6COS023W – Final Project Report

University of Westminster Social Platform Mobile Application for Students

UoWsa

UOWSA

EST.2022

Student: Christopher Wong

Supervisor: Markos Mentzelopoulos

This report is submitted in partial fulfilment
of the requirements for the
BSc (Hons) Computer Science degree
at the University of Westminster

School of Computer Science & Engineering
University of Westminster

Date: 18/04/2023

Declaration

This report has been prepared based on my own work. Where other published and unpublished source materials have been used, these have been acknowledged in references.

Word Count: 50058
Student Name: Christopher Wong
Date of Submission: 02/05/2023

Abstract

University of Westminster Social Application (UoWsa) is a mobile application created in Kotlin Android Operating System platform for the BSc (hons) Computer Science Final Year Project. This project demonstrates a passionate idea to design and create a functional mobile software/application to solve the complexity and dilemma of students finding communication difficult with their peers.

UoWsa is a social media mobile application, built to target University of Westminster students using a modern socialising platform to provide a safe space for students to communicate and interact with each other on various categories related to their academic and social lives. This application includes several features such as, support from the Student Union and Student Wellbeing, account profile, direct messaging, group messaging and forums. The purpose of the development is to support and expand interactions between students and also provide aid to students' mental health during and the aftermath of the COVID-19 pandemic.

The opportunity given in the Final Year Project combines all of the tools, skills and techniques learnt using the resources provided by the University of Westminster as well as independent learning. This report outlines the details of the development process, from the planning and design stages, implementation and challenges encountered, as well as the testing phase. There are also application statistical graphs in place to evaluate the frequency of interaction among students.

Acknowledgements

Firstly, I would like to express my deep appreciation to my supervisor, Markos Mentzelopoulos for his guidance, support, encouragement and inspiration throughout the planning and development of this project. His expertise, insights and feedbacks are invaluable, which has contributed massively through my questions and action processing throughout the obstacles and dilemma I've had.

The Student Union team and its societies provided many insights on the processes and requirements to expand its society to a larger scale. The advice, resources and advocacy has been integral part of the creation in the Student Union and Clubs and Societies feature of the application. I would like to also extend my appreciation to all students and past students who participated in the questionnaires in planning ideas and testing. Your responses were critical in gathering data to develop and test the front-end designs and features implemented in the project.

I would also like to thank all the University of Westminster staff, lecturers, tutors, and faculty members who provided me the resources, facility, and the opportunity to produce this project. Their support has been essential for me to complete this project successfully.

Finally, I am grateful for my friends and family motivating me throughout the 7 months of hard work and dedication in crafting and developing the project into the outcomes as it stands today. This support also relates to the downfall of contracting Covid and maintaining the desire to continue developing the project throughout the holidays.

Table of Contents

Contents

List of Figures

List of Tables

1. Introduction

The COVID-19 pandemic and its aftermath have a significant impact on the mental health and wellbeing of university student. The critical challenges faced by students is the difficulty finding peers with common interests and engaging with them. In order to solve and address this issue, a proposal was developed to create and implement a social mobile application that will enable students to communicate with each other along with pre-existing services such as Student Union and Student wellbeing. The aim of this project is to improve the social aspects of university life and partially aid students' mental health during a worldwide pandemic.

The current availability of social networking for students are incredibly limited, and many developed with the primary intention for educational purposes. High profile social media platforms such as, Facebook became extremely popular among students for socialising, however they lack features based on their fields of study and interests. In addition, the privacy and ethical concerns associated with these platforms makes them less suitable for academic environments.

The proposed social media mobile application platform will be developed specifically for University of Westminster students, and it provides a social space where students can connect and engage with each other. The main goal will allow students to communicate through direct messaging with other students and provide information relating to assist students to finding new group activities in Clubs and Societies, which is pre-existing from the Student Union. This project also aims to tackle the problems in recommending ideas for students who wish to join existing society groups and sports teams provided by the university.

The project's objectives require research in the most appropriate Kotlin methods and structures to create the mobile application, time management on a clear plan with deadlines, establishing a successful connection to the database, and researching appropriate software development kits (SDKs) for the project. The project will also produce throwaway prototypes prior to the final version and conducting initial questionnaires and testing questionnaires after implementation to gain feedbacks for user experience in the prototype created.

The mobile application aims to increase user-friendliness by finding friends with similar interests and study groups by implementing a forum and group chat for communication among students. The application will comply with General Data Protection Regulation (GDPR) and address ethical concerns with admin users and other development methods in place. Additionally, the project aims to expand to teachers and staff from the university, timetables, maps and many more features in future updates.

1.1 Problem statement

The dilemma of finding university students with common interest or simply engaging with students could be a difficult task, therefore providing help with a social mobile platform can partially aid students' mental health during a worldwide pandemic. Many students find it difficult to communicate and find other students (*Borup, Walters and Call-Cummings, 2020*), thus the application will also introduce other pre-existing services such as, students wellbeing, finding contact information for specific help and student's union along with their societies and sports clubs. Furthermore, the complexity and complication of social networking amongst students are extremely limited as the vast social platforms for students primarily are educational purposes such as Blackboard, EDU 2.0, 30hands Learning Network and many more (*Berry, 2021*).

In the past, students communicate through emails, however the generic public social media platforms including Facebook became an exceptional approach to socialise online. Despite the mass amount of involvement on Facebook, there are limitations as the social platform is developed with the intention of generalised social networking, therefore there is a space in the market specialising in students to develop their skills with their peers. Furthermore, there is a connection block in accessibility, ethical measures, and privacy, which researchers have discovered and should be eliminated (*Ukwishaka and Aghaee, 2020*).

The principle of this project is to connect the University of Westminster students together by creating a social space, this mobile application will allow engagement amongst students by direct messaging and provide information regarding a connection with other students in addition to other pre-existing methods including, the newly implemented student email on discord hub to find existing servers, student union, clubs, and societies. By creating this mobile application, I hope to prove my concept and findings where this could be a great example for students improving the social aspects for students to help each other without the use of formalities. Subsequently, a survey will be put out to find out what are student's needs and the effectiveness of communicating among students currently. The project aims to tackle difficulties of communicating among students directly and recommending ideas to students who wish to join clubs and societies provided by the university. The social mobile platform for the university intends to increase user friendliness by finding friends easier with similar interests, fields of studying and search through the application.

Note: Please refer to Appendix I for the current representation of Blackboard discussion board and Discord hub images.

1.2 Aims and Objectives

Aims:

- The project can deliver an android mobile application which is user friendly and able to accommodate multiple users concurrently using forums, direct messaging. This aspect of the project would be similar to the highly successful mobile apps in the market including Whatsapp and Discord.
- Student mobile application will also have the capacity for participating students to forward their interest to existing clubs and societies from the student union.
- The sign in process registers the user with an email and password.
- The ability to connect student social aspects to an acceptable degree using a discussion board / forum.
- Comply with the General Data Protection Regulation (GDPR).
- Although the application can be informal among students, admins users have the ability to tackle unethical concerns manually.
- Implement ideas for future updates such as expanding to teachers and admin staff from the university, timetables, access to university map / blueprint and many more.

Objectives:

- Research on the most appropriate Kotlin methods and structure to creating the mobile application.
- Time Management on a clear plan with deadlines.
- Establish a successful connection to the database.
- Research on appropriate software development kit (SDK) for the project.
- Produce throwaway prototypes prior to the final version.
- Conduct market research using initial surveys and user experience different versions of prototypes.
- Create wireframes and mock-ups for a clear understanding when creating the mobile application.
- Finding the best form of security encryption for messaging service within the application.
- When creating the application, the use of terms of service complying with GDPR and addressing ethical concerns.

1.3 Android Mobile Application for students

Mobile applications have become an integral tool for university students providing a significant overall experience in accessibility, flexibility, productivity, and engagement (*Gikas, and Grant, 2013*). Accessibility refers to the availability in university resources, for example, Student Union, Clubs and Societies and Student Wellbeing. Flexibility provides the student user the application resources wherever they are located at any given time. Productivity of the application in allowing engagement of social and academic help or perhaps a reminder of upcoming events and activities. Lastly, Engagement, where the mobile application facilitates the communication in messaging and forums along with collaborations to enhance social communication with other users on the application.

Android mobile applications became extremely popular among students. The availability, diversity in choices and affordability are several reasons why many people own Android devices. One of the main reasons why students would choose Android over their competitors in iOS is their affordability (*Lazarela, and Jakimoski, 2017*). Generally, Android devices brings a wider selection of products with multiple brands to choose from, therefore the project development in Android as an initial choice would be more suitable especially with University of Westminster students as the target audience. Furthermore, Android is an open-source platform integrated with Google services, therefore the alignment in development and the ease of use for users is a key factor in consideration during the planning phase.

1.4 Thesis Breakdown

The chapters break down into all aspects related to each development process of the mobile application.

Chapter 1 introduces and documents a problem statement, where an analysis of an underlying problem and how it can be addressed along with the aims and objectives set for the project. This chapter also details the why it's beneficial to develop the project in Android for students.

Chapter 2 focuses on the background and literature survey of the project, where it's divided into subchapters including, Student Social Apps, University Student's Mental Health, Mobile User Experience as well as the advantages and disadvantages of Android or iOS and Non-Structured or Structured Database.

Chapter 3 explains the legal, ethical and security issues associated with the development project with data collection and analysis of the corresponding research. The subchapters will include Data gathering, messaging, Firebase Authentication / Realtime Database / Data storage, Testing and Deployment the Mobile application.

Chapter 4 introduces the life cycle of the project and methodology used. Moreover, it also introduces the process of gathering of requirements and how it relates to the project in the following chapters.

Chapter 5 illustrates all the design methods used for the project. The use of agile methodology and how it applies to the project under development.

Chapter 6 covers all of the implementation methods and skills acquired. This chapter also introduces the code and final outcome of each element of the project.

Chapter 7 provides a comprehensive amount of testing process for the developed application. All features, both implemented and not implemented, will be examined during testing. The chapter will also address the various testing methods that were employed, as well as any challenges that arose during the process. These challenges may have resulted in the development of desired and luxury features or caused features to be disregarded.

Chapter 8 provides a comprehensive summary of the project by evaluating the accomplishments, areas of improvement, and skills acquired during the development of the mobile application. It also highlights the limitations of the project and proposed recommendations for its future development based on the overall project outcomes.

Chapter 9 includes all of the references cited in text related to the report and project.

Chapter 10 Bibliography is the general reading list of items in relation to the project development.

Chapter 11 Appendix provides all of the additional contents throughout the research report .

2. Background

The background chapter in this report will cover the three main areas: literature survey, Review of projects / application and Review of tools, frameworks, and techniques. The literature survey will explain student social apps, University Student's mental health, mobile user experience, the choice between Android or iOS for students and the choice between a non-structured or structured database. Review of projects / applications provides details on the background research conducted such as existing projects and applications on the market. Review of tools, frameworks, and techniques will include the advantages and disadvantages of the resources and tools used for the project under development.

2.1 Literature Survey

The literature survey section will highlight the findings on various research topics related to the ongoing mobile application development. The topic in discussions will relate to the wellbeing of students and mental health as well as mobile user experience and the mobile operating system platform.

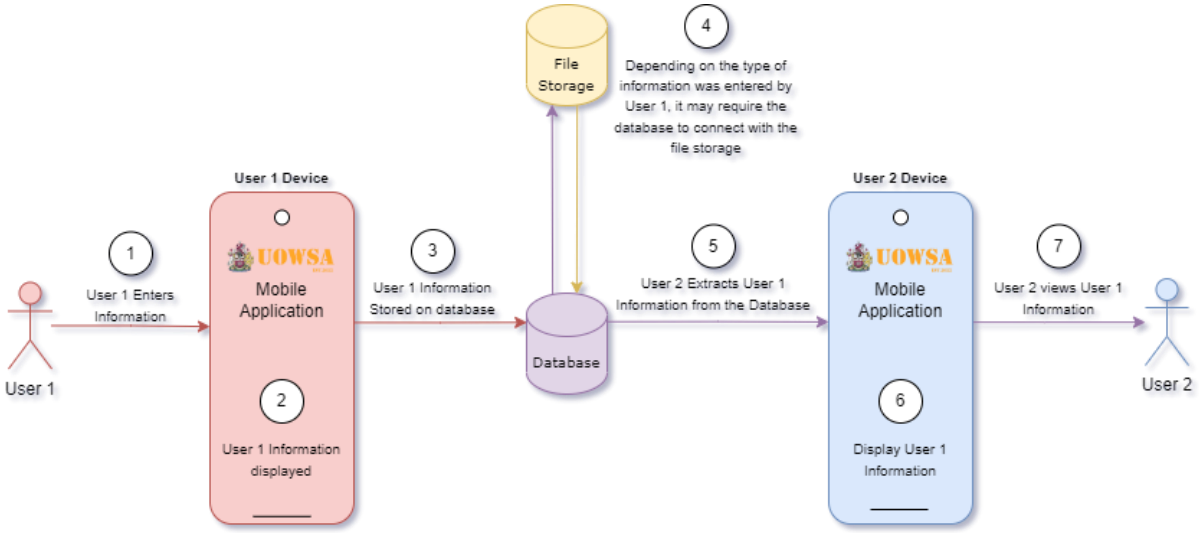


Figure 1 UoWsa Mobile Application Model

Steps	Description
1	User 1 Enters information on their mobile device and submits the information.
2	User 1 can view the information they have submitted previously.
3	User 1 information will be stored on the database.
4	If User 1 inserts information with a file type such as an image, the use of the file storage is required.
5	User 2 extracts User 1 information from the database.
6	User 1 information is displayed on User 2's device.
7	User 2 views information submitted by User 1.

2.1.1 Student Social Apps

Social applications have been on the uprise since the late 2000's when smartphones were released and subsequently proliferation in mobile apps. On the other hand, the majority of social applications for students have always been sub sections for major social media Platforms such as, Facebook or the application's main purpose is educational (*Al-Rahmi and Zeki, 2016*).

A survey conducted on the use of social networking sites by university students found the primary use of social media is socialising, although there are uses for educational and academic purposes, there is a significance in enhancing student engagement and learning if the social media platform integrates into the curriculum in a meaningful way (*Barnes and Tynan, 2007*).

A study was carried out to investigate the use of social media in higher education and found the positives in student engagements, learning and collaborations. On the other side, it has been noted that social media also can be distracting and lead to negative impacts if they're not used properly (*Manca and Ranieri, 2016*).

Overall, the student social apps literatures suggest social media for students can have both positive and negative impacts engagement and academic performance. Despite the primary use of social media in socialising, student social apps can be an effective tool to aid engagement among peers and promote students learning together if the study is built correctly and meaningful with ethical issues solved for the application.

2.1.2 University Student's Mental Health

There have been literatures written in the past after several investigations indicating the mental health of university students and found anxiety, stress and depression were major mental health concerns. The study (Stallman, 2019) suggests, social support was an important factor in aiding the mental health and wellbeing among university students.

A review released for the use and effectiveness of mobile apps in managing depression and anxiety (*Arean, Hallgren, Jordan, Gazzaaley, Atkins, Heagert and Anguera ,2016*), the study demonstrated and examined the effectiveness of mobile application measuring depression and anxiety. It has been found that apps can be effective for measuring and managing symptoms of depression and anxiety with the correct tools in place but also noted further research is required to determine optimal features in application to resolve this issue in a greater scheme.

A study was conducted to investigate the use of Facebook as a tool to promote social support and wellbeing for university students. The study found Facebook can be an extremely effective tool to promoting social support and reducing anxiety, stress, and depression among university students (*Grieve, Indian, Witteeveen, Anne Tolan and Marrington, 2013*).

2.1.3 Mobile User Experience

Mobile User experience (UX) refers to how a user interacts on a mobile device and the simplistic and satisfaction they derive from using elements and components within the device. UX requires studies, questionnaires to the target audience and testing its responsiveness within the design in order to build a successful mobile application.

UX is one of the most important factors for user engagement. According to Google Blogs, one of two people expect a page to load under 2 seconds, 46% of people believe waiting for pages to load is what they dislike most and 53% of users are likely to abandon the page if it takes more than 3 seconds to load (Google Blogs, n.d.).

User research is critical to gather data and understand the users' needs, behaviour and preferable experience. This can help designers and developers address issues and create an efficient and effective mobile experience for their targeted users. Section 4.4.3.3 Prototyping Questionnaire addresses the questions that were asked as well as the responses which was extremely impactful for the development of this project.

2.1.4 Android or iOS for Students

Table 1 Android and iOS Advantages and Disadvantages

Android Advantages	Description
Cost Effective	iOS devices are generally more expensive in comparison with Android devices. The affordability for students who purchases a mobile device may choose Android over iOS due to the customisability, budget, and brand choices.
Compatibility	Generally, Android devices has a huge range of software compatibility than iOS devices, this is due to the customisation and the support of a wider range of file types. iOS on the other hand, are designed to be compatible with other iOS devices and other apple products and services only.
Google Integration	Google integration with Android gives them an upper hand with students who use email, documents, and other academic tools. This makes android a significantly better choice for students who already heavily use Google services. Despite Apple iOS do have some integration with Google services, it does not compare to the extent Android provides.
Multiple Device Options	Android devices are available in a wider range of brands, styles, sizes, and costs. This gives students a wider choice of mobile devices to choose from within their budget.
Expandable Storage	Some Android devices offer expandable storage options using microSD card, this would be extremely useful for

	students who need to store large amount of data including research papers, studies, and other projects. On the other hand, iOS does not offer any expandable storage options. Over the past few years, iOS has tackled some of the issues using cloud services, however the user will always require a network connection to the Internet.
User Interface	iOS is generally considered to be more user friendly in comparison to Android devices, with simple and consistent designs. However, this could limit the user's customisability and would like to personalise their device further than what Apple iOS offers.
Security	Both Android and iOS offer strong and robust security to protect the user from security threats. Apple devices are generally considered to be more secure in comparison with Android. This could be important to students who have sensitive data stored on their device.
Accessibility	Both Android and iOS offer features for accessibility. Features including, high contrast, text-to-speech and many other features assists students with disabilities. iOS is considered to have better integration with assistive technologies, making them a better choice for students with disabilities.

(*Muchmore, 2020*)

While iOS devices are generally more expensive than Android devices, the customisability, budget, and brand choices make them a more attractive option for students. Software Compatibility, Google integration, expandable storage gives Android a more suitable choice for students. Although iOS offer strong security, accessibility, and user-friendly design, from a student's perspective Android is a more suitable choice over iOS.

2.1.5 Non-Structured Firebase Realtime Database or Structured MySQL

Firebase Realtime Database is a NoSQL cloud-based database which offers real-time data to synchronise and transfer across multiple clients. This means the non-structured database can be stored in a hierarchical structure in comparison with traditional structured tables. Furthermore, Firebase offers a wide range of libraries other than databases including authentication, cloud messaging, storage, analytics and many more (*Firebase Documentation Realtime Database, n.d.*).

MySQL is a traditional structured table database with a relational database management system using columns and rows to store data. The structured database is significantly more organised and extremely useful to create complex data structures and queries transactions. The benefit of MySQL is extremely scalable which makes it a great choice for large applications with high performance requirements and reliability (*MySQL Documentation, n.d.*).

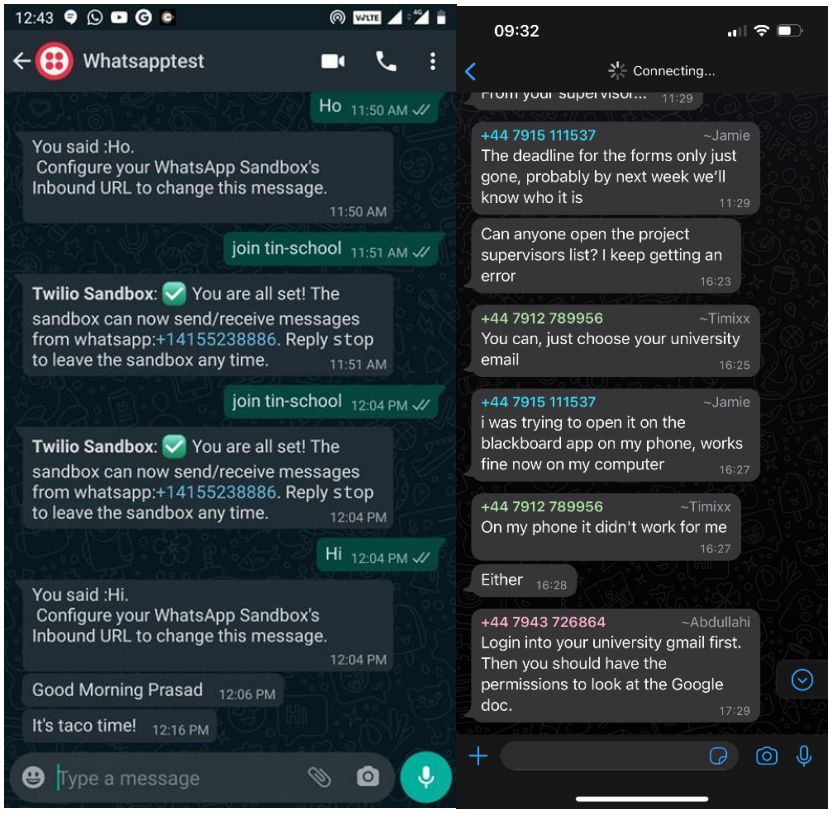
Overall, Firebase Realtime Database offers specific needs of the project, and the developer can create simple and quicker setups for real-time updates and data structures. On the other hand, MySQL is more preferable for complex projects with complex data structures and queries. Due to the integration of Google, Firebase Authentication Service and Android Studio, Google Firebase Realtime Database takes a greater edge for the development of this project.

2.2 Review of Projects / Applications

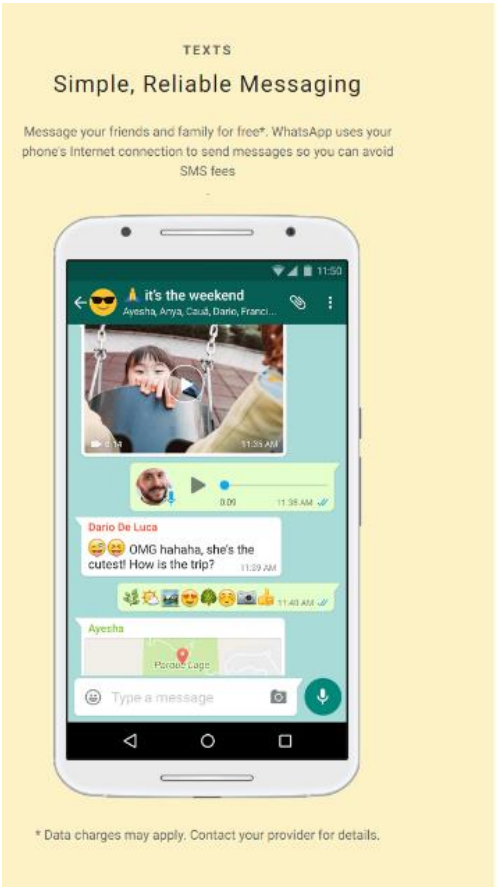
WhatsApp

The model of WhatsApp direct and group messaging distinguishes the main feature of the student's mobile application. Being able to create a messaging log as well as the ability add multiple participants with a clear understanding on which user is conversing gives clarity to the user reading. The intention to keep all other users messaging on the left side of the screen and only your conversation on the right provides a simplified usability feature for the user to differentiate their input. On the other hand, a case study has demonstrated students has said "I still prefer emailing my professors even though it seems old school. Honestly, Whatsapp doesn't give me a formal, official, or private impression. What if I get hacked or something?" *(Han, 2021)*. Taking this into consideration, this mobile application could be used for informal or formal intent depending on the user, nonetheless ethical principles and moderation will be monitored for large discussion areas. Moreover, WhatsApp uses end-to-end encryption security techniques to ensure messages are only available to the sender and recipient.

