        IconButton(
          icon: Icon(Icons.delete),
          onPressed: () {
            _deletePost(post.id);
          },
        ),
      ],
    ),
  ),
);
},
);
},
);
},
);
}
}

```

Post.dart

```

import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:housemateheaven/post_model.dart';
import 'package:housemateheaven/user_model.dart';

class PostService {
  List<PostModel> _postListFromSnapshot(QuerySnapshot snapshot) {
    return snapshot.docs.map((doc) {
      return PostModel(
        id: doc.id,
        house: doc['house'] ?? "",
        address: doc['address'] ?? "",
        amount: doc['amount'] ?? "",
        money: doc['money'] ?? "",
        phoneno : doc['phone no']?? "",
        usercaller: doc['user caller'] ?? "",
        user: doc['user'] ?? "",
      );
    }).toList();
  }
}

```

```

Future<void> savePost(String house, String address, String amount, String
money, String phoneno, String usercaller) async {
  User? currentUser = FirebaseAuth.instance.currentUser;
  if (currentUser != null) {
    // Get the current user's UID
    String uid = currentUser.uid;

    await FirebaseFirestore.instance.collection("post").add({
      'house': house,
      'address': address,
      'amount': amount,

```

```

        'money': money,
        'phone no' : phoneno,
        'user caller': usercaller,
        'user': uid, // Set the user field to the current user's UID
    });
}
}

```

```

Stream<List<PostModel>> getPostsByUser(String uid) {
    return FirebaseFirestore.instance
        .collection("post")
        .where('user', isEqualTo: uid)
        .snapshots()
        .map(_postListFromSnapshot);
}

```

```

Future<PostModel> getPostById(String postId) async {
    DocumentSnapshot snapshot = await
    FirebaseFirestore.instance.collection('post').doc(postId).get();
    return PostModel(
        id: snapshot.id,
        house: snapshot['house'] ?? "",
        address: snapshot['address'] ?? "",
        amount: snapshot['amount'] ?? "",
        money: snapshot['money'] ?? "",
        phoneno: snapshot['phone no'] ?? "",
        usercaller: snapshot['user caller'] ?? "",
        user: snapshot['user'] ?? "",
    );
}

```

```

Future<void> updatePost(String postId, String house, String address, String

```

```

amount, String money, String phoneno, String usercaller) async {
  await FirebaseFirestore.instance.collection('post').doc(postId).update({
    'house': house,
    'address': address,
    'amount': amount,
    'money': money,
    'phoneno': phoneno,
    'user caller': usercaller,
  });
}

Future<void> deletePost(String postId) async {
  await FirebaseFirestore.instance.collection('post').doc(postId).delete();
}

}

```

Rentalform.dart

```

import 'package:flutter/material.dart';
import 'package:housemateheaven/pages_student/shootout.dart';
import 'package:housemateheaven/pages_student/post.dart';

class RentalForm extends StatefulWidget {
  const RentalForm({Key? key}) : super(key: key);

  @override
  State<RentalForm> createState() => _RentalFormState();
}

class _RentalFormState extends State<RentalForm>{
  final PostService _postService = PostService();

  String house = "";

```



```

String address = "";
String amount = "";
String money = "";
String phoneno = "";
String usercaller = "";

var      _houseController,      _addressController,      _amountController,
_moneyController, _phonenoController, _usercallerController;

void _updateText(){

}

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      backgroundColor: Colors.black,
      title: const Text("Rental Form"),
      centerTitle: true,
    ),
    body: Container(
      padding: EdgeInsets.all(20.0),
      child: ListView(
        children: [
          TextFormField(
            onChanged: (val) {
              setState(() {
                house = val;
              });
            },
            controller: _houseController,

```

```

decoration: InputDecoration(
  labelText: 'House',
  prefixIcon: Icon(Icons.house_outlined),
  border: OutlineInputBorder(),
),
),
const SizedBox(height: 20.0),
TextFormField(
  onChanged: (val) {
    setState(() {
      address = val;
    });
  },
  controller: _addressController,
  decoration: InputDecoration(
    labelText: 'Address',
    prefixIcon: Icon(Icons.house),
    border: OutlineInputBorder(),
  ),
),
const SizedBox(height: 20.0),
TextFormField(
  onChanged: (val) {
    setState(() {
      amount = val;
    });
  },
  controller: _amountController,
  decoration: InputDecoration(
    labelText: 'Amount Housemate Needed',
    prefixIcon: Icon(Icons.supervised_user_circle),
    border: OutlineInputBorder(),
  ),
),

```

```

),
const SizedBox(height: 20.0),
TextFormField(
  onChanged: (val) {
    setState(() {
      money = val;
    });
  },
  controller: _moneyController,
  decoration: InputDecoration(
    labelText: 'Rental Amount for Each Person(RM)',
    prefixIcon: Icon(Icons.attach_money),
    border: OutlineInputBorder(),
  ),
),
const SizedBox(height: 20.0),
TextFormField(
  onChanged: (val) {
    setState(() {
      usercaller = val;
    });
  },
  controller: _usercallerController,
  decoration: InputDecoration(
    labelText: 'Posted by',
    prefixIcon: Icon(Icons.verified_user_outlined),
    border: OutlineInputBorder(),
  ),
),
const SizedBox(height: 20.0),
TextFormField(
  onChanged: (val) {
    setState(() {

```

```

        phoneno = val;
    });
},
controller: _phonenoController,
decoration: InputDecoration(
    labelText: 'Phone Number',
    prefixIcon: Icon(Icons.contact_phone_outlined),
    border: OutlineInputBorder(),
),
),
SizedBox(height: 40.0,),
myBtn(context),
],
),
)
);
}

```

```

OutlinedButton myBtn(BuildContext context) {
    return OutlinedButton(
        style: OutlinedButton.styleFrom(minimumSize: const Size(200, 50)),
        onPressed: (){
            _postService.savePost(house, address, amount, money, phoneno,
usercaller);
            Navigator.push(
                context,
                MaterialPageRoute(builder: (context){
                    return ShootOut();
                })
            );
        },
        child: Text(
            "Submit Form".toUpperCase(),

```

```

        style: const TextStyle(fontWeight: FontWeight.bold),
      ),
    );
  }
}

```

Shootout.dart

```

import 'package:flutter/material.dart';
import 'package:housemateheaven/pages_student/rentalform.dart';
import 'package:housemateheaven/pages_student/post.dart';
import 'list.dart';
import '../widget/navdrawer.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:provider/provider.dart';
import 'package:flutter/material.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:provider/provider.dart';
import 'package:housemateheaven/post_model.dart';

class ShootOut extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    final postService = PostService();
    final user = FirebaseAuth.instance.currentUser;

    return StreamProvider<List<PostModel>>.value(
      value: postService.getPostsByUser(user?.uid ?? ''),
      initData: [], // Provide initial empty list
      child: Scaffold(
        drawer: NavDrawer(), // Make sure you have implemented NavDrawer

```

correctly

```
    appBar: AppBar(
      backgroundColor: Colors.black,
      title: Text('ShootOut'),
    ),
    body: Column(
      children: [
        Expanded(
          child: ListPost(),
        ),
      ],
    ),
    floatingActionButton: FloatingActionButton(
      onPressed: () {
        Navigator.push(
          context,
          MaterialPageRoute(builder: (context) {
            return RentalForm();
          }),
        );
      },
      child: Icon(Icons.add),
      backgroundColor: Colors.black,
      foregroundColor: Colors.yellow,
      mini: true,
    ),
    floatingActionButtonLocation:
FloatingActionButtonLocation.centerDocked,
    bottomNavigationBar: BottomAppBar(
      color: Colors.black,
      shape: CircularNotchedRectangle(),
      child: Row(
        mainAxisAlignment: MainAxisAlignment.max,
```

```

        mainAxisAlignment: MainAxisAlignment.spaceBetween,
        children: <Widget>[
            IconButton(icon: Icon(Icons.menu), onPressed: () {}),
            IconButton(icon: Icon(Icons.search), onPressed: () {}),
        ],
    ),
),
),
);
}
}

```

Studentprofile.dart

```

import 'package:flutter/material.dart';
import '../widget/navdrawer.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:housemateheaven/text_box.dart';

```

```

class StudentProfile extends StatefulWidget {
  const StudentProfile({Key? key}) : super(key: key);

```

```

  @override
  State<StudentProfile> createState() => _StudentProfileState();
}

```

```

class _StudentProfileState extends State<StudentProfile> {
  final currentUser = FirebaseAuth.instance.currentUser;
  final usersCollection = FirebaseFirestore.instance.collection("users");

```

```

Future<void> editField(String field) async {
  String newValue = "";
  await showDialog(
    context: context,
    builder: (context) => AlertDialog(
      backgroundColor: Colors.grey[900],
      title: Text(
        "Edit $field",
        style: const TextStyle(color: Colors.white),
      ),
      content: TextField(
        autofocus: true,
        style: TextStyle(color: Colors.white),
        decoration: InputDecoration(
          hintText: "Enter new $field",
          hintStyle: TextStyle(color: Colors.grey),
        ),
        onChanged: (value) {
          newValue = value;
        },
      ),
      actions: [
        TextButton(
          child: Text(
            "Cancel",
            style: TextStyle(color: Colors.white),
          ),
          onPressed: () => Navigator.pop(context),
        ),
        TextButton(
          child: Text(
            "Save",
            style: TextStyle(color: Colors.white),

```