"WhatsApp's end-to-end encryption is used when you chat with another person using WhatsApp Messenger. End-to-end encryption ensures only you and the person you're communicating with can read or listen to what is sent, and nobody in between, not even WhatsApp. This is because with end-to-end encryption, your messages are secured with a lock, and only the recipient and you have the special key needed to unlock and read them. All of this happens automatically: no need to turn on any special settings to secure your messages." (*WhatsApp, n.d.*)



Direct Messaging Group Messaging



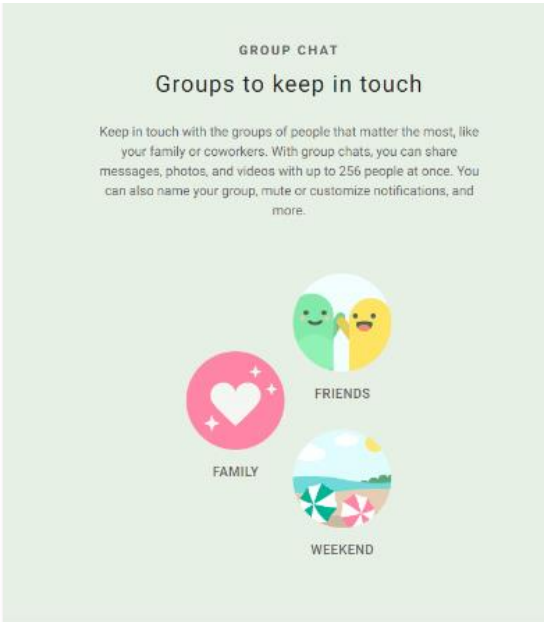


Figure 2 WhatsApp

Discord

Discord uses three main types of messaging communication among users, this includes direct, group and server text channels. This concept aligns with the student mobile application as the likelihood of finding friends on discord is through servers, therefore a similar concept of forums or discussion board could lead to finding fellow students to discuss further in private messaging and group messaging areas. Discord servers usually have one primary owner, however in some cases there could be multiple server owners with all privileges to the server Figure 4 Advanced Permissions Administrator it also indicates “This is a dangerous permission to grant.”, accordingly the ethics to ensure the owner understand a specified member of the server will gain the same access and privileges as them.

In addition, privacy and safety features also include well explained description of their roles and the capability of automatically scanning for suspicious and explicit content, consequently it provides the user safety with language filtering and etc depending on the user’s choice Figure 5 Safe Direct Messaging. Alternatively, the last resort could be the block function can ensure the user will not be able to communicate with you Figure 6 Block Function.

Figure 3 Discord App

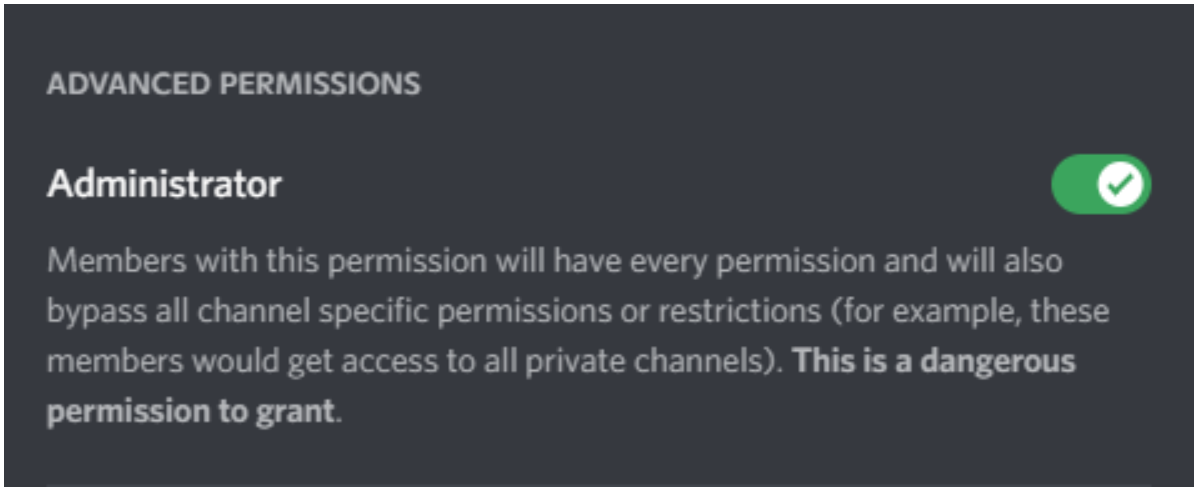


Figure 4 Advanced Permissions Administrator

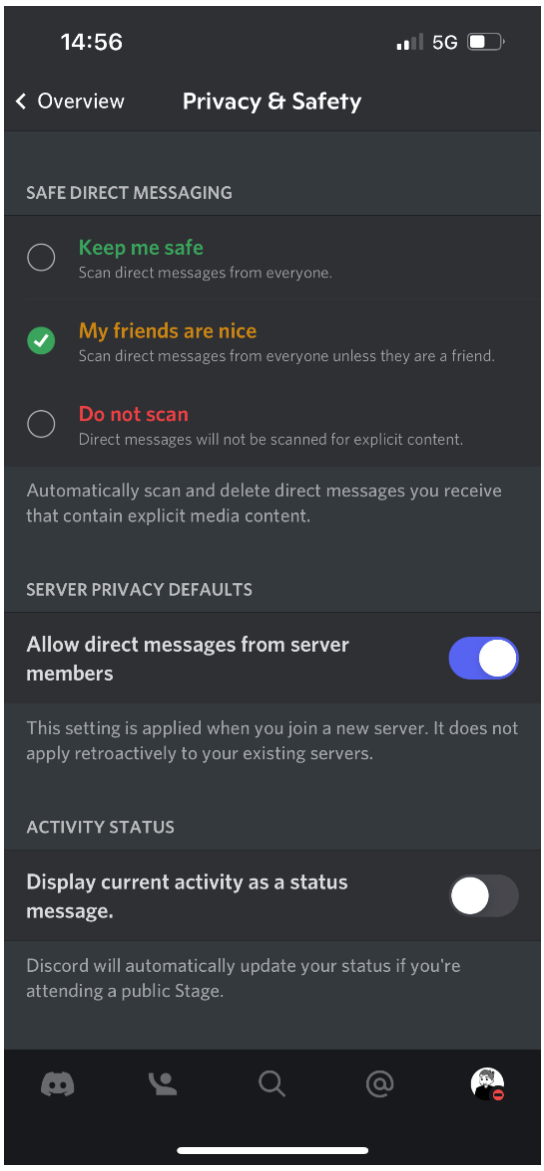


Figure 5 Safe Direct Messaging

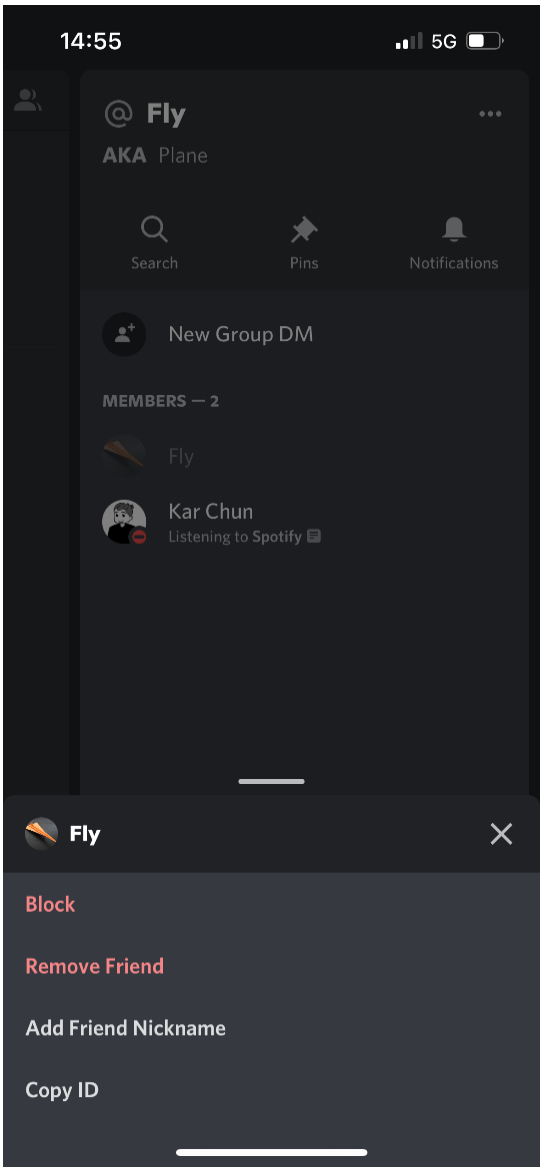


Figure 6 Block Function

CampusGroups

CampusGroups offer multiple features to imbed easy and efficient communication. Direct messaging, Group Messaging are the two options where students can communicate with their peers and share resources. CampusGroups are partnered with thousands of universities in the world, and users can search other users who is also attending the same university as them (*CampusGroups, n.d.*).

The application also prioritises user privacy and security. Users can control who can send them messages and all messages are encrypted to protect user data. The business follows the FERPA regulation (*U.S. Department of Education, n.d.*). The app can be evaluated with Figure 7 CampusGroups App.

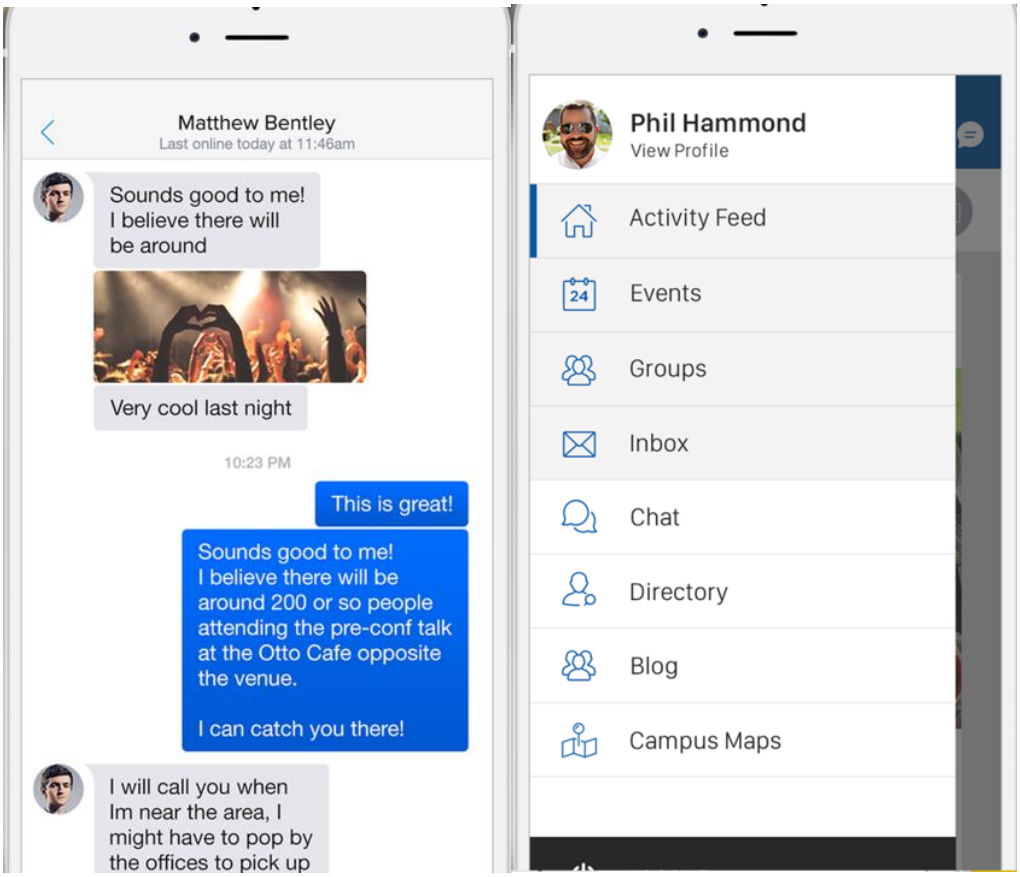


Figure 7 CampusGroups App

Table 2 Advantages of similar applications

Features	WhatsApp (Figure 2)	Discord (Figure 3)	CampusGroups (Figure 7)
Navigation Menu	✓	✓	✓
Informational for each area of the application			✓
Friends List		✓	✓
Add, Remove and Block users easily		✓	
Multipurpose	✓	✓	
Good Security and Privacy Features	✓	✓	
Accessibility	✓	✓	✓
Some form of notifications	✓	✓	✓
Great for Student Communication	✓	✓	✓
Group Messaging	✓	✓	✓
End-to-end encryption	✓		✓

Table 3 Disadvantages of similar applications

Features	WhatsApp (Figure 2)	Discord (Figure 3)	CampusGroups (Figure 7)
Overloading of features			✓
No Home Page	✓	✓	
Difficult Navigation			✓
Hard to distinguish its purpose			✓
Limited Messaging Features			✓
Unappealing Design (UX)			✓
Limited customisation	✓		✓

The comparison Table 5 Similar Application Comparison with UoWsa project are illustrated with none, limited, or fully integrated with the mobile application. This can then be evaluated against the features listed for comparable existing similar applications and the development project.

(N) = None

(L) = Limited

(F) = Fully Integrated

Table 4 Similar Application Comparison with UoWsa project

Features	WhatsApp (Figure 2)	Discord (Figure 3)	Campus Groups (Figure 7)	UoWsa Mobile App
Android	**F**	**F**	**F**	**F**
Notifications	**F**	**F**	**F**	**L**
Friends List	**N**	**F**	**F**	**F**
Direct Messaging	**F**	**F**	**F**	**F**
Group Messaging	**F**	**F**	**F**	**L**
Forums	**N**	**N**	**N**	**F**
Add, Remove and Block users easily	**N**	**F**	**L**	**F**
Understand the purpose of the application	**F**	**F**	**L**	**F**
Messaging Features	**F**	**F**	**L**	**L**
Notifications	**F**	**F**	**L**	**L**
Informational throughout the application	**L**	**L**	**L**	**F**
Can delete user account easily	**F**	**F**	**L**	**F**
Good UX Design	**F**	**F**	**L**	**F**
Great for student communication	**F**	**F**	**F**	**F**
Accessibility	**F**	**F**	**L**	**L**
Admin Support	**N**	**L**	**N**	**F**
Connectivity Status	**L**	**F**	**L**	**F**

2.3 Review of tools frameworks and techniques

This section of chapter 2 covers the literature survey of the relevant tools along with its advantages and disadvantages for programming languages, environment, and libraries for the project in development.

2.3.1 Android Studio

The mobile application in development is built on the Android Studio platform. Android Studio is an integrated development environment (IDE) in conjunction with Android Operating System and a replacement to Eclipse Android Development Tools (ADT) plugin (Android Developers, 2013).

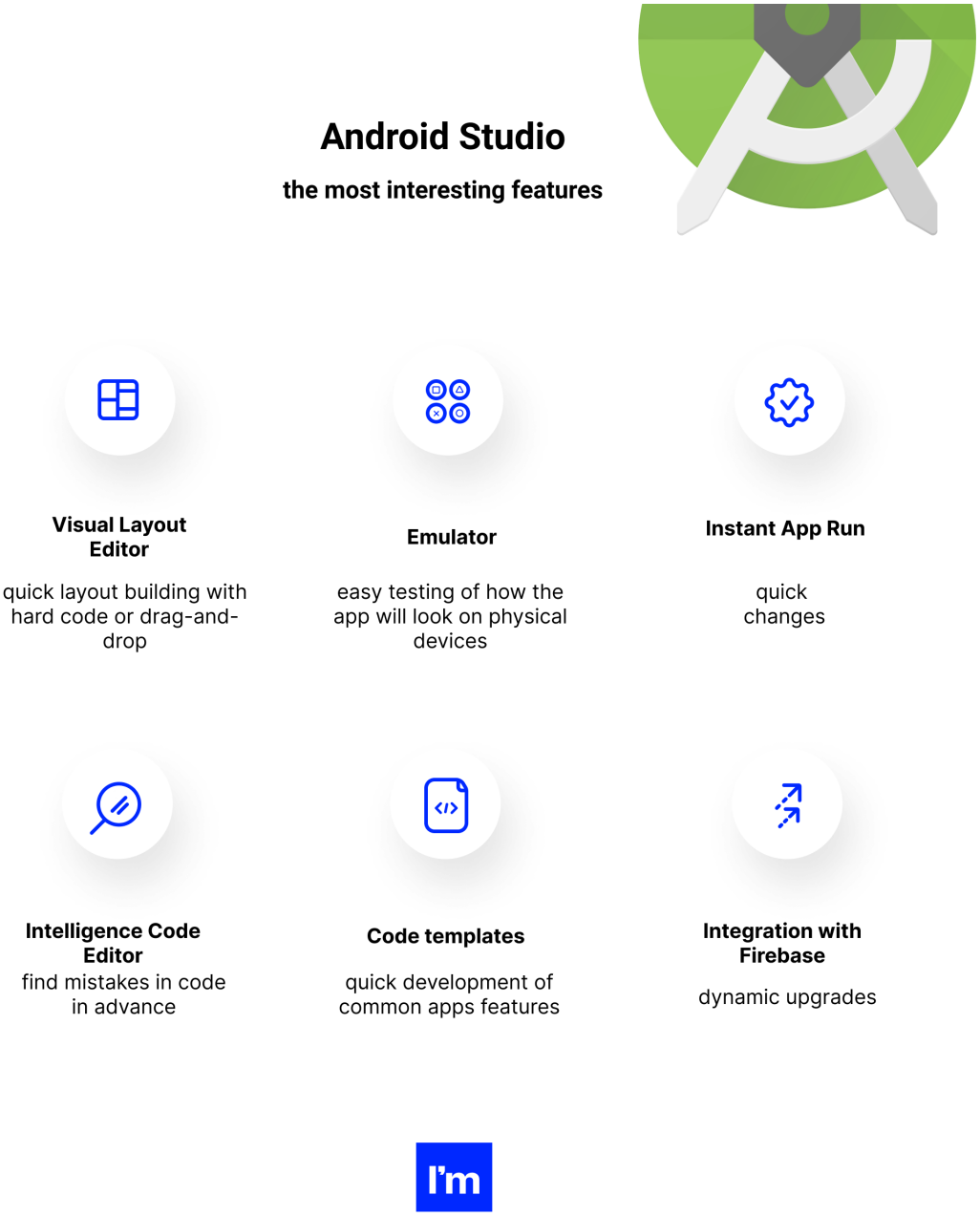


Figure 8 Android Studio Features

Table 5 Android Studio Advantages

Advantages	Description
Integrated Development Environment (IDE)	The IDE in Android Studio provides a powerful tool to implement their ideas onto the platform along with integration with Android Software Development Kit (SDK) platform, which provides a wide range of libraries and tools for Android development.
Built-in Emulator	Although some developers enjoy testing their development project on a physical device, a built-in emulator can provide a quicker process in testing their project efficiently. The emulator includes a wide range of devices; therefore, the developer can also test using multiple devices.
Visual Layout Editor	The visual layout editor gives the developer an opportunity to preview the interface of the project. In addition, it also provides the developer to create a pixel perfect structure of the design without having to launch the emulator or physical device every time they would like to view or test.
Performance Profiling	Android Studio integrates with performance profiling tool, this means the software can track and monitor usage such as, memory leaks, CPU overload and network usage. This can assist the developer in creating a well optimised application.
Firebase Integration	Firebase provides powerful tools including, authentication, database, cloud messaging, crash reports, machine learning and the integration makes all of these tools easy to set up. In theory if it was set-up manually on the Firebase console, it requires extra downloads placed into the app folder and manual dependencies are required in the Gradle file.
Kotlin Support	Although Java was the primary source in implementing features in a mobile application previously, Kotlin has become the better fit for mobile app development. Kotlin provides simple and concise syntax, built in null safety, data classes specifically designed to store data and it is also fully interoperable with Java.
Open Source	Android Studio was created by Google, and they believe in the power of collaboration to advance technology. Open source provides a great platform for developers to use their platform and contribute towards the platform, improvements, and report bugs.

Table 6 Android Studio Disadvantages

Disadvantages	Description
Limited Support	Although Android Studio provides very frequent updates, there are limited support on the platform, in comparison to another platform such as xCode or VSCode. External libraries and tools are hard to integrate with Android Studio, which makes it extremely difficult to develop, test and deploy the application.
High-End System Requirement	It can be a struggle to run Android Studio smoothly in general and it is extremely difficult to develop productively running on a lower-end system. It has been recommended by other developers to use powerful processors, additional RAM, and dedicated graphics card. In Figure 9 Android Studio System Requirements illustrates all of the system requirements to use the software application with stability.
Large Disk Space Requirement	Developers usually will have to purchase an extra drive to complete multiple projects on their systems due to the size of the project package pilling up in file size.
Resource Intensive	Running Android Studio uses a significant amount of CPU and Memory usage. Lower-end system may require the developer to close all other software applications in order to run the platform efficiently for productivity.
Debugging Issues	Android Studio can be complex to many developers who is unfamiliar with the platform, which would be extremely challenging to identify and debug an issue. Furthermore, Android Studio can sometimes indicate inconsistent errors, making it difficult to pinpoint the cause of the error.
Gradle file could take a long time to build. Especially for projects under development.	Gradle is the build tool used to incorporate and assemble all the components used in the project. This could be a long waiting time even for higher end system and especially for larger project files with multiple complex project structure.

(*Karczewski, 2021*)

SYSTEM REQUIREMENTS

MacOS®

- MacOS® 10.14 (Mojave) or higher
- ARM-based chips, or 2nd generation Intel Core or newer with support for Hypervisor.Framework
- 8 GB RAM or more
- 8 GB of available disk space minimum (IDE + Android SDK + Android Emulator)
- 1280 x 800 minimum screen resolution

SYSTEM REQUIREMENTS

Linux

- Any 64-bit Linux distribution that supports Gnome, KDE, or Unity DE; GNU C Library (glibc) 2.31 or later.
- x86_64 CPU architecture; 2nd generation Intel Core or newer, or AMD processor with support for AMD Virtualization (AMD-V) and SSSE3
- 8 GB RAM or more
- 8 GB of available disk space minimum (IDE + Android SDK + Android Emulator)
- 1280 x 800 minimum screen resolution

SYSTEM REQUIREMENTS

Microsoft® Windows®

- 64-bit Microsoft® Windows® 8/10
- x86_64 CPU architecture; 2nd generation Intel Core or newer, or AMD CPU with support for a Windows Hypervisor.Framework
- 8 GB RAM or more
- 8 GB of available disk space minimum (IDE + Android SDK + Android Emulator)
- 1280 x 800 minimum screen resolution

SYSTEM REQUIREMENTS

Chrome OS

- For information on recommended devices and specifications, as well as Android Emulator support, visit chromeos.dev.

Figure 9 Android Studio System Requirements

2.3.2 Kotlin

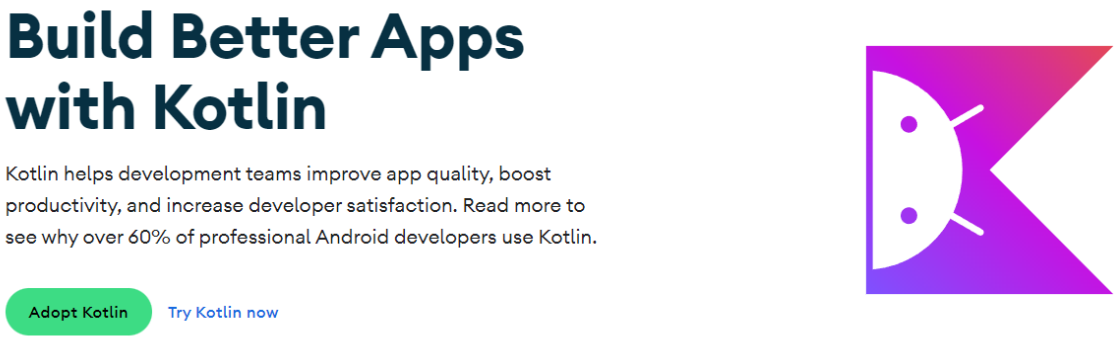


Figure 10 Android Developers Kotlin Introduction (Developers Android Kotlin, n.d.)

Figure 10 Android Developers Kotlin Introduction indicates the release in statistics from Official Jetbrains Android Developers website. It illustrates over 60% of professional Android developers use Kotlin to create mobile applications.

Statistics of Kotlin users compared to Java users on Android Studio

2017:

- Kotlin was officially supported by Google for Android Development in May 2017.
- It was estimated 5-6% of Android Developers who uses Kotlin primarily for project development.

(*Wharton, Muntenescu and Lau, 2018*)

2018:

- It was estimated 7% of Android Developers who uses Kotlin primarily for project development.
- Approximately, 20% of Android Developers who have used Kotlin, however Java was still their preferable choice.

(*JetBrains Kotlin Census 2018, n.d.*)

2019:

- According to a survey in 2019 on Stack Overflow, 36.6% of professional Android Developers use Kotlin, compared to 64.9% using Java.

(*Stack Overflow Developer Survey Results 2019, n.d.*)

2020:

- According to the “State of Kotlin” survey conducted by JetBrains the company who provides support and updates for Android Studio, 62% of developers use Kotlin for Android Development, however within the same survey 60% of Android developers were using Java.

(*JetBrains Kotlin Census 2020, n.d.*)

2021:

- Statistics were release from the Google Play Console indicating 96% of Android apps on the Google Play Store was created using Java, however in August 2021, over 83% of the top 1,000 apps were created using Kotlin.

(*Raj, 2021*)

2022:

- As of April 2022, Kotlin was ranked 31st in the Tiobe Index from 47th a year before. Tiobe Index is a measurement on the popularity of programming languages based on data gathered from search engines.

(*Statistic Times, 2022*)

Kotlin, is a programming language design to run on the Java Virtual Machine (JVM), was supported by Google in 2017. Initially a small percentage of Android developers used Kotlin primarily for project development, however over the years, Kotlin’s popularity has been on the rise. In 2020, it was reported over 62% of Android developers used Kotlin for Android development.

Table 7 Kotlin Advantages

Advantages	Description
Null Safety	Kotlin is built with null safety into the language. This means the developer does not have to think about null pointer exceptions.
Functional Programming	Kotlin supports features of functional programming such as, High-Order and Lambdas, which makes the developer's life easier without having to add additional code to solve the issue.
Community	Kotlin has been on the uprise for the past few years and there is a large and active community for assistance in support and resources for learning.
Coroutines	Coroutines is a simple alternative to threads, this makes Coroutines a lightweight and efficient way to write concurrent code.
Type Interface	Kotlin supports Type Interface, where the complier can infer the type of variable based on the context or value in which was used. This makes coding much easier to read and write.
Extension Functions	Extension functions allow Kotlin developers to add functions to existing classes without modifying existing code. This feature allows the developer to add new functionality or test them without breaking the original code.
Backwards Compatibility / Interoperability	Kotlin is backwards compatible, this means Kotlin code can run in Java projects and vice versa.

(*Sagara Technology Idea Lab, 2020*)

Table 8 Kotlin Disadvantages

Disadvantages	Description
Limited Adoption	Kotlin's popularity is rising, however it's not nearly as adopted as Java at the time of research. Opportunities such as career may be fewer in comparison with Java.
Limited Documentation	The documentation for Kotlin is not nearly as developed as Java, especially for advanced features.
Performance	Although Kotlin was developed to be efficient and fast, there can be performance issues based on the complexity of the code.
Community	The community is rising, however assistance and help from the community is limited in comparison with Java.
Interoperability Issues	Even though Kotlin was designed to be backwards compatible and vice versa, it could lead to issues in translating the code between one and another. This issue happens frequently with older Java codebases in transition to the Kotlin Programming language

(*Ajas, 2022*)

2.3.3 Firebase Authentication

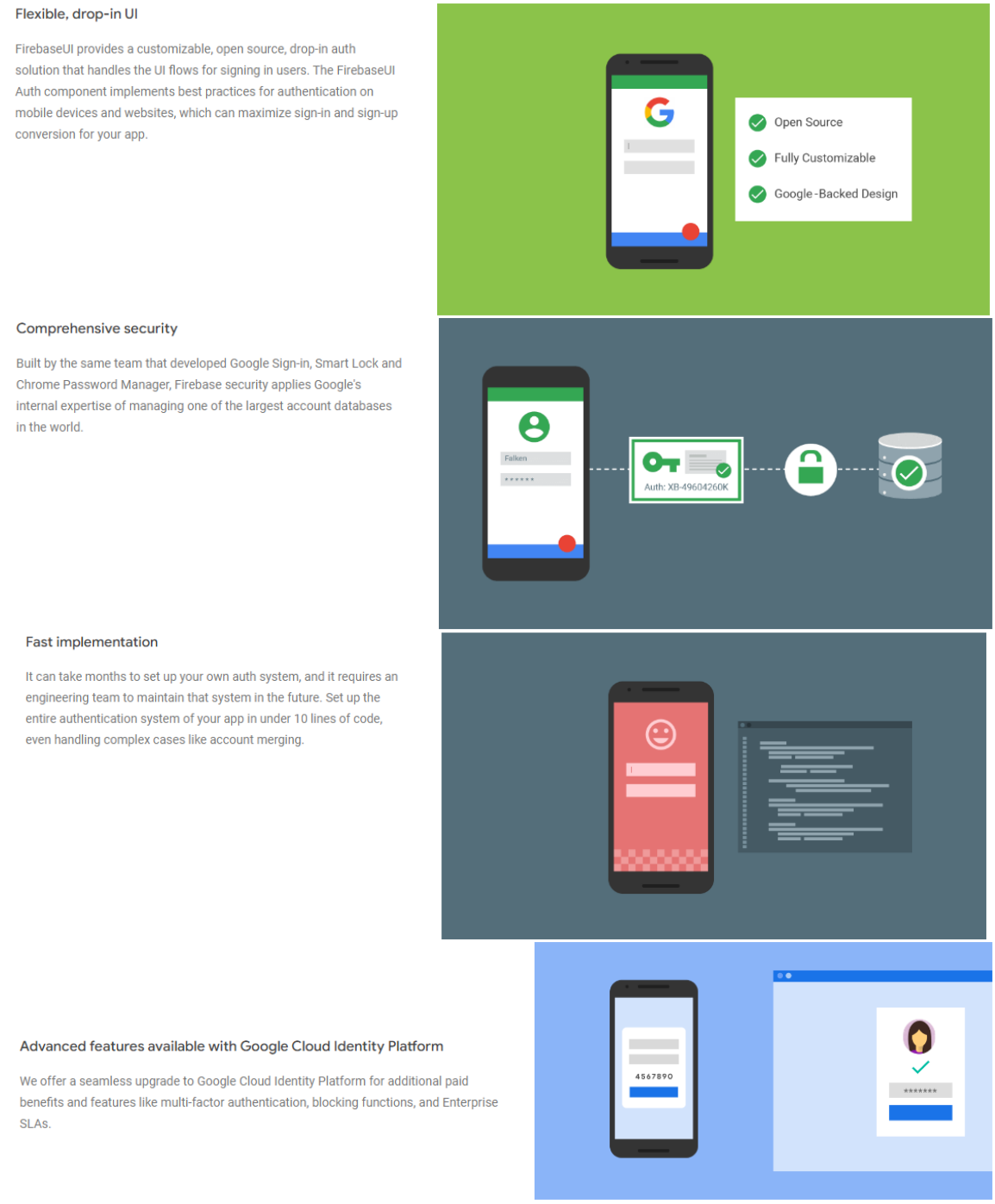


Figure 11 Firebase Authentication Features (Firebase Authentication, n.d.)

Figure 11 Firebase Authentication Features illustrates the main features of Firebase Authentication Services to handle sign-up and sign-in for the client as well as comprehensive security, fast implementation and features working in conjunction with Google Cloud Identity Platform for extra safety in multi-factor authentication.

Table 9 Firebase Authentication Advantages

Advantages	Description
Secure Authentication	Firebase Authentication was developed with a secure infrastructure to assure the integrity of data. Features including, Email verification, password resets and account management tool gives the developer assurances on the security of user data.
Easy Integration	Firebase Authentication provides easy integration with other Firebase services and Software Development Kits (SDKs), making it easier for developers to build the backend infrastructure.
Third-Party Integration	Firebase Authentication supports third-party authentication providers such as, Google, Facebook, Twitter, LinkedIn and many more.

(*Firebase Authentication, n.d.*)

Table 10 Firebase Authentication Disadvantages

Disadvantages	Description
Limited Customisation	Firebase Authentication supplies with multiple features, however there are limited options for developers who would like to develop their projects with complex authentication.
Technical issues	Firebase Authentication is a cloud-service, therefore if a problem occurs with their services, it will impact the availability or performance of the application.
Cost	Firebase Authentication is a paid service once the user base grows, this could be costly for businesses in comparison to other authentication alternatives.
Dependency	Firebase Authentication is a dependency in which the developer must rely on. Any changes from Firebase Authentication may impact the stability of the application.

(*Clark, n.d.*)

2.3.4 Firebase Realtime Database

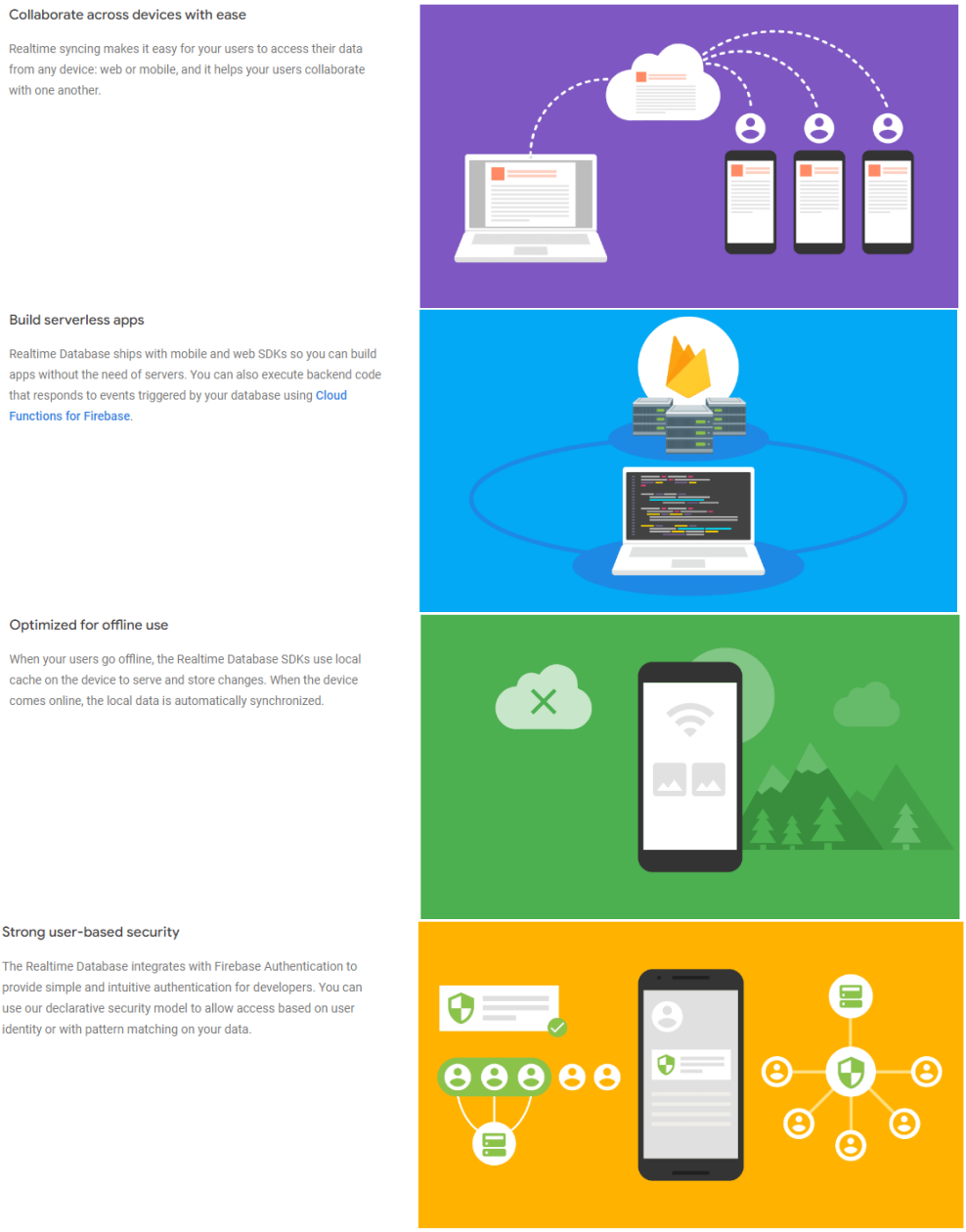


Figure 12 Firebase Realtime Database Features (Firebase Sync, n.d.)

Figure 12 Firebase Realtime Database Features illustrates the NoSQL cloud-hosted database allows the developer and client user to store and sync data across multiple devices to provide a collaborative and immersive experience for users without the use of complex networking code. It is also designed to work offline, meaning the apps remain operative without a network connection. The SDK automatically preserves data onto the user's disk and once connectivity is restabilised, any changes with be synchronised with the server side. It can also be accessed across multiple client devices without the need for an application server. Security and data validation are available on the database security rules, which are expression-based rules that are executed when data is read or written.

Table 11 Firebase Realtime Database Advantages

Disadvantages	Description
Cross-Platform Support	Firebase Realtime Database includes cross-platform support, this allows developers to create multiple platforms such as, Android, iOS, Web to use the same database at the same time.
Realtime Sync	Firebase Realtime Database grants real-time synchronisation between devices. This makes the development process quicker as well as applications which requires real-time updates.
Offline Support	Firebase Realtime Database allows applications to continue to function without an Internet connection.
Scalability	Firebase Realtime Database was designed to be scalable; this allows developers and users to insert large amounts of data without the database bottlenecking.
Easy Integration	Firebase Realtime Database provides easy integration with other Firebase services and Software Development Kits (SDKs), making it easier for developers to build the backend infrastructure.

Table 12 Firebase Realtime Database Disadvantages

Disadvantages	Description
NoSQL Structure	Firebase Realtime Database may not be suitable for developers which requires complex querying and data modelling.
Limited Queries	Firebase Realtime Database provides limited querying and indexing capabilities, developers could find this difficult to target data in large datasets with complex queries.
Limited Transactions	Firebase Realtime Database has limited transaction support, this could be extremely difficult to execute atomic operations in large datasets.

2.3.5 Firebase Storage

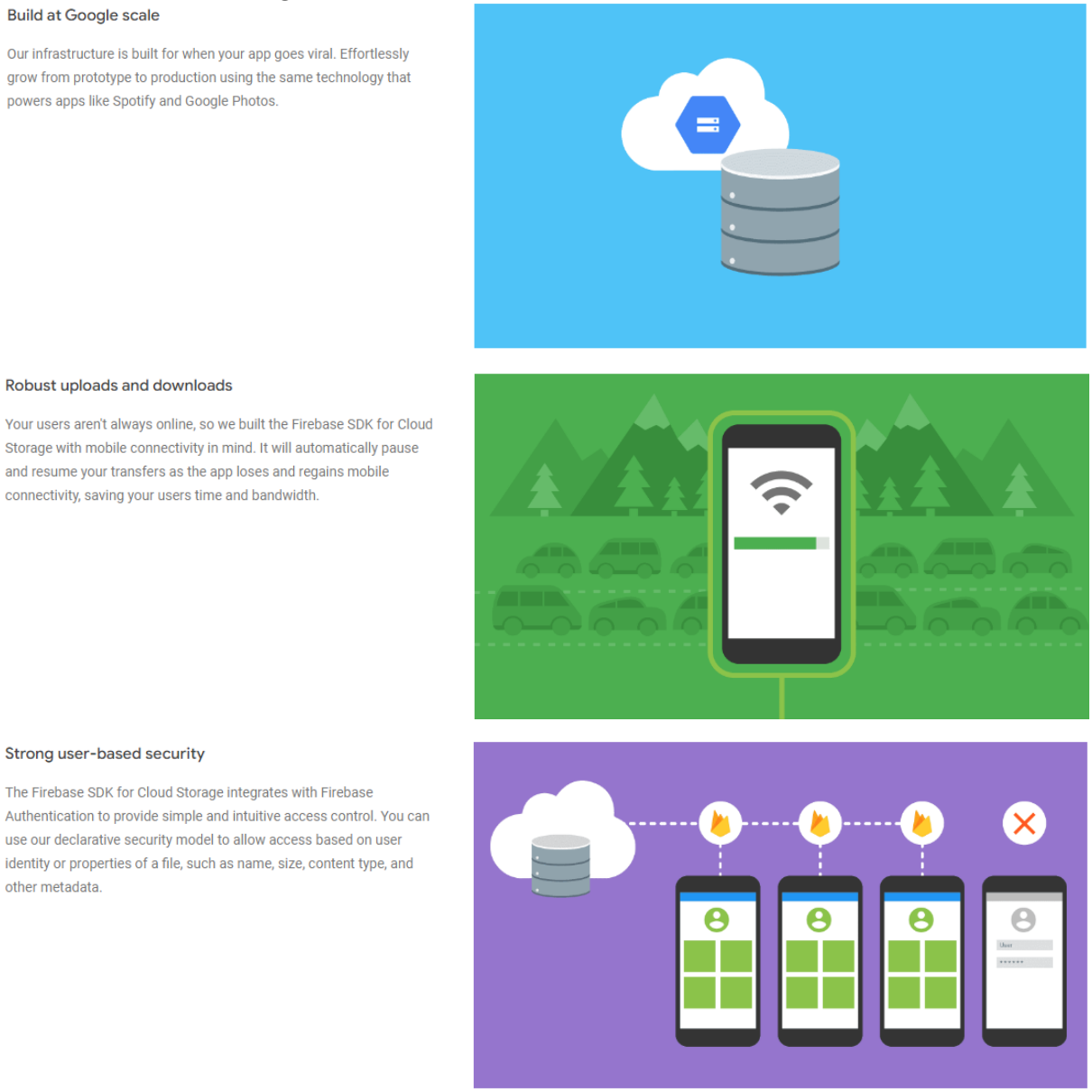


Figure 13 Firebase Storage Features (Firebase Storage, n.d.)

Firebase Storage is a cloud-based storage solution that enables developers to store and server user-generated content such as, images, videos, and other static related files to their apps. Moreover, the platform provides a robust uploads and downloads of files even when users are offline, this means it can automatically pause and resume file transfers as the app lose and regains mobile connectivity.

The strong user-based security integrates with Firebase Authentication; therefore, developers can use declarative security model to enable access control based on the user identity of a file using name, size, and content type.

3. Legal, Social and Ethical Issues

This chapter will cover the legal, ethical and security issues that associated with the development project along with data collection and analysis of the corresponding research.

3.1 Gathering Data

There is considerable amount of legal and ethical issues prior to collecting or using data from others.

Legal

The legal aspects of gathering data for the corresponding research involve data protection legislation, privacy policies, copyright and intellectual property, and data breach notification.

As the Data Protection Act 2018 was introduced with the General Data Protection Regulation (GDPR), it has stated in article 6(1) “Lawfulness of processing” that data must be collected for specific, explicit, and legitimate purposes. Organisations should only be collecting, processing, and storing personal data on a lawful basis (*intersoft consulting [1], 2018*).

Privacy Policies are compulsory in many jurisdictions in order to provide accountability and transparency in the collection of personal data. The GDPR Act article 13 “Information to be provided where personal data are collected from the data subject” requires organisations to provide individuals/users a privacy notice which includes information data processing activities and the rights of the individuals/users (*intersoft consulting [2], 2018*).

Data breach notification is an important aspect of data protection. Informing individuals and relevant authorities when the integrity of data has been compromised as a result of a data breach. It is important to have a plan in place for notifying individuals and authorities if a data breach occurs. The GDPR requires organisations to notify authorities of the event within 72 hours of the knowledge gained of a data breach in Article 33 unless it’s unlikely to result in a risk to individuals (*intersoft consulting [5], 2018*).

Obtaining legal consent is an important legal requirement when collecting and processing personal data. Article 4(11) “any freely give, specific, informed and unambiguous indication of the data subject’s wishes by which he or she, by a statement or by a clear affirmative action, signifies agreement to the process of personal data relating to him or her” (*intersoft consulting [4], 2018*). Furthermore, the UK Information Commissioner’s Office (ICO) provides guidance on obtaining valid consent under the GDPR (ico [1], n.d.) illustrated in Figure 14 ICO Consent Guidance.