```

        ),
        onPressed: () => Navigator.of(context).pop(newValue),
      ),
    ],
  ),
);

if (newValue.trim().length > 0) {
  await usersCollection.doc(currentUser!.uid).update({field: newValue});
}
}

```

```

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      backgroundColor: Colors.black,
      title: Text('Student Profile'),
    ),
    body: FutureBuilder<DocumentSnapshot>(
      future: usersCollection.doc(currentUser!.uid).get(),
      builder: (context, snapshot) {
        if (snapshot.connectionState == ConnectionState.waiting) {
          return Center(child: CircularProgressIndicator());
        } else if (snapshot.hasError) {
          return Center(child: Text('Error: ${snapshot.error}'));
        }
        else{

          final userData = snapshot.data!.data() as Map<String, dynamic>?;

```

```

return ListView(
  children: [
    const SizedBox(height: 50),
    Icon(
      Icons.person,
      size: 72,
    ),
    Text(
      currentUser?.email ?? 'N/A',
      textAlign: TextAlign.center,
      style: TextStyle(color: Colors.grey[700]),
    ),
    Padding(
      padding: const EdgeInsets.only(left: 25.0),
      child: Text(
        " Details ",
        style: TextStyle(color: Colors.grey[600]),
      ),
    ),
    MyTextBox(
      text: userData?['username'] ?? "",
      sectionName: 'username',
      onPressed: () => editField('username'),
    ),
    MyTextBox(
      text: userData?['full name'] ?? "",
      sectionName: 'full name',
      onPressed: () => editField('full name'),
    ),
    MyTextBox(
      text: userData?['password'] ?? "",
      sectionName: 'password',
      onPressed: () => editField('password'),
    ),
  ],
);

```

```

    ),
    MyTextBox(
      text: userData?['university name'] ?? "",
      sectionName: 'university name',
      onPressed: () => editField('university name'),
    ),
    MyTextBox(
      text: userData?['university id'] ?? "",
      sectionName: 'university id',
      onPressed: () => editField('university id'),
    ),
  ],
);
}
},
),
);
}
}

```

Favourite_provider.dart

```
import 'package:flutter/material.dart';
```

```
import '../post_model.dart';
```

```
class FavoritePostsProvider extends ChangeNotifier {
```

```
  List<PostModel> favoritePosts = [];
```

```
  List<String> bookmarkedPostIds = [];
```

```
  void toggleBookmark(PostModel post) {
```

```
    if (bookmarkedPostIds.contains(post.id)) {
```

```
      bookmarkedPostIds.remove(post.id);
```

```
      favoritePosts.removeWhere((favPost) => favPost.id == post.id);
```

```

    } else {
      bookmarkedPostIds.add(post.id);
      favoritePosts.add(post);
    }
    notifyListeners();
  }

  void addFavoritePost(PostModel post) {
    bookmarkedPostIds.add(post.id);
    favoritePosts.add(post);
    notifyListeners();
  }

  void removeFavoritePost(PostModel post) {
    bookmarkedPostIds.remove(post.id);
    favoritePosts.remove(post);
    notifyListeners();
  }
}

```

Favourite.dart

```

import 'package:flutter/material.dart';
import 'package:housemateheaven/widget/navdrawer2.dart';
import '../post_model.dart';
import '../widget/navdrawer2.dart';
import 'favourite_provider.dart';
import 'package:provider/provider.dart';

class Favourite extends StatelessWidget {
  const Favourite({Key? key, required List favoritePosts}) : super(key: key);

  @override

```

```

Widget build(BuildContext context) {
  final favoritePostsProvider = Provider.of<FavoritePostsProvider>(context);
  final favoritePosts = favoritePostsProvider.favoritePosts;

  return Scaffold(
    drawer: NavDrawer2(),
    appBar: AppBar(
      backgroundColor: Colors.black,
      title: Text('Favourite'),
    ),
    body: ListView.builder(
      itemCount: favoritePosts.length,
      itemBuilder: (context, index) {
        final post = favoritePosts[index];

        return Padding(
          padding: EdgeInsets.symmetric(horizontal: 16, vertical: 8),
          child: Card(
            elevation: 2,
            child: ListTile(
              contentPadding: EdgeInsets.symmetric(vertical: 12, horizontal: 16),
              title: Text(
                post.house,
                style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),
              ),
              subtitle: Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                  SizedBox(height: 8),
                  Text(
                    'Address: ${post.address}',
                    style: TextStyle(fontSize: 14),
                  ),
                ],
              ),
            ),
          ),
        );
      },
    ),
  );
}

```