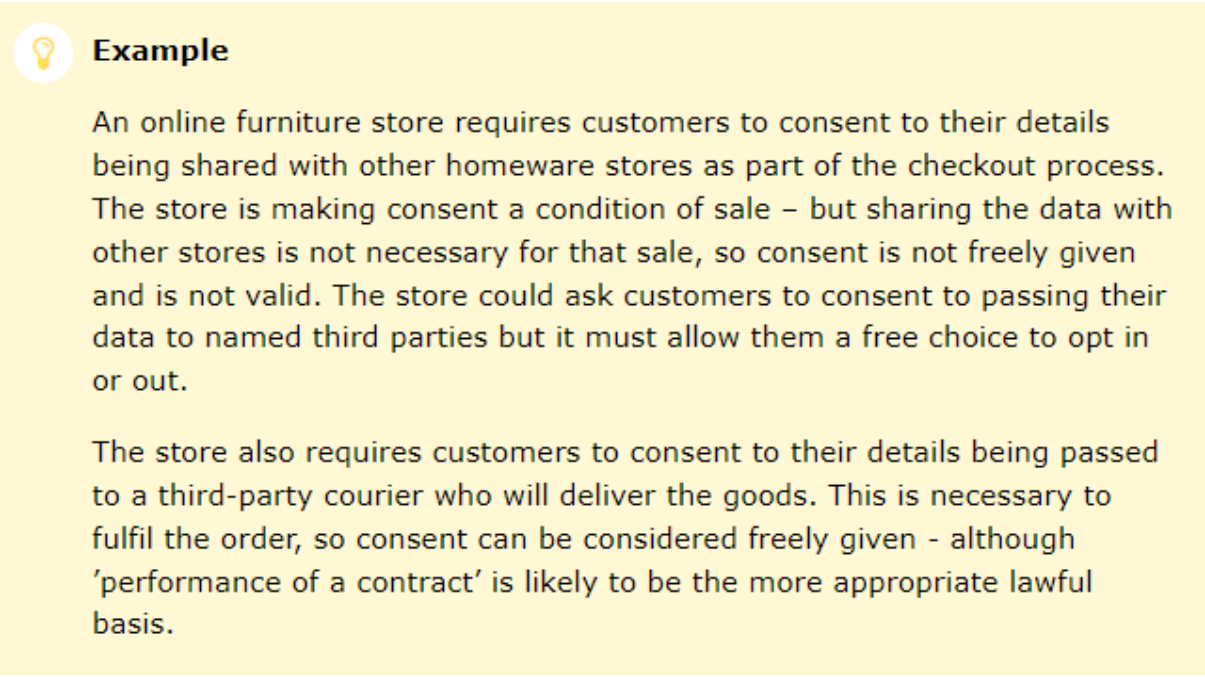


Figure 14 ICO Consent Guidance

Note: An example of consent form were used between the supervisor and the developer in this project in Appendix II.

To avoid these legal breaches, written consent should be provided to all participants including all of the information and process conducted, which should be signed off by both the conductor and participant to ensure both parties are aware of its contents and agreement.

Note: An example of participation form were used between the supervisor and the developer in this project in Appendix III.

Ethical

The ethical practices for data gathering can assist to ensure personal data is collected and used with the respect of individuals and demonstrate transparency and fairness.
Laerd Dissertation outlines ethics as “Ethical principles mean that as a researcher, you need to: (a) obtain informed consent from potential research participants; (b) minimise the risk of harm to participants; (c) protect their anonymity and confidentiality; (d) avoid using deceptive practices; and (e) give participants the right to withdraw from your research” (*Laerd Dissertation, n.d.*).

Principle One Minimising the risk of harm:
- No physical harm to participants
- No psychological distress and discomfort
- No social disadvantage
- No invasion of participants' privacy and anonymity

Principle Two Obtaining informed consent and participant understanding that:
- They are taking part in research
- What the research requires from them

Principle Three Protecting anonymity and confidentiality:
- Protecting the anonymity and confidentiality of research participants is important in research ethics
- Identifiable information can be removed or provided with proxies when writing up.
- Seek permission for access to data and analysis to be restricted to published material

Principle Four Avoiding deceptive practices
- Deceptive practices in research can pose challenges to informed consent.
- Covert research may be necessary when it’s not feasible to let everyone in a research setting know what is being studied, or when overt observation may alter the research being studied.

Principle Five Providing the right to withdraw
- Participants should always have the right to withdraw from the research
- Participants should have the right to withdraw at any stage in the research process
- Participants should not be pressured or coerced to continue with the research if they choose to withdraw

Note: An example of ethics form was used between the supervisor and the developer in this project in Appendix IV.

3.2 Messaging

Legal

The legal issues in messaging are privacy and data protection, intellectual property, and security.

Privacy and data protection refers to the GDPR and how organisations must follow data protection regulations. It is also important to add Terms of Service within the application to ensure the collection and processing of data is not misused along with the availability of access to delete messages. GDPR Article 17 “Right to erasure (‘right to be forgotten’)”, where individuals have the right to request the erasure of their personal data without undue delay in certain circumstances (*intersoft consulting [5], 2018*).

Intellectual property is the organisations’ requirement to ensure users are aware of the potential risks of sharing copyrighted material or content over messaging platforms. Policies and guidelines should be implemented to ensure users are aware (*Oturu, 2022*) of unauthorised content.

Organisations that use messaging platforms must take appropriate measures to ensure the security of the data they collect and process. This may include implementations on encryption, access controls, and security updates. Overall, the organisations are responsible for the safety and integrity of the data in transit over their network.

Furthermore, new rules are in place to expand upon the security and privacy measures to boost consumer security and privacy.

- Share security and privacy information in a user-friendly manner
- Robust and transparent app betting process to ensure only secure apps are published
- Allow apps to work even if optional functionality and permissions are disabled
- Vulnerability disclosure process in place for reporting and resolving software flaws
- Ensuring app developers continue to update their applications to reduce the number of security vulnerabilities

(*Gov.uk, 2022*)

Ethical

The ethical issues involve privacy and confidentiality, bias and discrimination and transparency.

Users have the right to believe and expect their messages will be kept private and confidential through direct messaging and not shared anywhere else. The messaging platform has the responsibility to ensure the user's privacy is respected. It is ethical to ensure transparency on what data is collected and clear indication of protection policies and practices.

Moreover, organisations should be aware of the potential for bias and discrimination, therefore developers should design with inclusivity in mind, including underrepresented personnel. Datasets and algorithms should be more diverse to ensure it reflects on the real-world population and target audience. Regular monitoring of user-generated content can help identity and remove any instances of bias or discrimination. Inserting administrators inside the application can also help towards the following of the terms of service and privacy policy in place.

(*Morris, Scott, and Mars, 2018*)

Social

The social aspects of messaging platforms may include cyberbullying, this subsection includes what it is and how it can be prevented.

According to Unicef cyberbullying is "bullying with the use of digital technologies, it is repeated behaviour aimed to scare anger and shame those who are targeted".

In order to tackle this problem, there are several ways to stop this from occurring:

- Strong community guidelines
- Moderation
- Blocking and reporting features

(Unicef, n.d.).

3.3 Firebase Authentication, Database and Storage

Legal

Firebase Authentication, Database, and Storage collects data from users during the authenticating process provided by Google. Google implemented measures to ensure Firebase services are GDPR compliant (Google Cloud pg12, n.d.).

Google provides data processing agreement (DPA) for users to comply with GDPR. The DPA outlines the obligations and responsibilities of both its customer / developer and Google for Firebase services. However, Firebase services does not make developer projects GDPR compliant, as the developer and data controller should be responsible for ensuring the application comply with all security and GDPR requirements. In addition, there are features and tools the developer can implement such as data retention policies and the ability to request user data deletion (*Google Cloud, n.d.*).

Ethical

The ethical concerns of Firebase Services always depend on how the developer implements the code due to the responsibility to handle personal data carefully, be transparent with users and obtain informed consent and take appropriate action to protect user data. Despite, the defined list of GDPR definitions set by Firebase. Firebase services also introduced vendor lock-in, where it becomes difficult and costly to switch to a different platform or service. The data controller should consider long term implications and ensure a plan is in place to migrate to another platform (*Cloudflare, n.d.*).

Note: Firebase GDPR definitions are available in Appendix VI.

Security

Due to Firebase being an inclusive platform, it could lead to the following issues:

- Weak Server-side controls
- Poor authorisation and authentication

Weak server-side controls lead to the limited capabilities of the developer to build a mobile application to its full potential. This can lead to data hijacking by hackers. It is extremely important to keep data secure and the best solution is to encrypt the data. If poor authorisation and authentication occur, it allows unauthorised criminals to access the application through other user's accounts. The best solution to this is to only allow login whilst there is an internet connection. By default, Firebase is integrated with offline mode. Alternatively, multi-factor authentication can be integrated for enhanced security (*Aditya's Blog, 2022*).

3.4 Testing

Legal

There are multiple factors to consider for legal issues associated with testing the application.

Testing:

- Lengthy testing phase to ensure the application does not show signs
- As the developer, you have the ethical and legal responsibility to ensure sensitive user data is protected from unauthorised access
- Ensure the application complies with GDPR regulations

(*Hamilton, 2023*)

“The General Data Protection Regulation (GDPR) is a set of regulations that aim to protect the privacy and personal data of European Union citizens. As a developer it is important to ensure your application complies with GDPR regulations, such as obtaining user consent for data collection and providing user with the ability to access and delete their data” (*European Commission, n.d.*).

Ethical

Testing (ACM Code of Ethics):

- Reduce risk and avoid harm to testers
- Be Fair and act not to discriminate
- Respect Privacy
- Honour Confidentiality

(*ACM, n.d*.)

In order to prevent ethical issues during testing phase it is important to:

- Include a clear plan and outline potential risks and minimise harm
- Implement policies to promote fairness and non-discrimination
- Only collect data that are necessary
- How sensitive information including its procedures and how to report breaches.

(ico [2], n.d.)

3.5 Deployment of the Application

Legal

The application must comply with laws and regulations related to data protection, privacy, and intellectual property. Furthermore, the developer must ensure the app does not violate any copyright or trademark laws, and that they do not engage in any fraudulent or deceptive procedures. Moreover, the End User Licence agreement / terms of service must be included within the application before deployment.

Note: Terms of Service were used in the development of the project, and it can be evaluated in Appendix VII

Documentation

- A comprehensive record of data collected, processed, and utilised by the application
- Technical data collected about the user's device and operating system
- Function aspects of the application, include both free and paid features
- Company registration location or copyright holder location for the application
- Nationality or residency of the majority of users
- Age requirements

Google Developer Policies

- Restricted content definition
- App Store listing and promotion
- Impersonation and intellectual property
- Rules for monetisation
- Privacy, security, and deception regulation
- Spam and minimum functionality

Technical Requirements

- Unique Bundle ID
- Signed App with a signing certificate
- Size of the application
- File format

(*Kharychkova, 2022*)

Ethical

Mobile app addiction is an ethical issue that has gained attention in recent years, in particular social media and gaming apps. The addition has a negative to users' mental health, therefore Google has guidelines in place for Android Apps for promoting healthy app use. The guidelines include the app's features and functionality to avoid the use of deceptive tactics to encourage usage. Also, a tool to manage their screen time was introduced to tackle the issue at hand (*Google Play Developer Policy Center, n.d.*).

Many experts have called for a proactive approach to address mobile app addiction and developments in UX designs that will improve the developer's knowledge in users' wellbeing in planning. This could include features such as, the time limit spent on an app, reminders to take a break and feedbacks from users to address such issue (*Kwon, 2016*).

Social

The developer should consider any social responsibility and how their app impacts society. Ensuring the app's contents do not promote offensive, inappropriate, harmful, or discriminatory behaviour and do not contribute to harmful stereotypes and misinformation. Developers can take social responsibility in conducting app research with the target audience, where the developer can use this data to evaluate issues and concerns. By taking a socially responsible approach to app development, developers can create apps that are not only successful and profitable but contribute with ethics and establish a good image towards society (*Redis Growth Team, 2022*).

4. Methodology

The life cycle stages of the project, methodology and development techniques used to design and implement in the project.

4.1 Agile Methodology

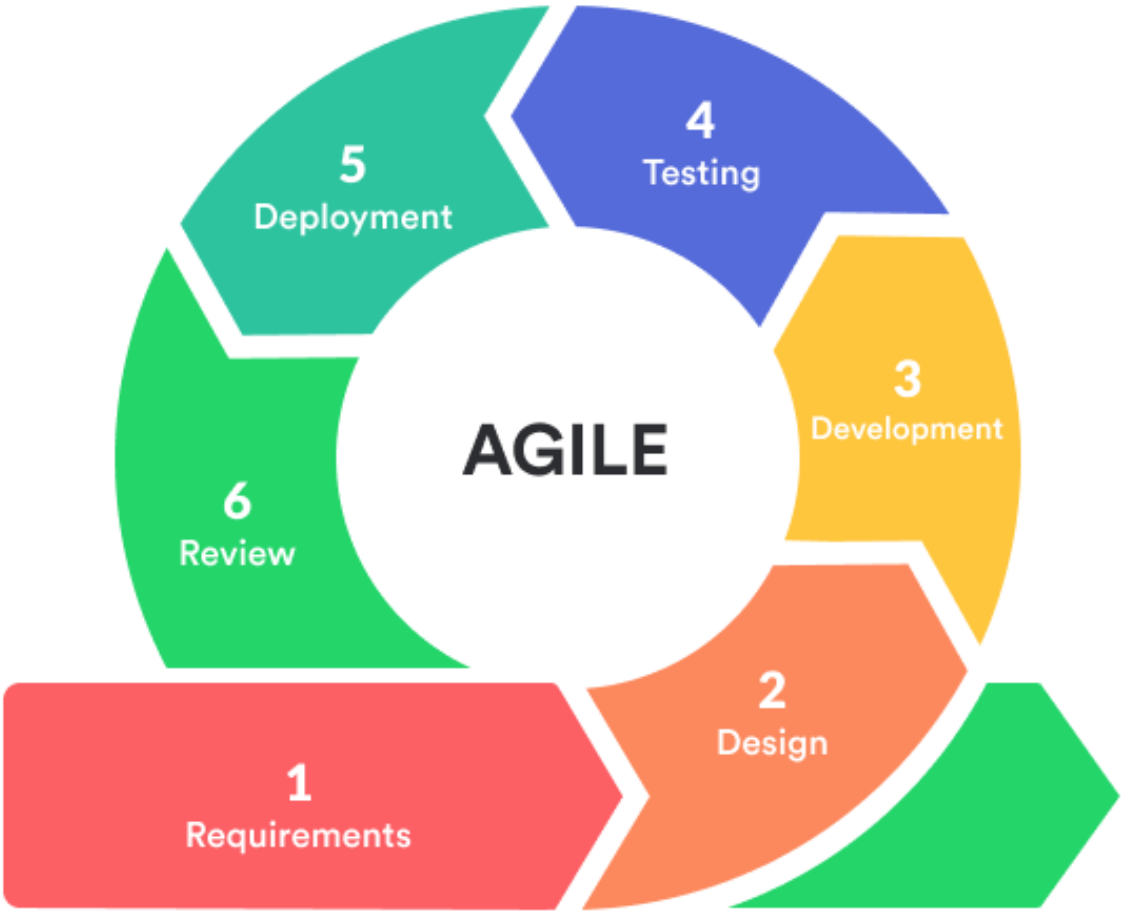


Figure 15 Agile Software Development Cycle Diagram, (Anurina, 2021)

In Figure 15 Agile Software Development Cycle Diagram, it provides a swift process in developing mobile applications due to its flexibility and iterative approach to allow feedbacks and adjustments throughout the development process. This methodology has been selected for the development of the mobile application due to its suitability for this social media project.

The agile software development methodology is the ideal methodology for the development of the University of Westminster Student Application (UoWsa), this is due to the fact that the mobile application has a dynamic and changing requirement, such as adding and removing features based on the feedback gained from the end users, this methodology allows for flexibility in responding to the changes and incorporating them into the development process. Furthermore, the methodology provides iterative development to assist building and testing small components of the application, allowing for swift detection of problems and make responsive changes.

The mobile application is a smaller scope project with a very specific defined set of features with the requirements in place (Subchapter 4.3.8 Refined List of Requirements), this provides a clear management of development process in testing ideas and remove unnecessary features rapidly. Agile methodology also emphasises collaboration and communication between stakeholders and end user, this helps to ensure all parties are on the same page and the app is developing accordingly to the requirements set. Trial and error are supported in development, this is helpful in testing throwaway prototypes and features to evaluate which features, tools and processes works well with the end user, therefore the developer can identify errors and problems and adjust or make changes quickly.

Despite the advantages of Agile methodology, the disadvantages are:

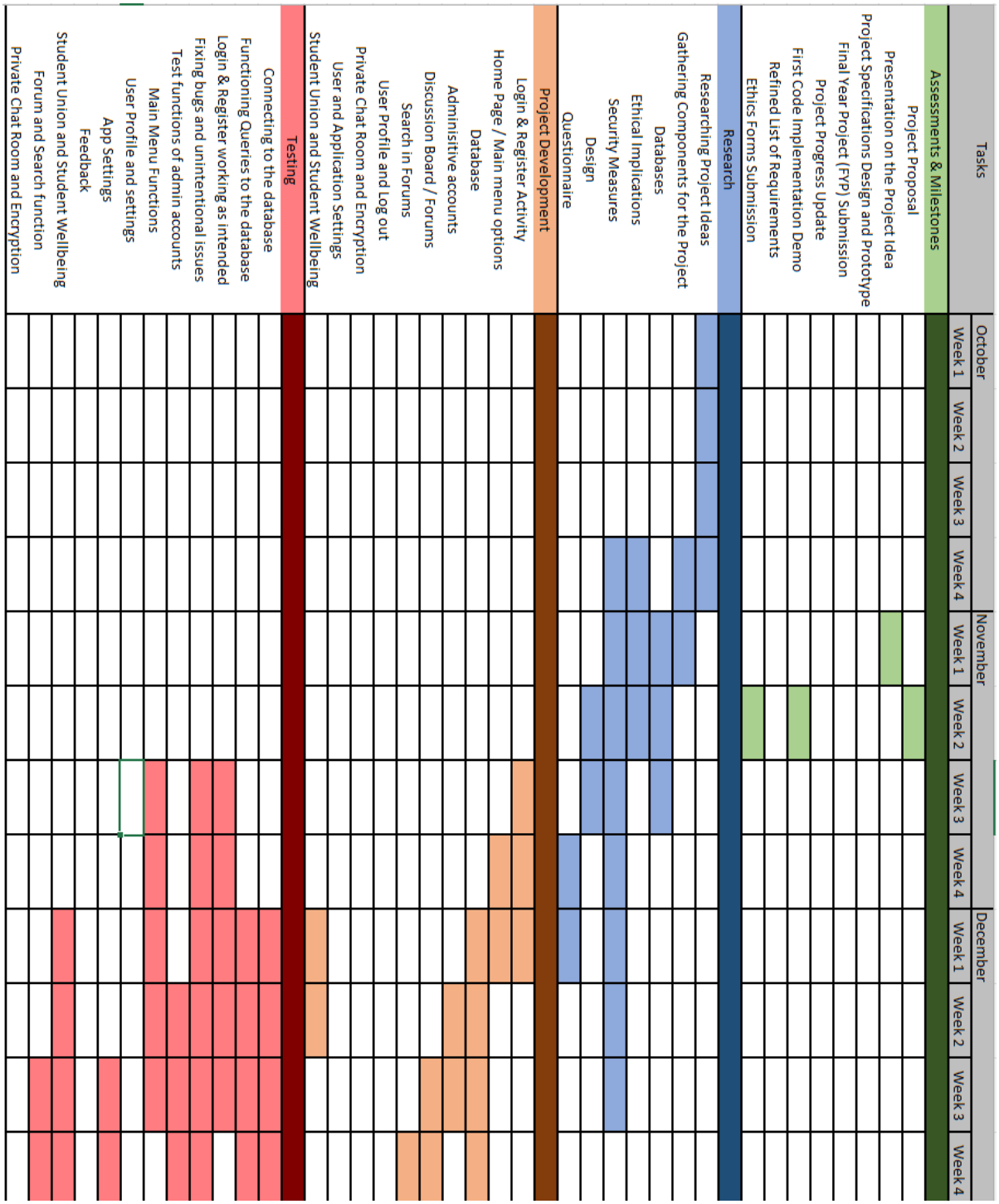
- Lack of upfront planning
- Dependence on end users'
- Cost
- Require experience developers

Agile methodology relies on swift iterative development cycles to deliver the project quickly. However, this approach with bring a lack of upfront planning, this could lead to unclear and misunderstandings from stakeholders and end users. Also, it relies heavily on end user and relational personnel involvement and feedback, therefore if they're unwilling to participate or provide feedback in an untimely manner, it would cause worthless contributions.

The cost of the agile methodology can be more expensive in comparison with the traditional Waterfall method due to the requirement of a highly skilled team or developer. Nonetheless, although there are some disadvantages in the Agile methodology, the merits it provides gives me confidence in the project in development.

Note: For more information on the life cycle stages of the project, please refer chapter 8 Conclusion and Reflections.

4.2 Gannt Chart



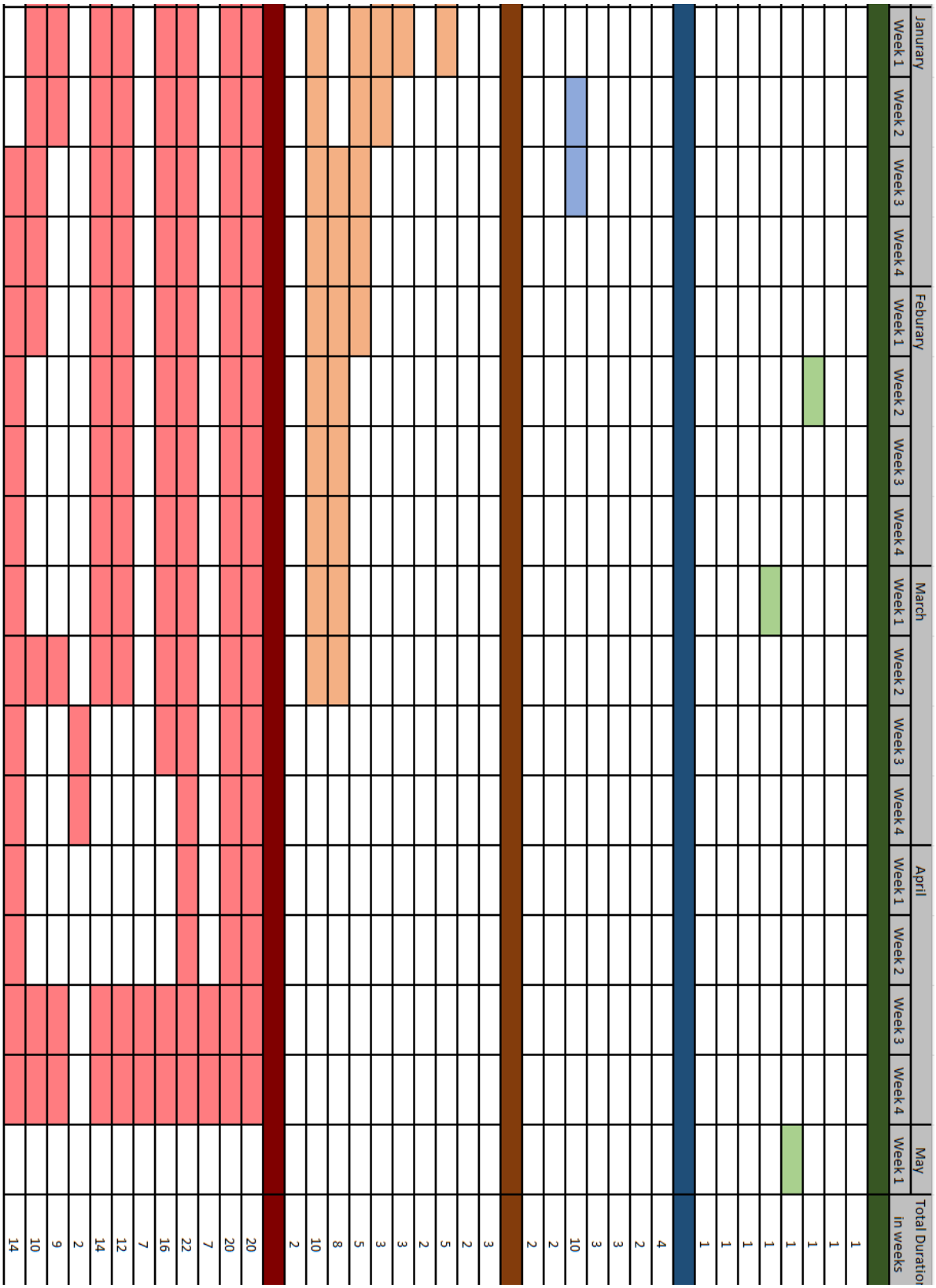


Figure 16 Gannt Chart

In figure 16 Gannt Chart, it illustrates the planning / research, development, and testing process in the Agile methodology stages of requirements, design, implementation, and testing.