```

        SizedBox(height: 4),
        Text(
          'Amount Housemate Needed: ${post.amount}',
          style: TextStyle(fontSize: 14),
        ),
        SizedBox(height: 4),
        Text(
          'Expected Rental Amount(RM): ${post.money}',
          style: TextStyle(fontSize: 14),
        ),
        SizedBox(height: 4),
        Text(
          'Posted by: ${post.usercaller}',
          style: TextStyle(fontSize: 14),
        ),
        SizedBox(height: 4),
        Text(
          'Phone Number: ${post.phoneno}',
          style: TextStyle(fontSize: 14),
        ),
      ],
    ),
  ),
);
},
),
);
}
}
}

```

Homepage2.dart

```
import 'package:flutter/material.dart';
```

```
import 'package:introduction_screen/introduction_screen.dart';
import '../widget/navdrawer2.dart';
```

```
class HomePage2 extends StatefulWidget {
  const HomePage2({
    Key? key,
  }) : super(key: key);

  @override
  _HomePage2State createState() => _HomePage2State();
}
```

```
class _HomePage2State extends State<HomePage2> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      drawer: NavDrawer2(),
      appBar: AppBar(
        backgroundColor: Colors.black,
        title: Text('HousemateHeaven'),
      ),
      body: PageView(
        children: [
          Center(
            child: Container(
              child: Column(
                mainAxisAlignment: MainAxisAlignment.center,
                children: [
                  Image.asset('assests/image/logowhite.jpeg'),
                  SizedBox(height: 40),
                  Text(
                    'Welcome to HousemateHeaven',
                    style: TextStyle(
```

```

        color: Colors.black,
        fontSize: 20,
        fontWeight: FontWeight.bold,
      ),
      textAlign: TextAlign.center,
    ),
    Text(
      'This is a tutorial if you selected Looking For A House To Share
role.',

      style: TextStyle(
        color: Colors.black,
        fontSize: 16,
      ),
      textAlign: TextAlign.center,
    ),
    SizedBox(height: 10),
    Text(
      'Swipe right for more.',
      style: TextStyle(
        color: Colors.black,
        fontSize: 16,
        fontWeight: FontWeight.bold,
      ),
      textAlign: TextAlign.center,
    ),
  ],
),
),
),
Center(
  child: Container(
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,

```



```

children: [
  Image.asset('assests/image/sections2.jpeg'),
  SizedBox(height: 20),
  Text(
    'There are six sections and you need to press section that you
want to use.',
    style: TextStyle(
      color: Colors.black,
      fontSize: 16,
    ),
    textAlign: TextAlign.center,
  ),
  SizedBox(height: 10),
  Text(
    'Swipe right for more.',
    style: TextStyle(
      color: Colors.black,
      fontSize: 16,
      fontWeight: FontWeight.bold,
    ),
    textAlign: TextAlign.center,
  ),
],
),
),
),
Center(
  child: Container(
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Text(
          'ShootOut:',

```

```

        style: TextStyle(
          color: Colors.black,
          fontSize: 16,
          fontWeight: FontWeight.bold,
        ),
        textAlign: TextAlign.center,
      ),
      SizedBox(height: 40),
      Image.asset('assets/image/shootout.jpeg'),
      SizedBox(height: 40),
      Text(
        'In the shootout section, you can view form that created by other
users. You also can bookmark form that you want here.',
        style: TextStyle(
          color: Colors.black,
          fontSize: 16,
        ),
        textAlign: TextAlign.center,
      ),
      SizedBox(height: 10),
      Text(
        'Swipe right for more.',
        style: TextStyle(
          color: Colors.black,
          fontSize: 16,
          fontWeight: FontWeight.bold,
        ),
        textAlign: TextAlign.center,
      ),
    ],
  ),
),
),
),
),

```

```

Center(
  child: Container(
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Text(
          'Search:',
          style: TextStyle(
            color: Colors.black,
            fontSize: 16,
            fontWeight: FontWeight.bold,
          ),
          textAlign: TextAlign.center,
        ),
        SizedBox(height: 40),
        Image.asset('assets/image/search.jpeg'),
        SizedBox(height: 40),
        Text(
          'In the search section, you can search for form that created by
other users.',
          style: TextStyle(
            color: Colors.black,
            fontSize: 16,
          ),
          textAlign: TextAlign.center,
        ),
        SizedBox(height: 10),
        Text(
          'Swipe right for more.',
          style: TextStyle(
            color: Colors.black,
            fontSize: 16,
            fontWeight: FontWeight.bold,

```