4.3 Requirements

The requirement stage in this project is where the developer works with the stakeholders to define the scope of the project along with the specific features and functionalities the proposed project must incorporate. This involves gathering of user requirements and feedback to discover the goals and needs from the target end user.

4.3.1 Initial Drafted List of Requirements

The initial drafted list builds upon the very first stages of research and which will be refined into functional and non-functional requirements in subchapter 4.3.8 Refined List of Requirements.

Essential List:

- The mobile application is compatible with all Android devices regardless of size and orientation.
- The implementation of Terms of Service (TOS).
- Register and Login menu security implementation**.**
- Ensure the security measures comply with GDPR.
- Create a forum / discussion board for all users.
- Create an area for the application to showcase what the student union has to offer including clubs, societies, and student wellbeing.
- Navigation is user friendly and the ability to input and post effectively.
- Create and implement a database to connect users chatting / texting privately with security.
- Implement administrative accounts.
- Differentiate user settings and application settings. User settings include change of username, profile image and privacy settings. Whereas app settings may include accessibility and theme such as, light, and dark mode.
- The ability to add, remove and block users.
- Log out.

Desirable List:

- Deploying End-to-end encryption.
- The ability for other users to visually see if their friends are online, away, busy offline, by using a colour icon.
- Users are able to have a status message along with their profile.
- The mobile application can send notifications.

4.3.2 Case Study

The case study is to identify if a university student social mobile application can aid students' mental health especially during a worldwide pandemic using a peer-to-peer method. In addition, many students find limitations on engaging with their fellow students using existing and external platforms, therefore this study is to also find out if an exclusive application for University of Westminster students can be a better approach.

4.3.3 Stakeholders

This stakeholder subchapter defines the scope of who the stakeholders are for this development project.

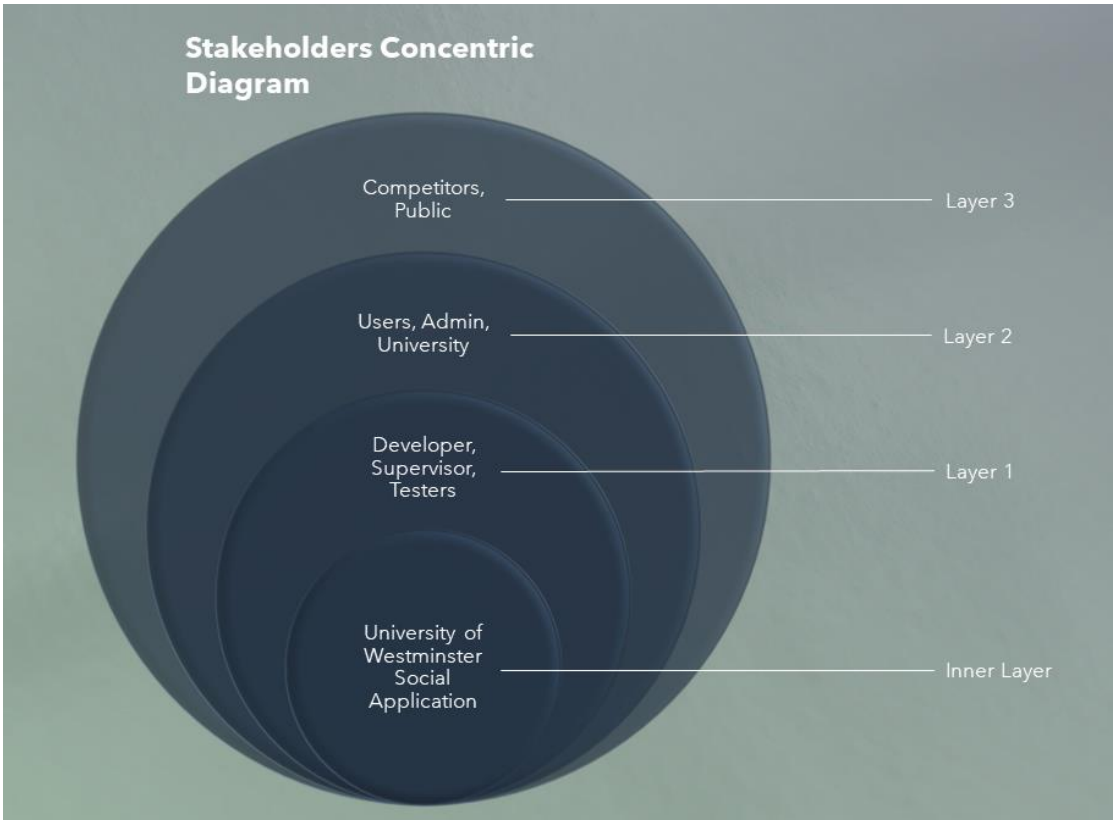


Figure 17 Stakeholders Concentric Diagram

Inner Layer: University of Westminster Social Application.

Layer 1: The direct stakeholders who are involved with the creation and development of the project.

The developer Christopher Wong, supervisor Markos Mentzelopoulos and selected students from the University of Westminster (testers) are the direct stakeholders of the mobile application. The developer is in full control of the research, development, gathering resources, questionnaire, and the testing control process. The supervisor

analyses the entire process of the project as well as contributing ideas for the developer to have an insight on possibilities of an efficient approach or requirements which needs to be met. The role of the tester has the ability to test the project with their unbiased opinions including functionality, UX design, visual interface, security and many more. This is incredibly useful for the developer as the testers are able to evaluate the advantages, disadvantages, and essentials of the application.

Layer 2: The continuous stakeholders who maintains the application when the project is complete.

Once the application is completed, the administrators take control of the in-app affairs as well as providing feedback to the developer of the application for changes and additional implementation, therefore they act upon as secondary stakeholders. Furthermore, the users are also recognised as the secondary stakeholders as they would have the most interaction with the application itself and understanding the advantages and disadvantages of the application. As the application directly involves the University of Westminster; social, ethical, and legal implications will affect the institution, as the project idea is targeted for University of Westminster students. This project is also developed with university resources; therefore, the university will also be secondary stakeholders.

Layer 3: External stakeholders

External stakeholders are individuals or groups outside this project who has interests or concerns in the operation and performance of this project **(Strategy, 2021).** Competitors are considered as external stakeholders due to the rival's application being competitive in comparison to this project. Also, the public can choose to enrol at the University of Westminster and thereby have a designated student accreditation to use the application, consequently they will have access to the application and become a layer 2 stakeholder.

4.3.4 Research

Once the problem statement, aims and objectives were set, the process of finding solutions and analysing with background research for similar social media platforms as well as mobile application designs has provided a clear concept in development. Condensing the essentials and desirables gives a clear understanding on the direction of the project moving forward. Furthermore, an initial blueprint on how each Android activity would function with each other was created Figure 18 University of Westminster Student Application Conceptual Blueprint.

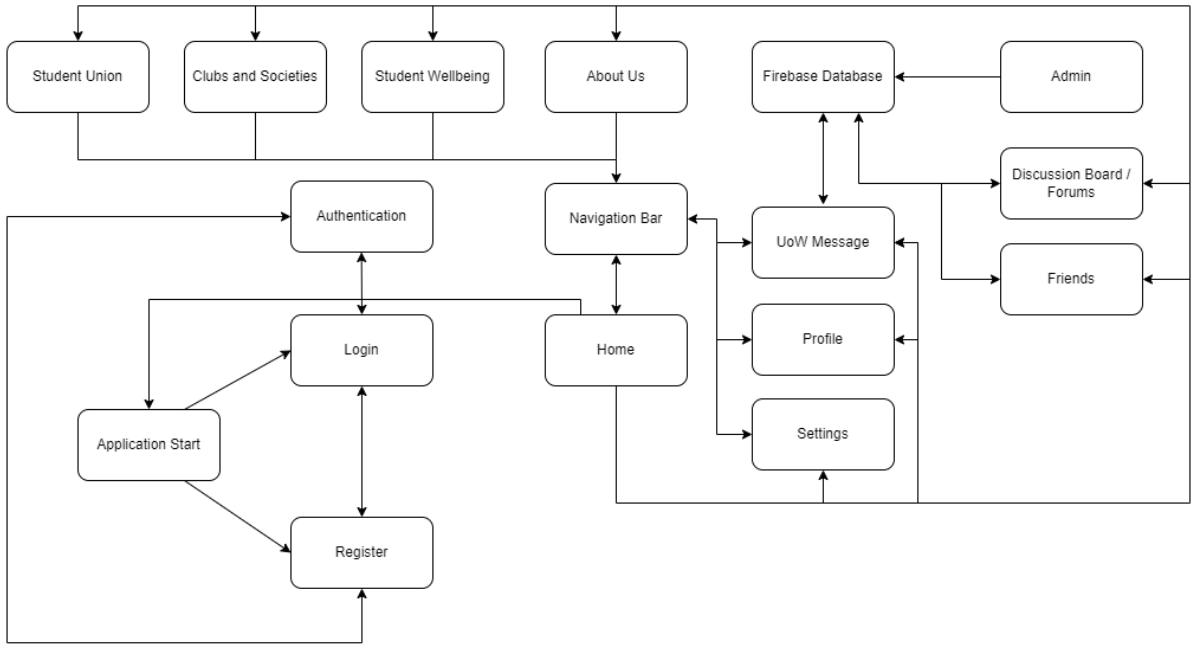


Figure 18 University of Westminster Student Application Conceptual Blueprint

Using the conceptual blueprint Figure 18 University of Westminster Student Application Conceptual Blueprint, the relationships between each section of the application was rebuilt into a concise method in order to find a greater approach to find technical alternatives and a plan to tackle conceptual arrangements which cannot be met Figure 19 University of Westminster Student Application Concise Blueprint.

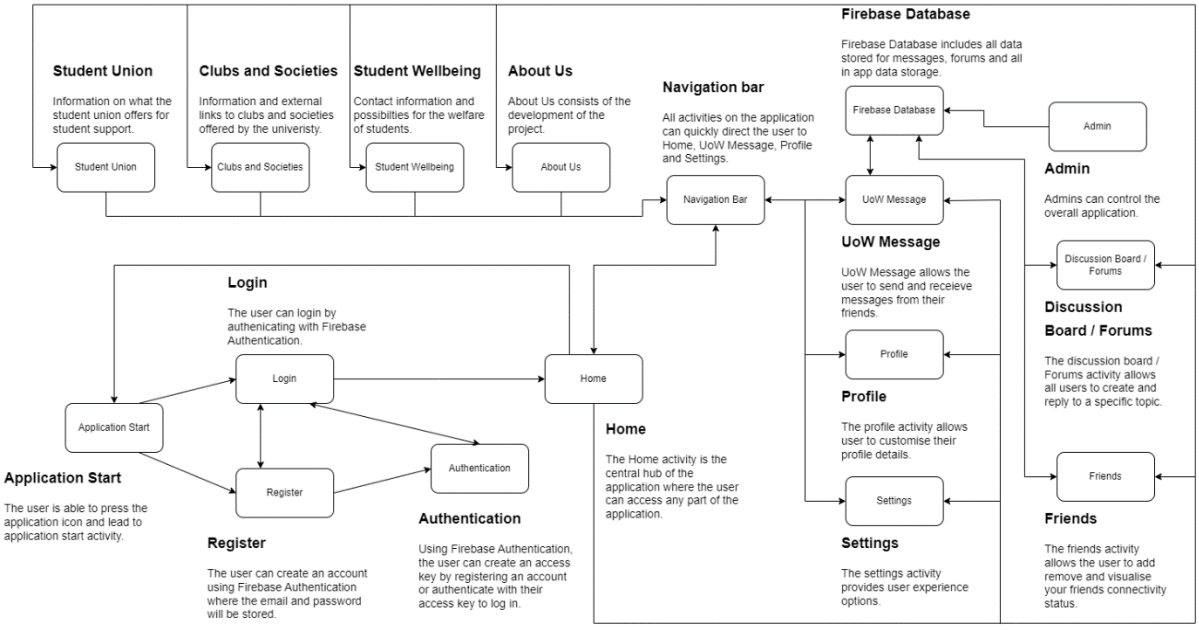


Figure 19 University of Westminster Student Application Concise Blueprint

Table 13 University of Westminster Student Application Concise Blueprint

Blueprint Name:	Description
Application Start	The user is able to press the application icon and lead to the application start activity.
Login	The user can login by authenticating using Firebase Authentication.
Register	The user can create an account using Firebase Authentication where the email and password will be stored.
Authentication	Using Firebase Authentication, the user can create an access key by registering an account or authenticate with their access key to login.
Home	The Home activity is the central hub of the application where the user can access any part of the application.
Navigation Bar	All activities on the application can quickly direct the user to Home, UoW Message, Profile and Settings.
Student Union	Information on what the Student Union offers for student support.
Clubs and Societies	Information and external links to clubs and societies offered by the university.
Student Wellbeing	Contact information and possibilities for the welfare of students.
About Us	About Us consists of the information on the development project.
Firebase Database	Firebase Database includes all data stored for messages, forums, and all in app data storage.
Admin	Admins control the overall application.
UoW Message	UoW Message allows the user to send and receive data.
Profile	The Profile activity allows user to customise their profile details.
Settings	The Settings activity provide user experience options.
Friends	The Friends activity allows the user to add, remove and visualise your friend’s connectivity status.
Discussion Board / Forums	The Discussion board / Forums activity allows all users to create and reply to a specific topic.

4.3.5 Prototyping Questionnaire

The participants targeted were students from the University of Westminster along with academic adults who has experienced the university life or used a social platform dedicated for students. This provides slightly more accurate results in comparison to data gathered by the mass public. The questions asked relates to the requirements and preferences of the participants in order to find out the essentials, luxury, desirables, psychological choices of aesthetics and necessities for the project. The data received was precisely 24 responses, therefore the data supports the case of unbiased opinion. The selected answers will be analysed, and any requirements will allow the developer to evaluate and revise on the essentials for users. The questionnaire consists of multiple choice, checkboxes for multiple

answers and multiple-choice grid to allow the participants to rank the most essential and desired to the least. The questions are designed to give the developer an understanding on the development and actions to take going forward. This includes front-end and back-end features as well as the choice of colours trending in mobile application design (*Envato, 2021*).

The questionnaire ensures all participants acknowledges the case study as well as the project being developed at hand. All data gathered will be private and anonymous, participants identification will not be taken part of the questionnaire, participation is voluntary and if the participant would like to withdraw their submission an email address for the developer / researcher is provided within the consent Figure 20 Prototype Case Study and Consent.

Note: Please refer to Appendix VIII for consent form responses details.

A total of 11 questions were conducted with 24 responses.

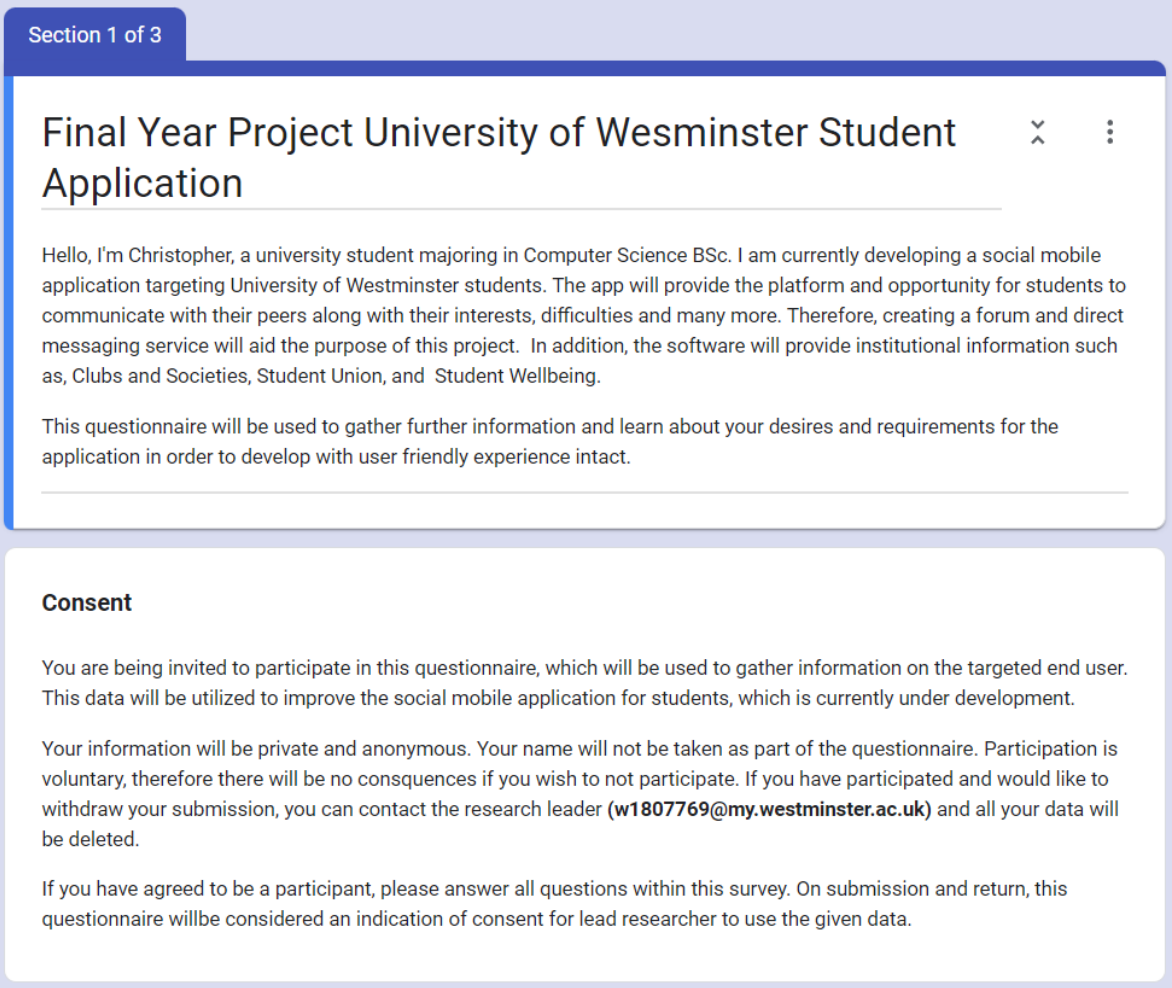


Figure 20 Prototype Case Study and Consent

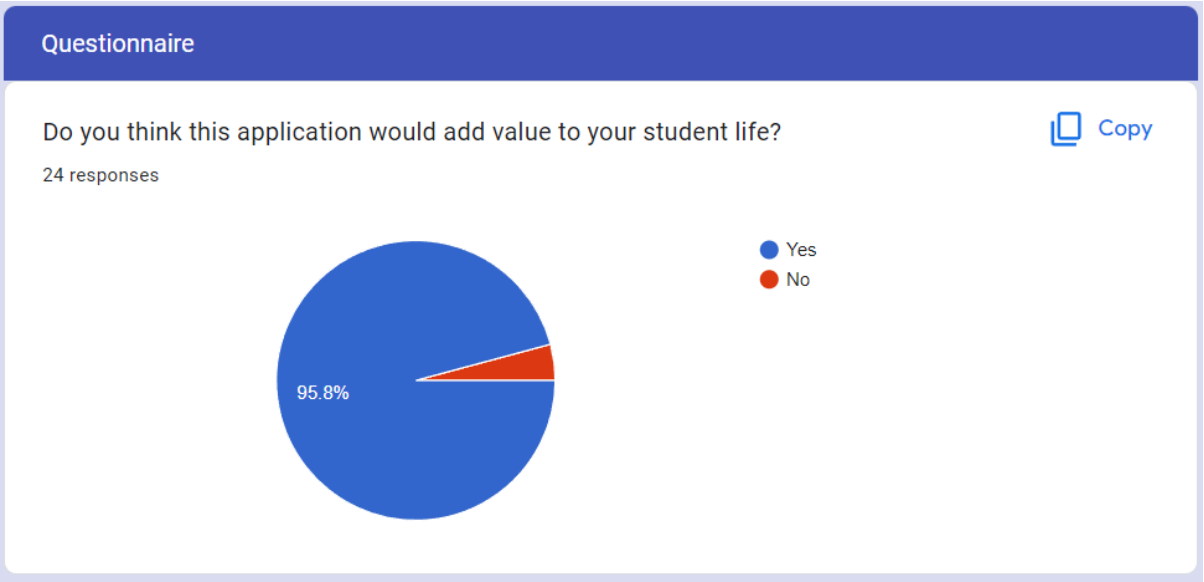


Figure 21 Prototype Questionnaire - Q1

From Figure 21 Questionnaire – Q1, 95.8% of the total participants believe a student social mobile application will add value to their life.

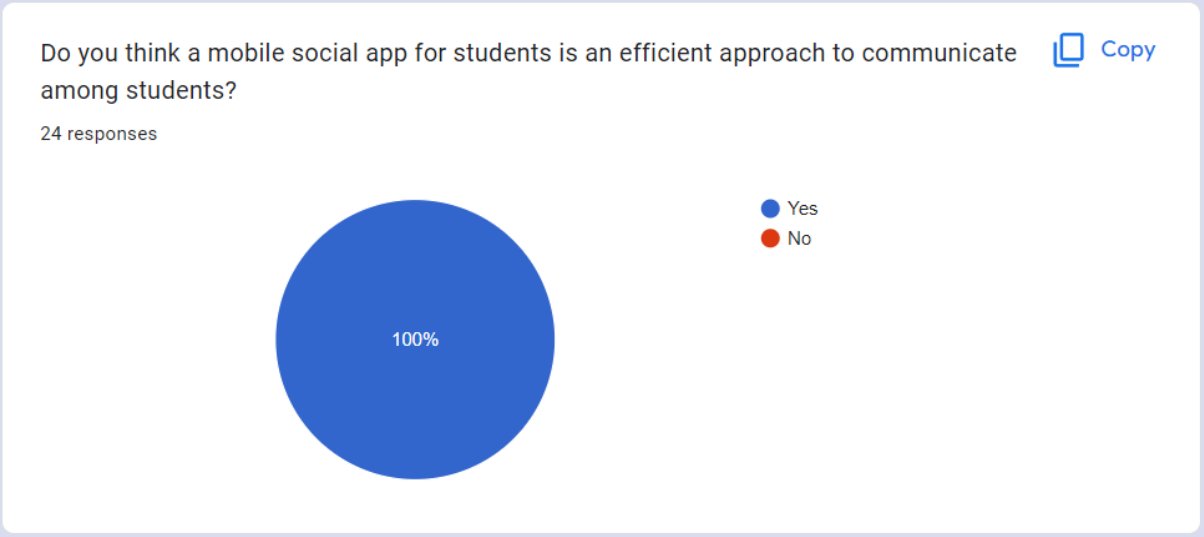


Figure 22 Prototype Questionnaire – Q2

From Figure 22 Questionnaire – Q2, all participants acknowledge the efficiency of using a mobile social app is a great approach to communicating with their peers.