```

        ),
        textAlign: TextAlign.center,
    ),
],
),
),
),
Center(
  child: Container(
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Text(
          'Favourite:',
          style: TextStyle(
            color: Colors.black,
            fontSize: 16,
            fontWeight: FontWeight.bold,
          ),
          textAlign: TextAlign.center,
        ),
        SizedBox(height: 40),
        Image.asset('assests/image/favourite.jpeg'),
        SizedBox(height: 40),
        Text(
          'In the favourite section, you can view form that has been
bookmarked from shootout section.',
          style: TextStyle(
            color: Colors.black,
            fontSize: 16,
          ),
          textAlign: TextAlign.center,
        ),

```

```

        SizedBox(height: 10),
        Text(
          'Swipe right for more.',
          style: TextStyle(
            color: Colors.black,
            fontSize: 16,
            fontWeight: FontWeight.bold,
          ),
          textAlign: TextAlign.center,
        ),
      ],
    ),
  ),
),
Center(
  child: Container(
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Text(
          'Profile:',
          style: TextStyle(
            color: Colors.black,
            fontSize: 16,
            fontWeight: FontWeight.bold,
          ),
          textAlign: TextAlign.center,
        ),
        SizedBox(height: 40),
        Image.asset('assests/image/profile.jpeg'),
        SizedBox(height: 40),
        Text(
          'In the profile section, you can view details that you registered

```

earlier. You can also edit any detail that you want',

```
        style: TextStyle(
          color: Colors.black,
          fontSize: 16,
        ),
        textAlign: TextAlign.center,
      ),
      SizedBox(height: 10),
      Text(
        'Swipe right for more.',
        style: TextStyle(
          color: Colors.black,
          fontSize: 16,
          fontWeight: FontWeight.bold,
        ),
        textAlign: TextAlign.center,
      ),
    ],
  ),
),
Center(
  child: Container(
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Text(
          'Logout:',
          style: TextStyle(
            color: Colors.black,
            fontSize: 16,
            fontWeight: FontWeight.bold,
          ),
        ),
```

```

        textAlign: TextAlign.center,
      ),
      SizedBox(height: 40),
      Image.asset('assests/image/logout.jpeg'),
      SizedBox(height: 40),
      Text(
        'In the logout section, you can logout from the application by pressing it.',
        style: TextStyle(
          color: Colors.black,
          fontSize: 16,
        ),
        textAlign: TextAlign.center,
      ),
      SizedBox(height: 10),
      Text(
        'That is all for tutorial on how to use the application. Hope you will have a great time with HousemateHeaven!!!',
        style: TextStyle(
          color: Colors.black,
          fontSize: 16,
          fontWeight: FontWeight.bold,
        ),
        textAlign: TextAlign.center,
      ),
    ],
  ),
),
],
),
);
}

```

```
}
```

List2.dart

```
import 'package:flutter/material.dart';
import 'package:housemateheaven/pages_student/edit_post.dart';
import 'package:housemateheaven/pages_studentrent/post2.dart';
import 'package:housemateheaven/pages_studentrent/post_details.dart';
import 'package:housemateheaven/pages_student/post.dart';
import 'package:housemateheaven/post_model.dart';
import 'package:provider/provider.dart';

import 'favourite_provider.dart';

class ListPost2 extends StatefulWidget {
  const ListPost2({Key? key}) : super(key: key);

  @override
  State<ListPost2> createState() => _ListPostState();
}

class _ListPostState extends State<ListPost2> {
  PostService2 _postService = PostService2();
  List<String> bookmarkedPostIds = []; // List to store the IDs of bookmarked
posts

  @override
  Widget build(BuildContext context) {
    final posts = Provider.of<List<PostModel>>(context);
    final favoritePostsProvider = Provider.of<FavoritePostsProvider>(context);

    bookmarkedPostIds = favoritePostsProvider.bookmarkedPostIds;

    return ListView.builder(
```



```

itemCount: posts.length,
itemBuilder: (context, index) {
    final post = posts[index];
    final isBookmarked = bookmarkedPostIds.contains(post.id); // Check if
the post is bookmarked

```

```

return Padding(
    padding: EdgeInsets.symmetric(horizontal: 16, vertical: 8),
    child: Card(
        elevation: 2,
        child: ListTile(
            contentPadding: EdgeInsets.symmetric(vertical: 12, horizontal: 16),
            title: Text(
                post.house,
                style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold),
            ),
            subtitle: Column(
                crossAxisAlignment: CrossAxisAlignment.start,
                children: [
                    SizedBox(height: 8),
                    Text(
                        'Address: ${post.address}',
                        style: TextStyle(fontSize: 14),
                    ),
                    SizedBox(height: 4),
                    Text(
                        'Amount Housemate Needed: ${post.amount}',
                        style: TextStyle(fontSize: 14),
                    ),
                    SizedBox(height: 4),
                    Text(
                        'Expected Rental Amount(RM): ${post.money}',
                        style: TextStyle(fontSize: 14),

```

```

    ),
    SizedBox(height: 4),
    Text(
      'Posted by: ${post.usercaller}',
      style: TextStyle(fontSize: 14),
    ),
    SizedBox(height: 4),
    Text(
      'Phone Number: ${post.phoneno}',
      style: TextStyle(fontSize: 14),
    ),
  ],
),
trailing: Row(
  mainAxisAlignment: MainAxisAlignment.min,
  children: [
    IconButton(
      icon: Icon(
        isBookmarked ? Icons.bookmark : Icons.bookmark_border,
        color: isBookmarked ? Colors.orange : null,
      ),
      onPressed: () {
        favoritePostsProvider.toggleBookmark(post);
      },
    ),
  ],
),
),
),
);
},
);
}

```

```
}
```

Post_details.dart

```
import 'package:flutter/material.dart';
import 'package:housemateheaven/post_model.dart';

class PostDetails extends StatelessWidget {
  final PostModel post;

  const PostDetails({Key? key, required this.post}) : super(key: key);

  @override
  Widget build(BuildContext context) {
    // Implement the UI for displaying the post details
    return Scaffold(
      appBar: AppBar(
        title: Text('Post Details'),
      ),
      body: Center(
        child: Column(
          mainAxisAlignment: MainAxisAlignment.center,
          children: [
            Text('Post ID: ${post.id}'),
            Text('House: ${post.house}'),
            Text('Address: ${post.address}'),
            Text('Amount: ${post.amount}'),
            Text('Money: ${post.money}'),
            Text('Phone Number: ${post.phoneno}'),
            Text('Name: ${post.usercaller}'),
          ],
        ),
      ),
    );
  }
}
```

```
}  
}
```

Post2.dart

```
import 'package:cloud_firestore/cloud_firestore.dart';  
import 'package:firebase_core/firebase_core.dart';  
import 'package:firebase_auth/firebase_auth.dart';  
import 'package:housemateheaven/post_model.dart';
```

```
class PostService2 {  
  List<PostModel> _postListFromSnapshot(QuerySnapshot snapshot) {  
    return snapshot.docs.map((doc) {  
      return PostModel(  
        id: doc.id,  
        house: doc['house'] ?? "",  
        address: doc['address'] ?? "",  
        amount: doc['amount'] ?? "",  
        money: doc['money'] ?? "",  
        phoneno: doc['phone no'] ?? "",  
        usercaller: doc['user caller'] ?? "",  
        user: doc['user'] ?? "",  
      );  
    }).toList();  
  }  
}
```

```
Future<void> savePost(String house, String address, String amount, String  
money, String phoneno, String usercaller) async {  
  User? currentUser = FirebaseAuth.instance.currentUser;  
  if (currentUser != null) {  
    // Get the current user's UID  
    String uid = currentUser.uid;  
  
    // Create a new document reference in the "post" collection
```

```

        DatabaseReference          postRef          =
        FirebaseFirestore.instance.collection("post").doc();

```

```

        // Create a new document reference in the "users" collection with the same
        ID as the post

```

```

        DatabaseReference          userRef          =
        FirebaseFirestore.instance.collection("users").doc(postRef.id);

```

```

        // Batch write to update both collections atomically
        WriteBatch batch = FirebaseFirestore.instance.batch();

```

```

        // Set the user document with the post ID
        batch.set(userRef, {});

```

```

        // Set the post document with the user ID
        batch.set(postRef, {
            'house': house,
            'address': address,
            'amount': amount,
            'money': money,
            'phone no': phoneno,
            'user caller': usercaller,
            'user': uid,
        });

```

```

        // Commit the batch write
        await batch.commit();
    }
}

```

```

Stream<List<PostModel>> getPostsByUser(String uid) {
    return FirebaseFirestore.instance
        .collection("post")

```