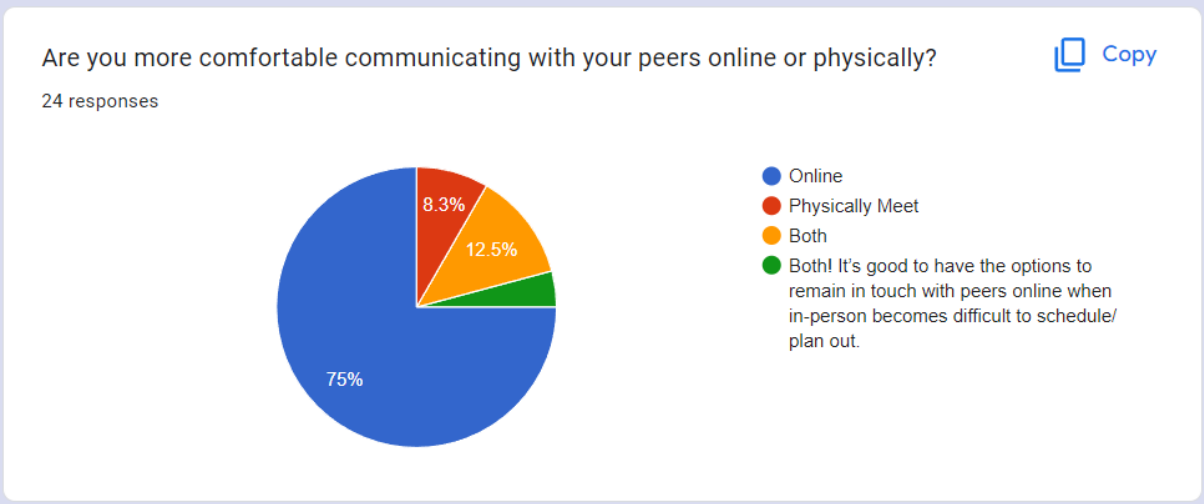


Figure 23 Prototype Questionnaire – Q3

From Figure 23 Questionnaire – Q3, three quarters of participants prefer communicating with fellow students online.

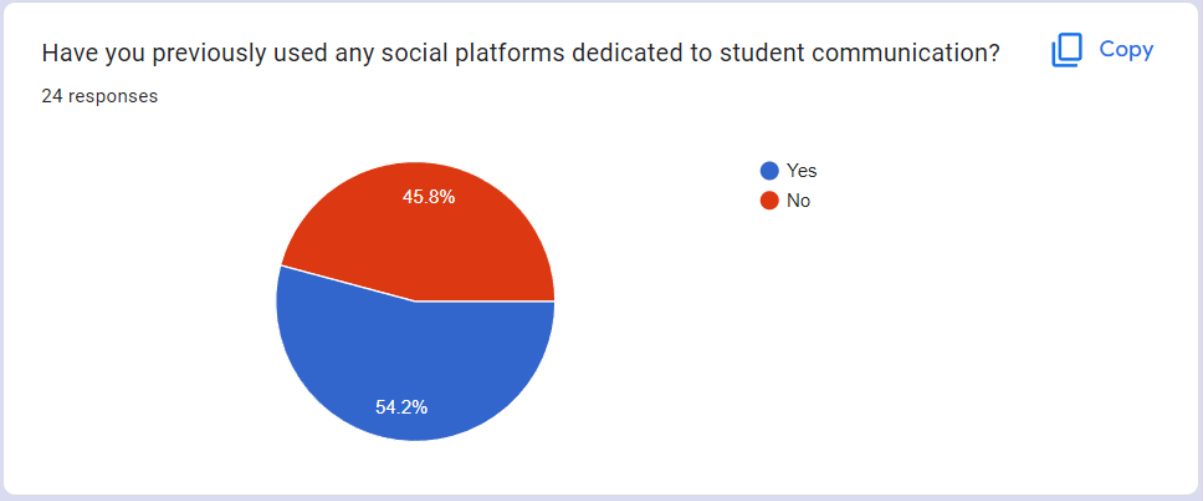


Figure 24 Prototype Questionnaire – Q4

From Figure 24 Questionnaire – Q4, there are approximately half of the participants who have previously used social platforms dedicated for students.

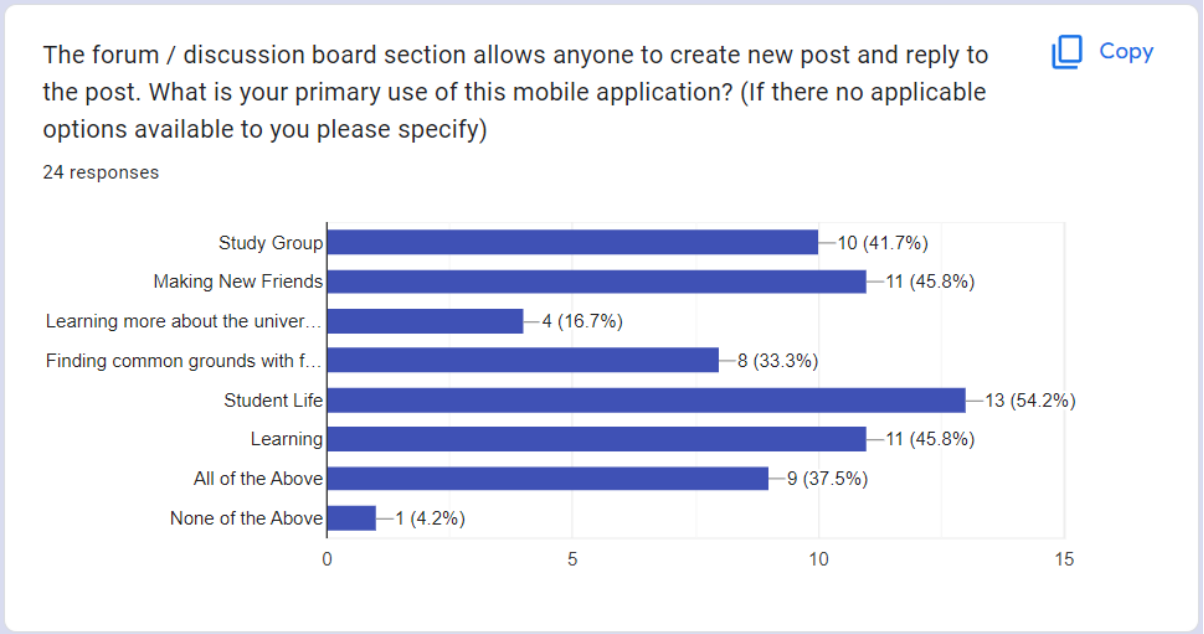


Figure 25 Prototype Questionnaire – Q5

From Figure 25 Questionnaire – Q5, it is clear study group, making new friends, student life and learning are among the highest priorities in forum section of the application. Despite this analysis, is 37.5% of participants who selected all of the above, therefore the aspect of learning more about the university and finding common grounds with fellow students should not be disregarded.

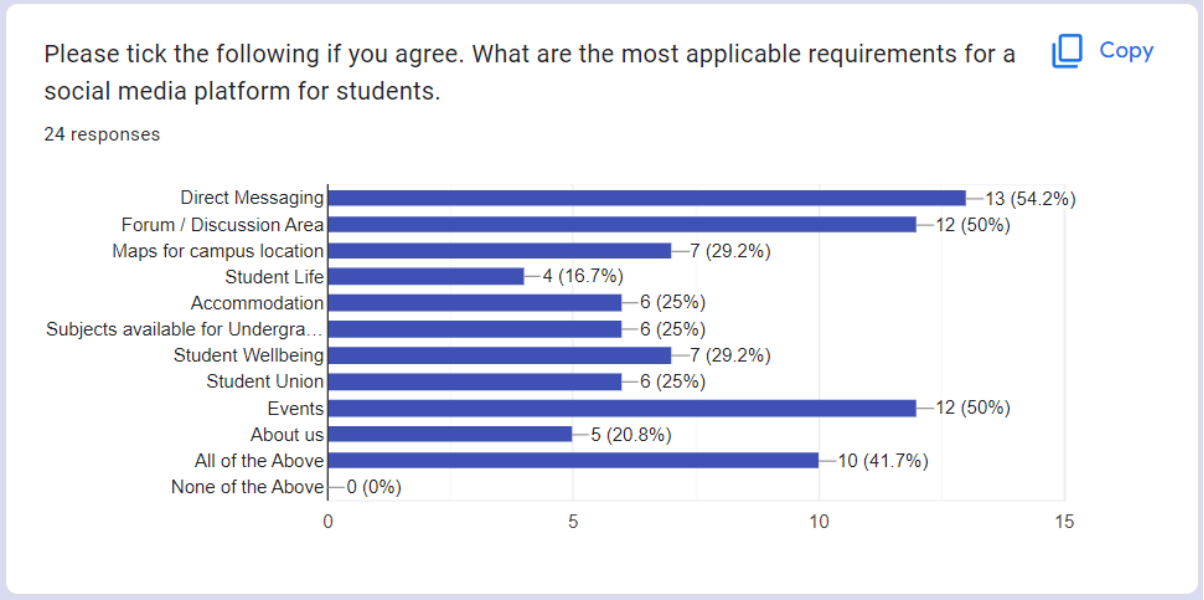


Figure 26 Prototype Questionnaire – Q6

From Figure 26 Questionnaire – Q6, the primary use of the application will be the direct messaging, forum, and events. Just under half of the participants chose all of the above, therefore there is an indication they would like the application to include many desirables as well as indicating the essential requirements of the application.

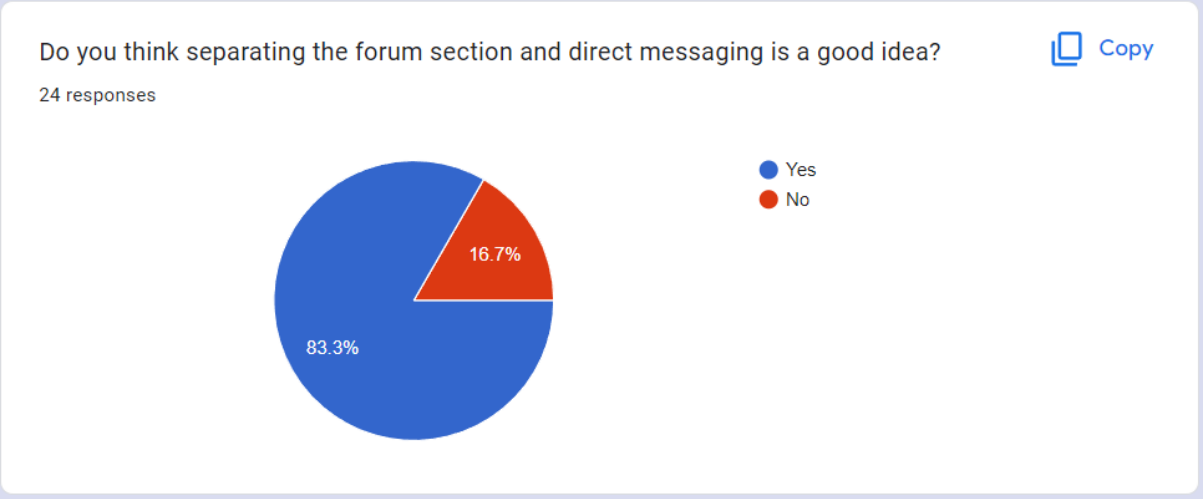


Figure 27 Prototype Questionnaire – Q7

From Figure 27 Questionnaire – Q7, it demonstrates a clear value of separating public and private messaging.

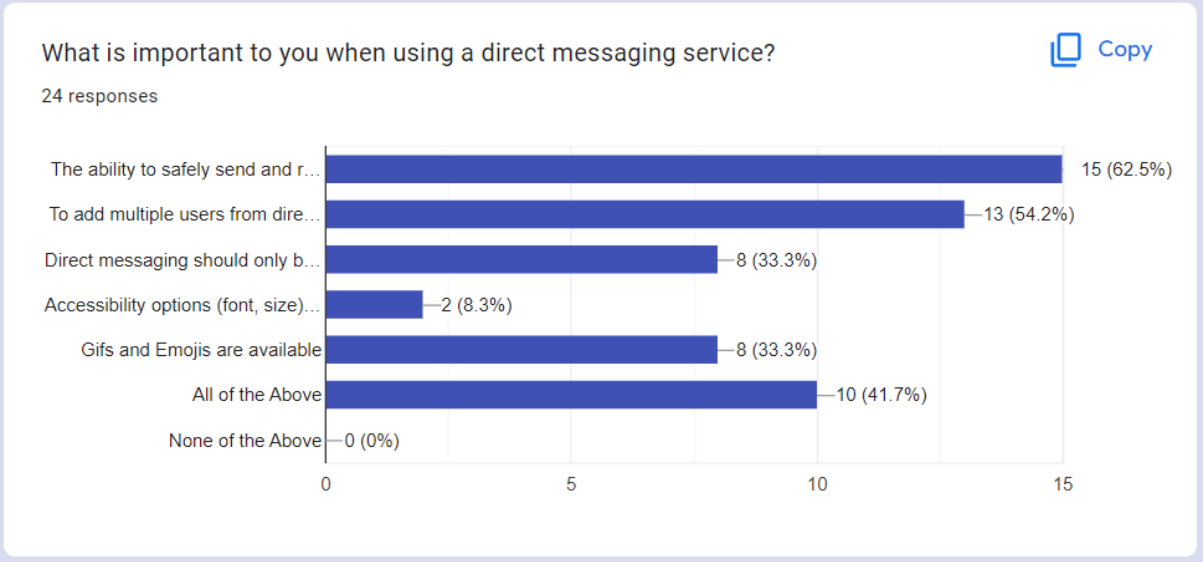


Figure 28 Prototype Questionnaire – Q8

From Figure 28 Questionnaire – Q8, the security aspect of the application is very important to the participants. The desires to add multiple students into a private chat is also highly appreciated.

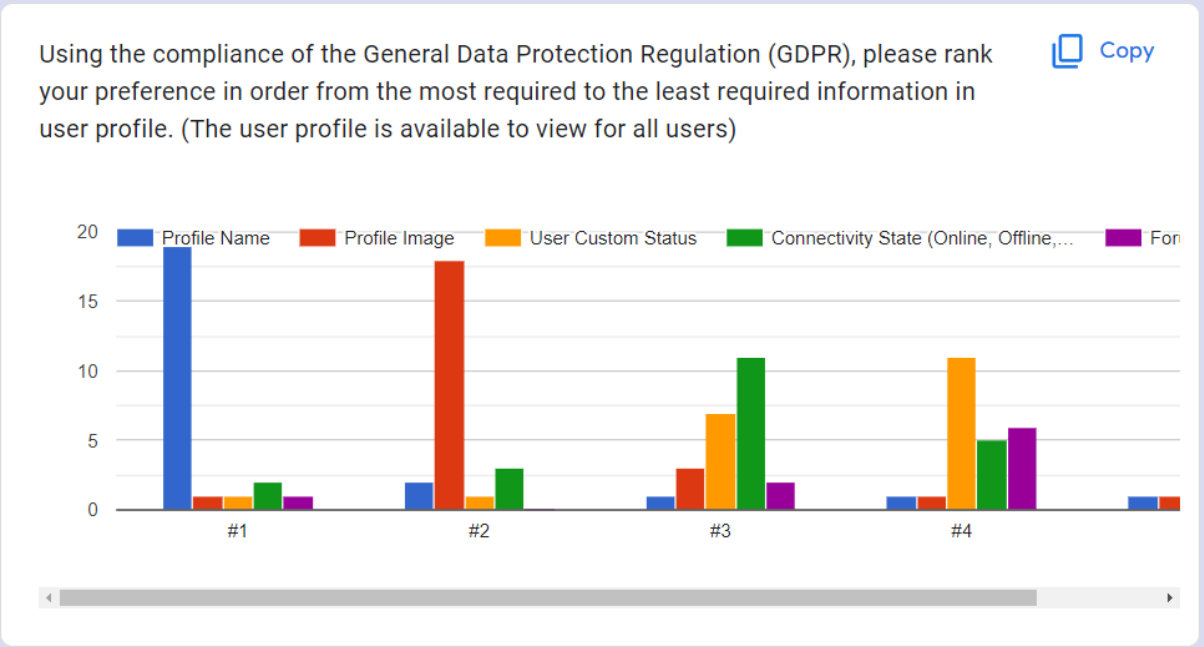


Figure 29 Prototype Questionnaire – Q8

From Figure 29 Questionnaire – Q9, it is clear the profile name is the most important entity followed by profile image, connectivity state and finally the user custom status.

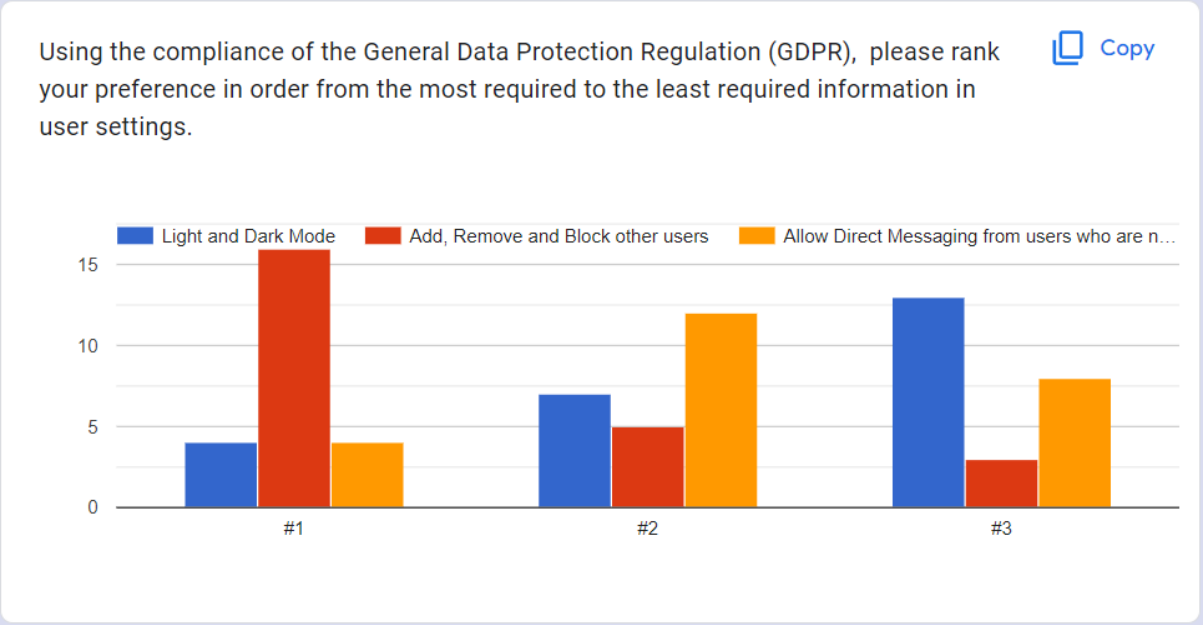


Figure 30 Prototype Questionnaire – Q10

From Figure 30 Questionnaire – Q10, the ability to add, remove and block users is a highly sort after feature as it also includes the ethical implications of the project. In addition, the option to allow direct messaging from users who are not in the user's friends list is an option which came before the switch of light and dark mode.

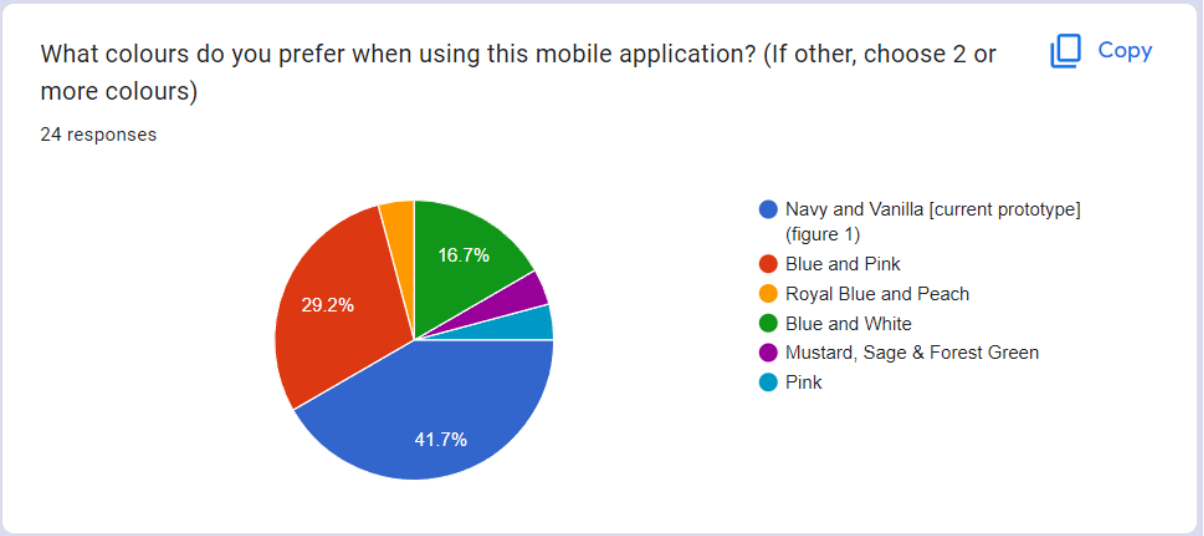


Figure 31 Prototype Questionnaire - Q11

From Figure 31 Questionnaire – Q11, the majority of participants chose Navy and Vanilla from the colour scheme.

Note: Please refer to Appendix IX for full prototype questionnaire report details.

Note: Please refer to Appendix X for design colour options.