```

        .snapshots()
        .map(_postListFromSnapshot);
    }

    Future<PostModel> getPostById(String postId) async {
        DocumentSnapshot snapshot = await
        FirebaseFirestore.instance.collection('post').doc(postId).get();
        return PostModel(
            id: snapshot.id,
            house: snapshot['house'] ?? "",
            address: snapshot['address'] ?? "",
            amount: snapshot['amount'] ?? "",
            money: snapshot['money'] ?? "",
            phoneno: snapshot['phone no'] ?? "",
            usercaller: snapshot['user caller'] ?? "",
            user: snapshot['user'] ?? "",
        );
    }
}

```

Search2.dart

```

import 'package:flutter/material.dart';
import 'package:housemateheaven/widget/navdrawer2.dart';
import '../widget/navdrawer.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:cloud_firestore/cloud_firestore.dart';

class Search2 extends StatefulWidget {
    const Search2({Key? key}) : super(key: key);

    @override

```

```

    State<Search2> createState() => _Search2State();
}

class _Search2State extends State<Search2> {
    final TextEditingController _searchController = TextEditingController();
    List<DocumentSnapshot> _searchResults = [];

    void _performSearch(String query) {
        String startRange = query;
        String endRange = query + '\uf8ff';

        FirebaseFirestore.instance
            .collection('post')
            .where('house', isGreaterThanOrEqualTo: startRange)
            .where('house', isLessThan: endRange)
            .get()
            .then((QuerySnapshot snapshot) {
                setState(() {
                    _searchResults = snapshot.docs;
                });
            }).catchError((error) {
                print('Error performing search: $error');
            });
    }

    @override
    Widget build(BuildContext context) {
        return Scaffold(
            drawer: NavDrawer2(),
            appBar: AppBar(
                backgroundColor: Colors.black,
                title: Text('Search'),
            ),
        ),
    }
}

```

```

body: Column(
  children: [
    TextField(
      controller: _searchController,
      decoration: InputDecoration(
        hintText: 'Enter house name',
      ),
    ),
    ElevatedButton(
      onPressed: () {
        String searchQuery = _searchController.text;
        _performSearch(searchQuery);
      },
      child: Text('Search'),
    ),
    Expanded(
      child: ListView.builder(
        itemCount: _searchResults.length,
        itemBuilder: (context, index) {
          DocumentSnapshot postSnapshot = _searchResults[index];

          // Extract post data from the snapshot
          String house = postSnapshot['house'];
          String address = postSnapshot['address'];
          String amount = postSnapshot['amount'];
          String money = postSnapshot['money'];
          String phoneno = postSnapshot['phone no'];
          String usercaller = postSnapshot['user caller'];

          // Display the post data in a Card
          return Card(
            child: ListTile(
              title: Text(house),

```



```

        subtitle: Column(
          crossAxisAlignment: CrossAxisAlignment.start,
          children: [
            Text('Address: $address'),
            Text('Amount Housemate Needed: $amount'),
            Text('Expected Rental Amount(RM): $money'),
            Text('Posted by: $usercaller'),
            Text('Phone Number: $phoneno'),
          ],
        ),
      ),
    );
  },
),
],
),
),
),
);
}
}

```

Shootout2.dart

```

import 'package:flutter/material.dart';
import 'package:housemateheaven/pages_student/rentalform.dart';
import 'package:housemateheaven/pages_studentrent/list2.dart';
import 'package:housemateheaven/pages_studentrent/post2.dart';
import 'package:housemateheaven/pages_student/post.dart';
import 'package:housemateheaven/widget/navdrawer2.dart';
import '../pages_student/list.dart';
import '../widget/navdrawer.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:cloud_firestore/cloud_firestore.dart';

```

```

import 'package:provider/provider.dart';
import 'package:flutter/material.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:cloud_firestore/cloud_firestore.dart';
import 'package:provider/provider.dart';
import 'package:housemateheaven/post_model.dart';

class ShootOut2 extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    final postService = PostService2();

    return StreamProvider<List<PostModel>>.value(
      value:
postService.getPostsByUser(FirebaseAuth.instance.currentUser?.uid ?? ''),
      initialData: [],
      child: Scaffold(
        drawer: NavDrawer2(), // Make sure you have implemented NavDrawer
correctly
        appBar: AppBar(
          backgroundColor: Colors.black,
          title: Text('ShootOut'),
        ),
        body: Column(
          children: [
            Expanded(
              child: ListPost2(),
            ),
          ],
        ),
        floatingActionButtonLocation:
FloatingActionButtonLocation.centerDocked,
        bottomNavigationBar: BottomAppBar(

```