4.3.6 Context Diagram

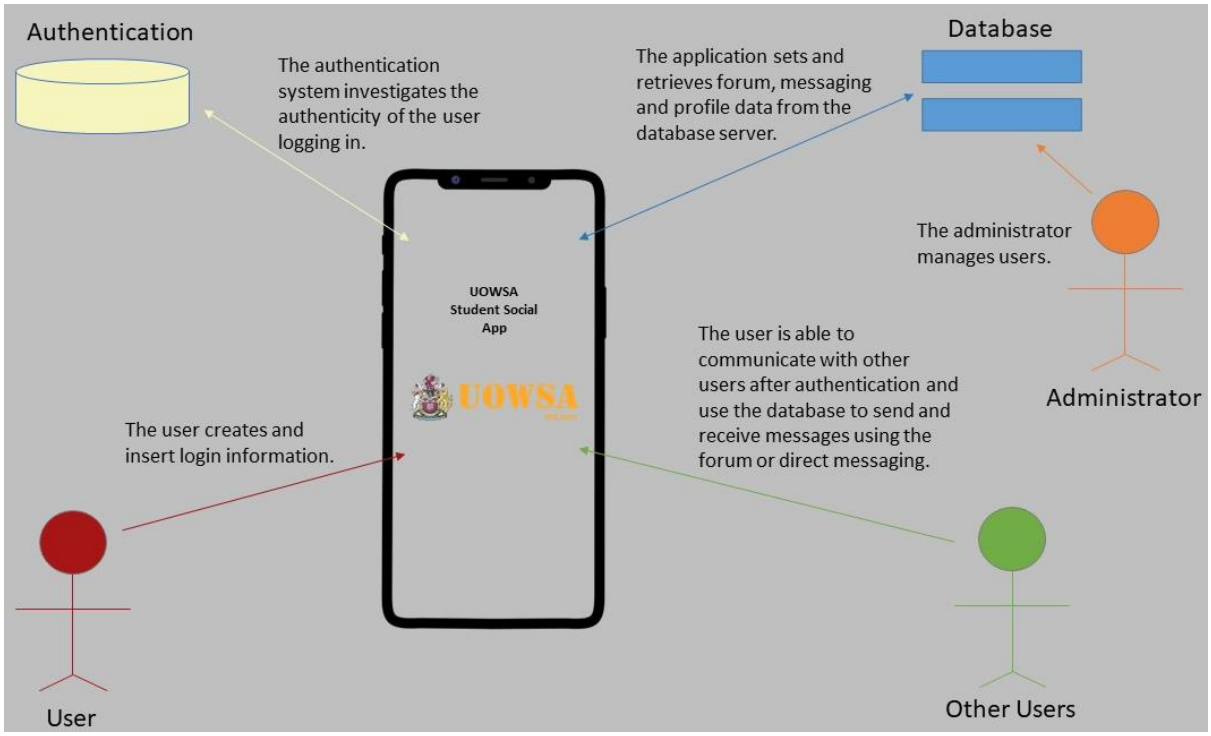


Figure 32 Context Diagram

In Figure 32 Context Diagram, it illustrates how the application factors in with external entities. Context diagrams is primarily used in system analysis and design, it is useful for visualising the scope of the project and the requirements a system (*Miro, n.d.*).

Entity 1 User: The user signs up for the application, this creates an authentication key within the system and evaluates if the user information is taken. If the data inserted is correct and available, the user can use their credentials to login to the system.

Entity 2 Authentication: The user logs into the application, the application will run through the authentication protocols to ensure the identity of the user.

Entity 3: Database: the users who write and read from the information from the forums and messaging area sets and retrieves data directly from the database. Information including their profile activity is also stored inside the database.

Entity 4 Administrator: The role of an Administrator monitors the users of the application; this includes the database and authentication. They also have the ability to monitor the application analytics from the Firebase console.

Entity 5 Other users: The user is able to send and receive messages and posts from other users, this can be achieved once the authentication and database connection is online.

4.3.7 Class Diagram

The class diagram is a high-value view of the system functionality using Unified Modelling Language (XML). The diagram consists of classes, interfaces, and objects (Visual Paradigm [1], n.d.). Each entity includes a class name, followed by a series of attributes and objects. In this project, the class diagram includes a name and interface in existence with the prototype. However, during this process, objects are still currently in development therefore, the object information currently inserted will not be excluded for fine tuning in the design stages.

Each entity along with their relationship is created for a visual purpose on how each activity and object with label, arrows and multiplicities can clearly illustrate the logic and functionality in the mobile system (Visual Paradigm [2], n.d.).

Note: Please refer to Appendix XI for Class diagrams.

4.3.8 Refined List of Requirements

The refined list of project requirements will be separated into two separate categories for functional and non-functional requirements.

(E) - Essential Requirements

(D) - Desirable Requirements

(L) - Luxury Requirements

4.4.8.1 Functional Requirements

Table 14 Functional Requirements

Priority	Requirements	Description
(E)	Starting Menu	The application should open a new window and lead to the starting menu on an Android device.
(E)	Log in	The user is able to login to the home menu.
(E)	Security Authentication Login	The use of security behind the login process.
(E)	Log out	The user is able to log out.
(E)	Sign-Up	The user is able to sign-up and creates a new entity in the log in authentication.
(E)	Home Menu	The home menu should lead to different areas of the application.
(E)	Discussion Board / Forums	The user can see all the discussions and forums posted on the Discussion board / Forums section of the application.
(E)	Create new posts in the discussion board / forum	The user can create a new post with a Title and description.
(E)	Reply to posts in discussion board / forum	The user can visually see previous posts and reply to those posts.
(E)	Delete post and reply in discussion board / forum	The user can delete any post they have created or replied previously.
(E)	Add, Remove and Block	The user can add, remove, and block from their friends list.
(E)	Previous Messages	The user can visually see all the direct messages from the sender and recipient from previous conversations.
(E)	Add new Messages	The user can create a new message with their targeted user.
(E)	Delete Messages	The user is able to delete any messages they have sent previously.
(E)	Student Wellbeing	The user can explore the options of mental welfare and support from the university.

(E)	Student Union	The user can view all other students who represents them and what role does the student union take for all students.
(E)	Clubs and Societies	The user can view all the clubs and societies offered by the University of Westminster including a description.
(E)	Settings	The user can customise their application to their preference.
(E)	Friends	The user is able to see all the friends they have added including their connectivity status.
(E)	Profile	The user can see their profile name, profile image, connectivity state.
(E)	Admin	The admin can take control the forum section of the application including delete as well as messaging users without a friend's request.
(D)	Navigation bar	The user can easily navigate to selected activities on all activities of the in the application.
(D)	Separate Admin Login area	The admin can login using alternative login area in comparison with the standard user.
(D)	Multiple users in private messaging	The ability for the user to add more users to a private messaging service.
(D)	Maps	The user can view the maps of all four campuses and directed to the targeted location.
(D)	About Us	The user can view the history of the application project including the developer of this app.
(D)	Courses	The degrees and modules with description available at the University of Westminster.
(D)	Events	The users are able to see the upcoming events offered by the university.
(L)	Explicit content filter	The ability for students to toggle a function where the application is able to determine if there are explicit content and automatically removes this for the user.
(L)	Dark Mode	The user has the ability to toggle from light to dark mode and vice versa.
(L)	Posting images and emojis	The user has the ability to send images and emojis in the discussion board and the private messaging area.
(L)	Forum counter	The user can visually see how many posts they have made in the forum section.
(L)	Accessibility options	The user can adjust the size and font of text.
(L)	Accommodation	The user can view the recommended accommodations for students at the University of Westminster.
(L)	Student Life	The user can read upon what can be expected in a student including learning and structuring their time and not exclusive to find help where it's needed.
(L)	Notifications	The user is notified of any updates.

4.4.8.2 Non-Functional Requirements

Table 15 Non-Functional Requirements

Priority	Requirements	Description
(E)	Navigation	The application is easy to navigate.
(E)	Interaction	The application is socially active.
(E)	Data Storage	The application can safely store data.
(E)	Information	The application provides efficient information for student's wellbeing, activities, and the institution.
(E)	Android Application	The user is able to run this application on any Android device.
(E)	Home Menu	The home menu should lead to different areas of the application.
(E)	Firebase Authentication	The login and sign out process is controlled and encrypted securely in Firebase.
(E)	Database Management	The application has adequate database controls.
(E)	Store relevant data	Only store relevant data to protect the user and under GDPR compliance.
(D)	Data Analytics	Data can be analysed separately from the application including user activity, posts counters for all users and many more.
(D)	Maintainability	The ability of the application to be updated and maintained a period of time.
(D)	Accessibility	The design of the application to be used by individuals with disabilities.
(D)	Performance	The application response time and data processing capabilities are optimal.
(L)	Versatility	The application can be used in different environments.
(L)	Explicit Content Filter	The application automatically detects any explicit contents within the application.

5. Design

Design is critical when it comes to packaging the product. The International Organization for standardization (ISO) defines user experience as “A person’s perceptions and responses that result from the use or anticipated use of a product, system or service” (Interaction Design Foundation, n.d.).

According to Peter Morville, an experienced information architect and user experience designer. He conducted testing and analysed well respected UX designs must include the following attributes, credible, findable, accessible, valuable, desirable, usable, and useful (Wesolko, 2016).

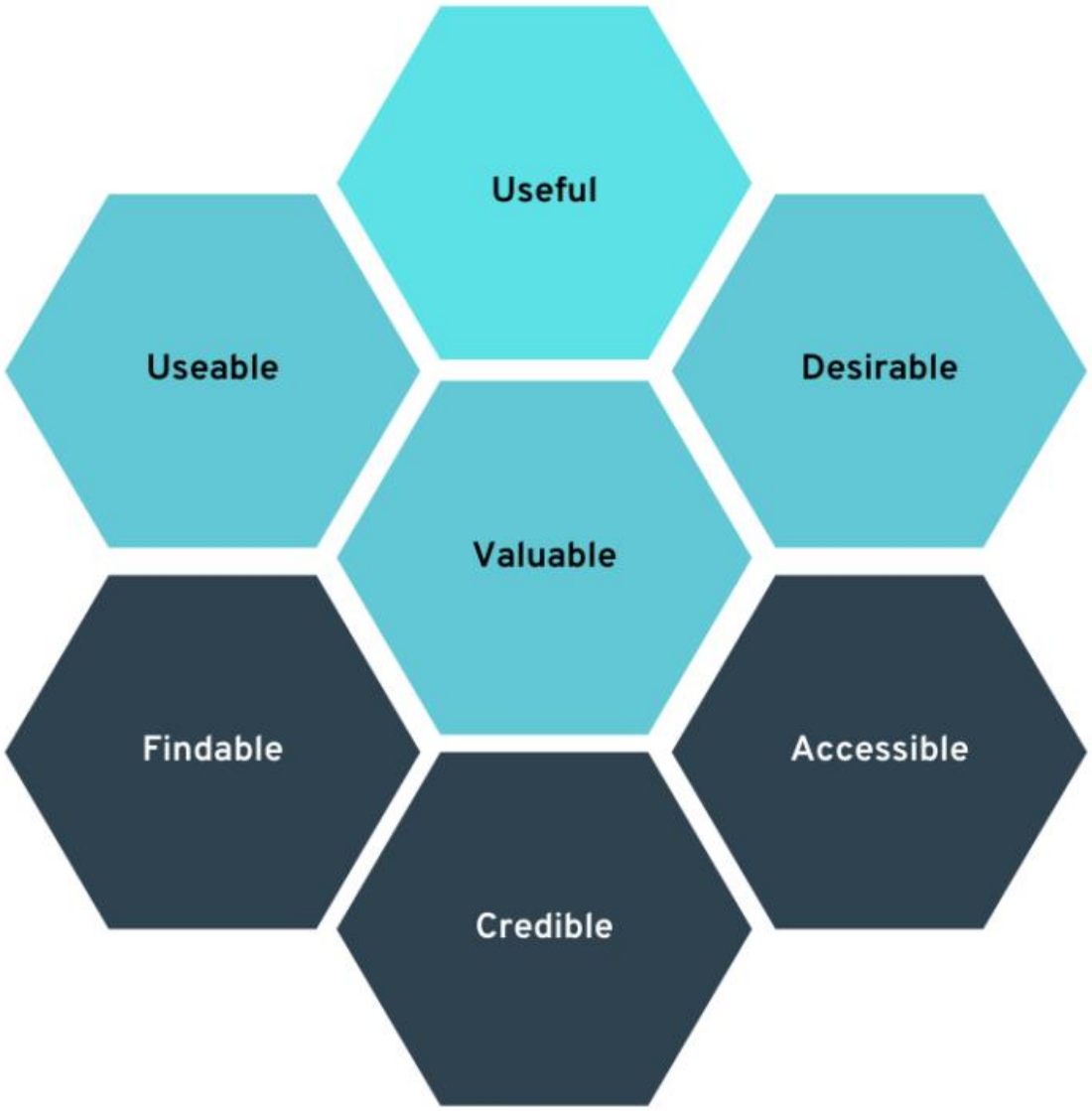


Figure 33 Peter Morville UX Honeycomb

The design section covers all of the design structures for the final version of the project. In addition, this section will also cover the tools used to aid the design aspects of implementation.

5.1 Mobile Application Wireframe

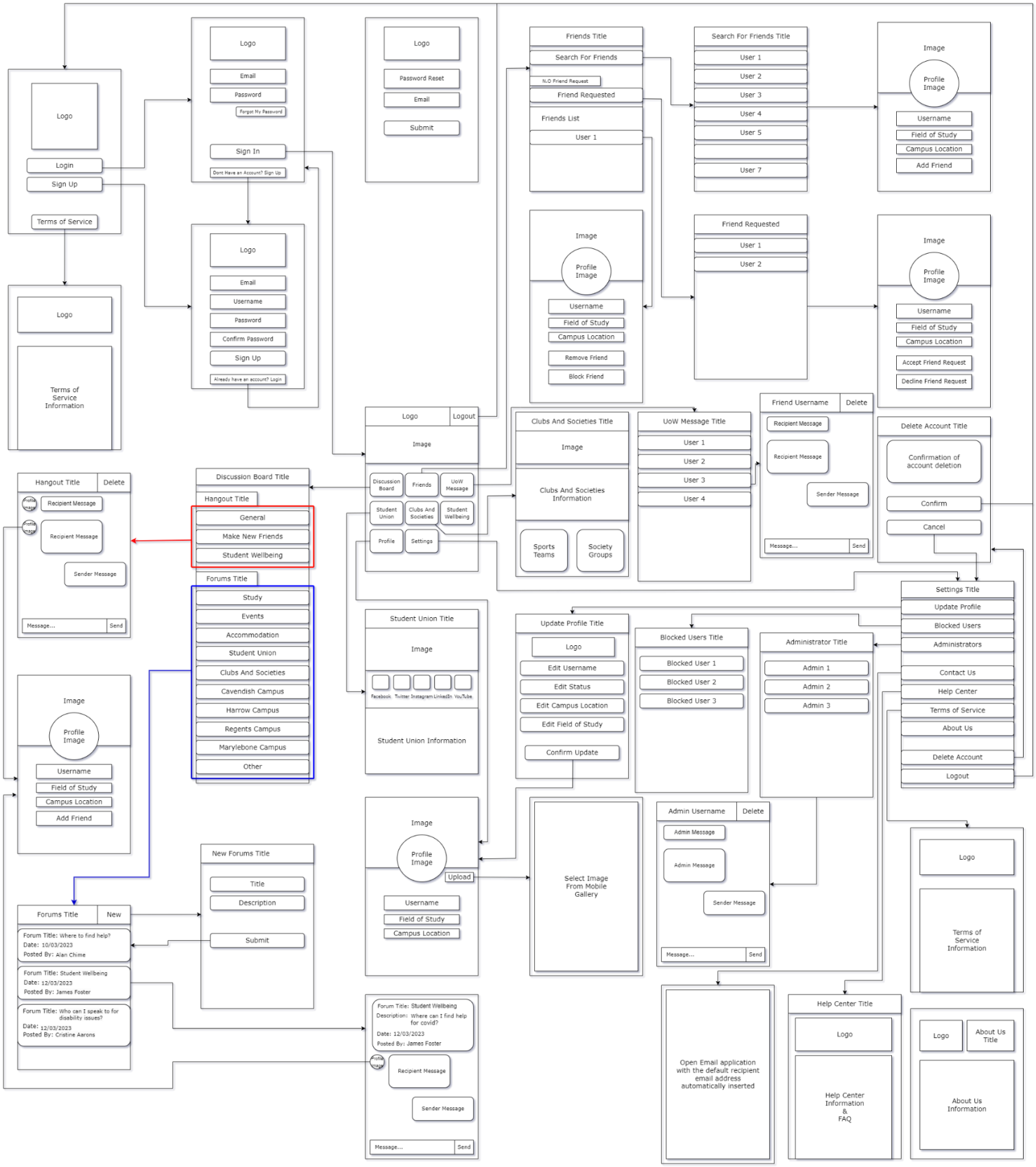


Figure 34 Mobile Application Wireframe Diagram

In Figure 34 Mobile Application Wireframe Diagram, illustrates a model of the basic structure and layout of each page and its element such as, images, text, buttons, navigation, and functionalities implemented in the project. This highlights a high degree of visualisation or blueprint during the development process, providing a clear understanding on how the app would work and ensuring it meets a high-level standard for users. Furthermore, the use of this diagram also demonstrates its capability of refinement and whilst gaining feedback in the earlier stages of development.

Table 16 Mobile Application Wireframe Table

Application Sections	
Start-up	Welcomes users to the application, this will prompt the user to select the login, register or terms of service page.
Terms of Service	The terms of service integrated to the application where the user can read upon the rules of the application. Note: this can be viewed in Appendix VIIII. The terms of service can also be located in the settings after the user logs into their account.
Sign Up	The Sign-up process includes text fields for the user to implement for validation. This includes username, email address, password and confirm password. The Register button will activate the users account as long as the user fulfils the validation requirements. If the user already has an account, the user can simply select the “Login” button in the “Already have an account? Login” section.
Log In	The login page requires the user to insert their registered email and password authentication details. If the user has forgotten their password, the user can select the “forgot my password” button, however if the user doesn’t have an account created with UoWsa, the user can select the “Register” button in the “Don’t have an account? Register” section.
Forgot My Password	The user will be prompted to insert their email address and select the “submit” button to confirm their password recovery process. A notification indicating an email was sent to the user to complete the password.
Home Page	The home page consists of the logout button on the top right, along with 8 other buttons where the user can branch towards, this includes, Discussion Board, Friends, UoW Message, Student Union, Clubs and Societies, Student Wellbeing, Profile and Settings.
Discussion Board	The discussion board are separated into two separate lists. Hangout and Forums. Hangout is indicated in a red box (Figure 34 Mobile Application Wireframe Diagram), where General, Make New Friends and Student Wellbeing buttons directs to the group chat section of the application. Whereas, Forums highlighted in a blue box (Figure 34 Mobile Application Wireframe Diagram), directs users to the

	forums section of the discussion board. The forums involve, Study, Events, Accommodation, Student Union, Clubs and Societies, Cavendish Campus, Harrow Campus, Regents Campus, Marylebone Campus and Other.
Hangout	The hangout section allows all users on the platform to conduct group chatting. The users' messages will be shown on the right side and all other users on the left. When another user messages shown on the left, their profile image will be shown, information where that specific user updated on their profile will be indicated. The user can also add specific users using this method on the group chat. The user can select the bottom left text field area to insert their message and select the send button at the bottom right. Selecting the delete button at the top right will delete all of the user's messages, selecting a specific message will delete that specific message.
Forums	The forums section contains a level of filtering using the titles assigned. The user can select the new button on the top right to create a new forum. Alternatively, the user can view the available forums already created by other users. Selecting a specific forum of interest, will direct the user to the custom forum page.
New Forum	The new forum page requires the user to insert both Title and Description prior to submission. This ensures all users can have clear view on the information shared.
View Recipient Profile	When the user selects a profile image from the hangouts or custom forum pages, the user can view all of the information on the user presented to them. This could vary depending on what information was contributed by that specific user. This may include optional information such as, status, field of study and campus location, the minimum information the user will receive is the targeted user's username and profile image.
Custom Forum	A custom forum will include the users profile image, the username of the user who created the forum as well as date, title, and description. The user can view their messages on the left and view all other messages posted by other users on the left. The user can also view other users' profile by selecting the specific user's profile on the left side of their message. The messaging section is on the bottom left with the send button on the bottom right. Selecting the delete button at the top right will delete all of the users' messages, selecting a specific message will delete that specific message.
Friends	The friend's section will include a list of friends with their connectivity status where those users were added and accepted, however if the user would like to view, remove, or block the friends, the user can simply select the targeted friend user. The user can also select the two available buttons on the page, search for friends or friends requested.

	The user can select the search for friends to find their targeted user, alternatively if there are friend requests along with the number of friend requests.
View Friend Details	The view friend details page allows the user to view the latest information updated by the friend selected by the user. There are also two buttons where the user can remove the selected user or block them.
Search For Friends	The search for friends will shows a list of users on the platform along with their username, status, and profile image. When the user selects the target user the user will be diverted to the view recipient profile page.
Friends Requested	A list of friend request will be indicated on which user has added you as a friend. Selecting the specific friend will direct the user to accept/ decline friend request page.
Accept / Decline Friend Request	The accept / decline friend request page will show the profile page of the user who added the user. Two optional buttons are indicated, if the user selects the accept friend request a new friend will be indicated on the friends list on the friends' page.
UoW Message	The UoW Message page shows a list of users and their connectivity state, online or offline. Each user will be indicated with their profile image, username, and status. Selecting a specific user will open the direct messaging page.
Direct Messaging	Once the user selected a specific user from the UoW Message page, the target user's username will be the title at the top of the page. The user can view their messages on the left and the target user on the left. A messaging section to open the keyboard is at the bottom left along with the send button on the right. Selecting the delete button at the top right will delete all of the users' messages, selecting a specific message will delete that specific message.
Student Union	The student union holds brief information about what the student union offers. There are 5 separate icon buttons including, Facebook, Twitter, Instagram, LinkedIn, and YouTube. Selecting any of these buttons will direct the user to their targeted platform for University of Westminster Student Union.
Clubs And Societies	The Clubs and Societies provides all of the activities and groups pre-created by the Student Union. The user can read upon general information the what the clubs and societies offer. There are also two selective buttons, society groups and sports teams. Selecting any of these two buttons will direct the user to the UWSU society groups or UWSU Sports Teams.
Student Wellbeing	The Student Wellbeing includes information on how the university support its students. There are also five buttons including, Wellbeing resource page, Feeling Good App, Report and Support Incident, Five Ways to Wellbeing and Disability learning support. All of these buttons direct the

	user outside the application to the selective web pages for more information and contact information.
Account Profile	The Account Profile allows the user to review their username, status, field of study, course location and profile Image. The user can select the upload button to select a new image from their mobile device. If the user would like to update other profile settings, this can be done in the settings, update profile section.
Settings	The settings provide a list of buttons with sections grouped with Account, Help and Privacy. Account includes the Update Profile, Blocked Users and Administrator buttons. Help includes the Contact Us, Help Center, Terms of Service and About Us buttons. Privacy includes Delete account and Logout.
Update Profile	Update profile allows the user to change their username, status, campus location and study course. This follows by a confirm update button which will update the user details and direct the user to the profile page. The keyword admin cannot be updated to “admin” as the username is restricted to standard users.
Blocked Users	Blocked users allow the user to review all of the blocked users the user has blocked previously. Selecting the user prompts a dialog to ensure the user would like to confirm the unblock function.
Administrators	The administrator section shows a list of administrators on the UoWsa application. Selecting a specific admin opens the admin messaging area where the user can contact user. Vice versa, if the user is an admin, it will show a list of users where both types of users can converse with each other.
Admin Messaging	Admin messaging works very similarly as the direct messaging, where the layout of messaging having the sender on the left and recipient of messages on the right. However, the purpose of this admin messaging only allows user to admin or admin to user messaging.
Contact Us	The contact us will open the mailing application on the user’s mobile device along with the email address pre inserted.
Help Center	The Help center provides a list of frequently asked questions (FAQ) where users can view questions and solutions without having to contact support for assistance. This provides a simple way for users to get around the application and learn the application if there are any misunderstandings.
About Us	The about us includes brief information on the UoWsa project.
Delete Account	The user can view details on if they are sure they would like to delete their account. Information such as reassuring the user the account cannot be recovered if it will be permanently deleted. If the user would like to use the application again, they must sign up again.