```

        color: Colors.black,
        shape: CircularNotchedRectangle(),
        child: Row(
          mainAxisAlignment: MainAxisAlignment.max,
          mainAxisSize: MainAxisSize.spaceBetween,
          children: <Widget>[
            IconButton(icon: Icon(Icons.menu), onPressed: () {}),
            IconButton(icon: Icon(Icons.search), onPressed: () {}),
          ],
        ),
      ),
    ),
  );
}
}

```

Studentprofile2.dart

```

import 'package:flutter/material.dart';
import 'package:housemateheaven/text_box.dart';
import 'package:housemateheaven/widget/navdrawer2.dart';
import 'package:firebase_core/firebase_core.dart';
import 'package:firebase_auth/firebase_auth.dart';
import 'package:cloud_firestore/cloud_firestore.dart';

class StudentProfile2 extends StatefulWidget {
  const StudentProfile2({Key? key}) : super(key: key);

  @override
  State<StudentProfile2> createState() => _StudentProfile2State();
}

class _StudentProfile2State extends State<StudentProfile2> {
  final currentUser = FirebaseAuth.instance.currentUser;

```

```
final usersCollection = FirebaseFirestore.instance.collection("users");
```

```
String username = "";  
String fullName = "";  
String password = "";  
String universityName = "";  
String universityId = "";
```

```
@override  
void initState() {  
  super.initState();  
  fetchData();  
}
```

```
Future<void> fetchData() async {  
  final snapshot = await usersCollection.doc(currentUser!.uid).get();  
  final userData = snapshot.data() as Map<String, dynamic>;
```

```
  if (userData != null) {  
    setState(() {  
      username = userData['username'] ?? "";  
      fullName = userData['full name'] ?? "";  
      password = userData['password'] ?? "";  
      universityName = userData['university name'] ?? "";  
      universityId = userData['university id'] ?? "";  
    });  
  }  
}
```

```
Future<void> editField(String field) async {  
  String newValue = "";  
  await showDialog(  
    context: context,
```

```

builder: (context) => AlertDialog(
  backgroundColor: Colors.grey[900],
  title: Text(
    "Edit $field",
    style: const TextStyle(color: Colors.white),
  ),
  content: TextField(
    autofocus: true,
    style: TextStyle(color: Colors.white),
    decoration: InputDecoration(
      hintText: "Enter new $field",
      hintStyle: TextStyle(color: Colors.grey),
    ),
    onChanged: (value) {
      newValue = value;
    },
  ),
  actions: [
    TextButton(
      child: Text(
        "Cancel",
        style: TextStyle(color: Colors.white),
      ),
      onPressed: () => Navigator.pop(context),
    ),
    TextButton(
      child: Text(
        "Save",
        style: TextStyle(color: Colors.white),
      ),
      onPressed: () => Navigator.of(context).pop(newValue),
    ),
  ],

```

```

    ),
  );

  if (newValue.trim().length > 0) {
    await usersCollection.doc(currentUser!.uid).update({field: newValue});
  }
}

```

```

@override
Widget build(BuildContext context) {
  return Scaffold(
    appBar: AppBar(
      backgroundColor: Colors.black,
      title: Text('Student Profile'),
    ),
    body: FutureBuilder<DocumentSnapshot>(
      future: usersCollection.doc(currentUser!.uid).get(),
      builder: (context, snapshot) {
        if (snapshot.connectionState == ConnectionState.waiting) {
          return Center(child: CircularProgressIndicator());
        } else if (snapshot.hasError) {
          return Center(child: Text('Error: ${snapshot.error}'));
        }
        else{

          final userData = snapshot.data!.data() as Map<String, dynamic>?;

          return ListView(
            children: [
              const SizedBox(height: 50),
              Icon(
                Icons.person,
                size: 72,

```

```

    ),
    Text(
      currentUser?.email ?? 'N/A',
      textAlign: TextAlign.center,
      style: TextStyle(color: Colors.grey[700]),
    ),
    Padding(
      padding: const EdgeInsets.only(left: 25.0),
      child: Text(
        " Details ",
        style: TextStyle(color: Colors.grey[600]),
      ),
    ),
  ),

```

```

MyTextBox(
  text: username,
  sectionName: 'username',
  onPressed: () => editField('username'),
),

```

```

MyTextBox(
  text: fullName,
  sectionName: 'full name',
  onPressed: () => editField('full name'),
),

```

```

MyTextBox(
  text: password,
  sectionName: 'password',
  onPressed: () => editField('password'),
),

```

```

MyTextBox(
  text: universityName,
  sectionName: 'university name',
  onPressed: () => editField('university name'),
),

```

```
    ),  
    MyTextBox(  
      text: universityId,  
      sectionName: 'university id',  
      onPressed: () => editField('university id'),  
    ),  
  ],  
);  
}  
},  
),  
);  
}  
}
```