Logout	Logout is located in both home page and settings page. Selecting the Logout button will log the user out of their account and direct the user back to the application start up page.

5.2 User Flow Diagram

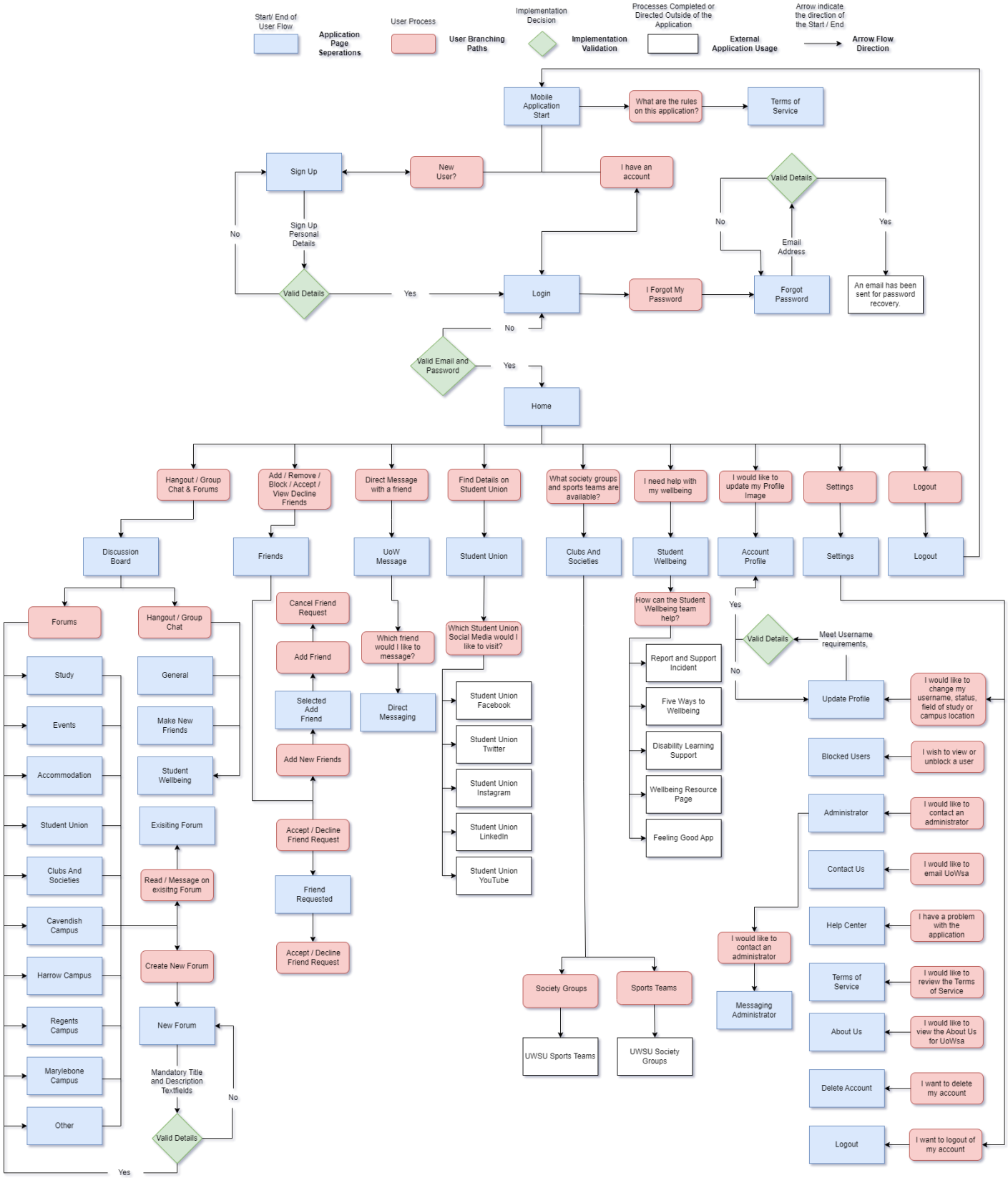
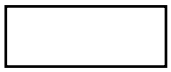


Figure 35 User Flow Diagram

The user flow diagram is a visual representation, and it depicts the sequence of pages a user would encounter whilst using the mobile application. The main advantage of the user flow diagram can illustrate and pinpoint the potential issues the user may face in user experience, troubles in continuing a process and confusion in navigation. This diagram can improve the developers plan on optimising performance and provide a clear implementation journey.

Table 17 User Flow Diagram Guide

Symbol	Title	Summary
	Application page separation	Start / End of User Flow The blue rectangle symbol represents all of the existing pages within the mobile application.
	User Branching Paths	Users thought process The red cornered rectangle symbol represents the thought process the user may come across when using the application.
	Implementation Decisions	Implementation Validation The green rhombus symbol represents the decisions made based on the implementation made by the developer in the application, which may affect the user directly.
	External Application Usage	Processes completed or directed outside of the application The white rectangle symbol represents all the links and functions which the application provides, however these activities are conducted outside the application.
	Arrow Flow Direction	Arrow indicates the direction of start and end. The black arrow symbol represents the possible directions the user may branch to an application page or validation processes.

5.3 Throwaway Prototype Designs

The core throwaway prototype interface design has been created from the inspirational design methods at Design Rush (*Design Rush, n.d.*).

Throwaway Prototype Part 1 – Register and Login Design

Aims:

- Create a basic UX design.
- Button functionalities for starting, register and login activity.
- Following the steps in Figure 19 University of Westminster Student Application Concise Blueprint.

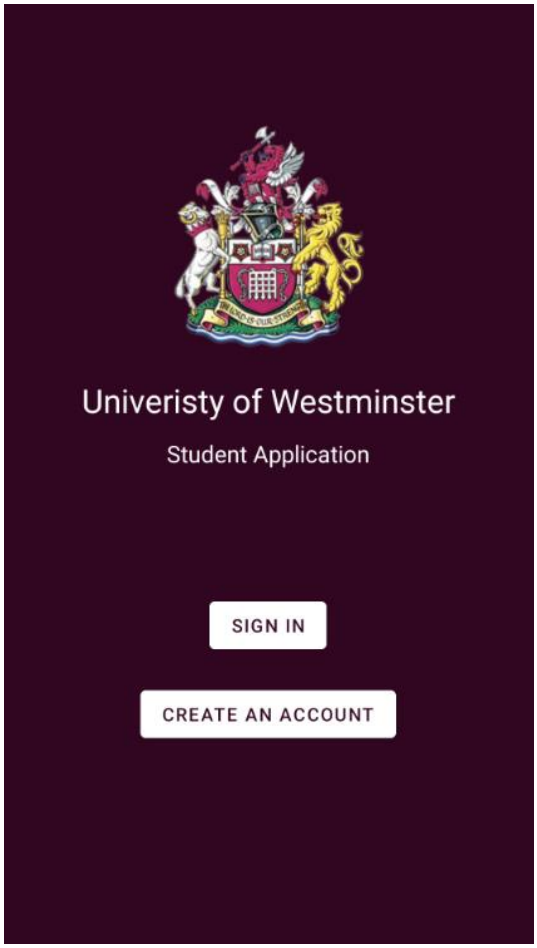


Figure 36 Starting App Initial Prototype

Throwaway Prototype 1 is made with the initial conceptual and planning process, this includes a simplistic and basic UX design along with buttons in order to transfer to other activities of this application. In the application, the user is presented with two buttons, login, and signup. If the user selects the signup button, the user will be directed to the register activity. The register activity includes the email and password

input fields and two buttons including “I already have an account” and “Sign up”. The login activity includes the email and password field and three buttons, the button includes “I don’t have an account”, “I forgot my password” and “login”. Once the user signs in the user will be directed to the home activity. The demo can be reviewed in the video link: https://youtu.be/DpZoNjPYZ5M.

Note: Please refer to Appendix XII for more throwaway prototype 1 images.

Throwaway Prototype Part 2 – Authentication Testing

Aims:

- Button functionality for starting, register, login, forgot my password and home activity.
- Create register activity with only the absolutely necessary input fields such as email and password and confirm password.
- Use Firebase authentication to test if an account is created and shown on the Firebase console.
- Once the user successfully creates an account the user will be directed to the home activity.
- If the user is logged in the logout button will bring the user back to the starting app activity.
- Create login activity with login credentials such as email and password and forgot my password.
- Create Forgot my password activity with email and submit button.
- Upon Forgot my password submission an email will be sent for password reset.
- The new password will replace the old password and can be used to login to the home menu.

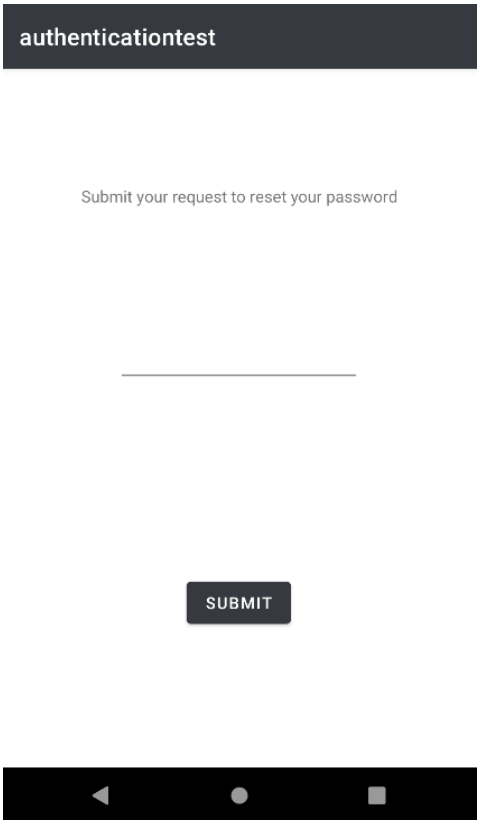


Figure 37 Authentication Test Prototype

The Throwaway Prototype 2 is created without the intention of UX designs. The purpose of this throwaway prototype is to ensure the functions on creating a user with authentication in place is functional. In the application, the starting screen is presented with the login and signup buttons. If user selects sign up button the user will be directed to the register activity in which the user will be required to insert the email, password, and password confirmation in order to successfully create an account. A text will be shown for a few seconds to the user if the user creates an account successfully or unsuccessfully. Once the user creates the account successfully, the user will automatically go to the home activity. The home activity only includes a sign out button, if the user selects sign out, the user will be logged out and be sent to the application start activity.

If the user selects login from the application start screen, the login activity will prompt the user to enter their login (email and password) credentials to authenticate and forward to the home menu. Alternatively, the user may select “Forgot my password” in the login activity, this is where the user inserts their email and submit their password reset request. Once the submission is complete, an email will be sent to their email address and prompt the user to reset the password and when the request is complete and saved. The new password can now be used to sign into the home activity. The demo of throwaway prototype 2 can be reviewed in the video link: https://youtu.be/oOjlb4P2kRk.

Note: Please refer to Appendix XIII for more throwaway prototype 2 images.

Throwaway Prototype 3 – App design (Light and Dark Mode), External Hyperlinks and Navigation Menu

Aims:

- Create multiple activities including starting app menu, register, login, forgot password, home, about us, discussion board, friends, profile, settings, society, sports, student community, student union and UoW Message.
- Ensure all buttons can direct to the targeted activity.
- Create logos and images using Clip Studio Paint with the correct resolution with consistent width and height of image.
- Ensure all objects in all activities are constraint correctly.
- Using colour xml and themes xml to create the light and dark mode for all pages.
- Create hyperlinks using image buttons.
- Create a navigation menu with image buttons including Home, Profile, UoW Message and Settings.

Figure 38 Starting App Light Mode

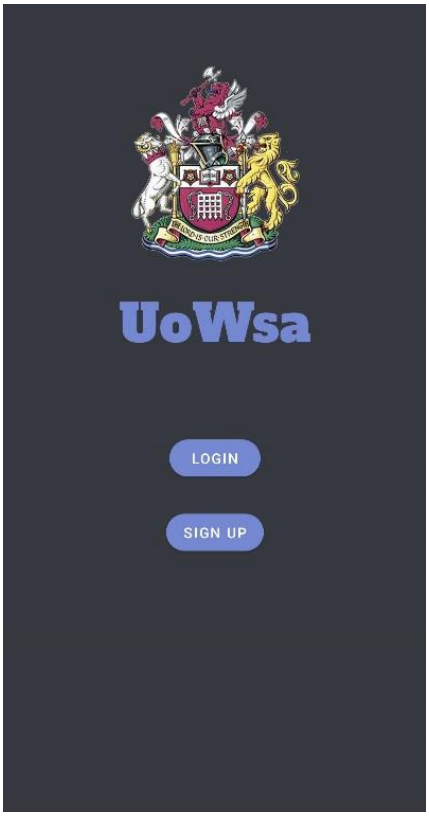


Figure 39 Starting App Dark Mode

The prototype includes all the UX designs collected from the questionnaire in section 4.3.5. This prototype is the overall idea of how the UX could potentially be implemented in the final product. The design of the application starting menu to the home menu can be represented in the throwaway prototype 1, however the design and colours has been improved upon. Once the user is logged in, they will be directed to the home menu where it would be recognised as the central hub of the application, therefore all areas of the platform after login will be presented in this activity. Furthermore, a navigation bar is implemented on all activities with the Home, Profile, UoW Message and Settings buttons, therefore the user will be able to access these activities in a swift and efficient manner. The Student Union section also presents 5 social media logo image buttons where the user can access external Student Union social platforms. Finally, if the user uses the dark mode primarily on their device, the whole application supports a darker visual. This can be reviewed in the demo video link: https://youtu.be/ozsK-MasgOA.

Note: Please refer to Appendix XIV for more throwaway prototype 3 images.

5.4 Figma Front-End Design and colour schemes

The use of Figma to draft the design for the most important pages in the refinement stages of font-end development provides organisation and clear visuals of the procedures of implementation for both light and dark mode. The Figma designs can be separated using clear indications on the typefaces used, consistent rounded colours for the home page, profile page and messaging page.

5.4.1 Typefaces and colour consistency

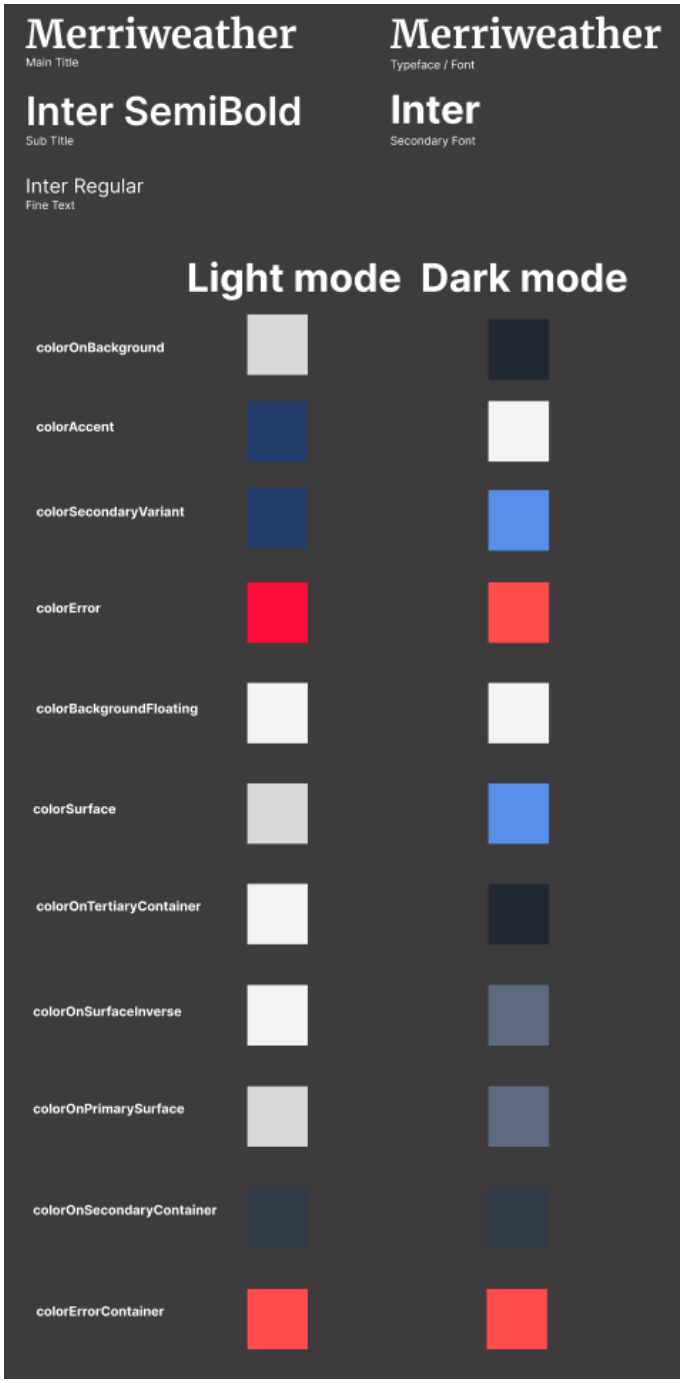


Figure 40 Typeface and colour consistency for light and dark mode

Using the colour palette for light and dark mode in Figure 40 Typeface and colour consistency for light and dark mode, the design became well-structured for implementation and refinement. The colour palette are the colours only used throughout the application. Colours set from Figma can be translated to the XML themes during implementation, which is illustrated in Figure 41 XML Themes.

```xml
<resources xmlns:tools="http://schemas.android.com/tools">
    <!-- Base application theme. -->
    <style name="Theme.UoWsa" parent="Theme.MaterialComponents.NoActionBar">
        <!-- Primary brand color. -->
        <item name="colorPrimaryVariant">@color/navy</item>
        <item name="colorOnPrimary">@color/white</item>

        <item name="colorOutline">@color/navy</item>
        <item name="color">@color/white</item>
        <!-- Secondary brand color. -->
        <item name="colorSecondary">@color/backgroundColorLight</item>
        <item name="colorOnSecondary">@color/navLight</item>


        <!-- Status bar color. -->
        <item name="android:statusBarColor">@color/systemMenuColor</item>
        <item name="textInputStyle">@color/darkModePrimary</item>
        <!-- Customize your theme here. -->
        <item name="colorOnBackground">@color/backgroundLightMode</item>
        <item name="colorAccent">@color/buttonColorLightMode</item>
        <item name="colorSecondaryVariant">@color/buttonColorLightMode</item>
        <item name="colorError">@color/colorErrorLightMode</item>
        <item name="colorBackgroundFloating">@color/navBarLightMode</item>
        <item name="colorSurface">@color/backgroundLightMode</item>
        <item name="colorOnTertiaryContainer">@color/navBarLightMode</item>
        <item name="colorOnSurfaceInverse">@color/navBarLightMode</item>
        <item name="colorPrimary">@color/darkModePrimary</item>
        <item name="colorOnPrimarySurface">@color/backgroundLightMode</item>
        <item name="colorOnSecondaryContainer">@color/pendingLightMode</item>
        <item name="colorErrorContainer">@color/colorErrorDarkMode</item>
        <item name="colorControlHighlight">@color/buttonColorLightMode</item>
        <item name="colorControlNormal">@color/onlineLightMode</item>
    </style>

    <style name="MyTextInputLayout_overlay">
        <item name="colorOnSurface">@color/darkModePrimary</item>
        <item name="colorPrimary">@color/darkModePrimary</item>
        <item name="hintTextColor">@color/darkModePrimary</item>
        <item name="textOutlineColor">@color/darkModePrimary</item>
    </style>

    <style name="DialogTheme" parent="ThemeOverlay.MaterialComponents.Dialog.Alert">
        <item name="android:colorBackground">@color/navLight</item>
        <item name="android:textColor">@color/black</item>
        <item name="android:textColorPrimary">@color/black</item>
    </style>

</resources>
```

Figure 41 XML Themes

5.4.2 Figma Home Page

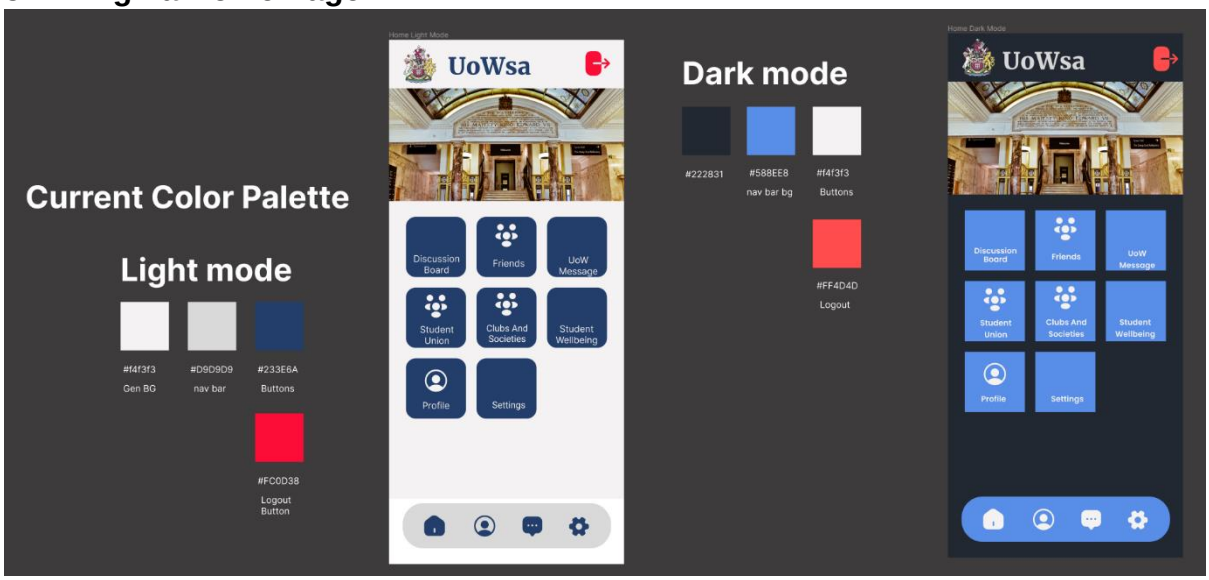


Figure 42 Home Page

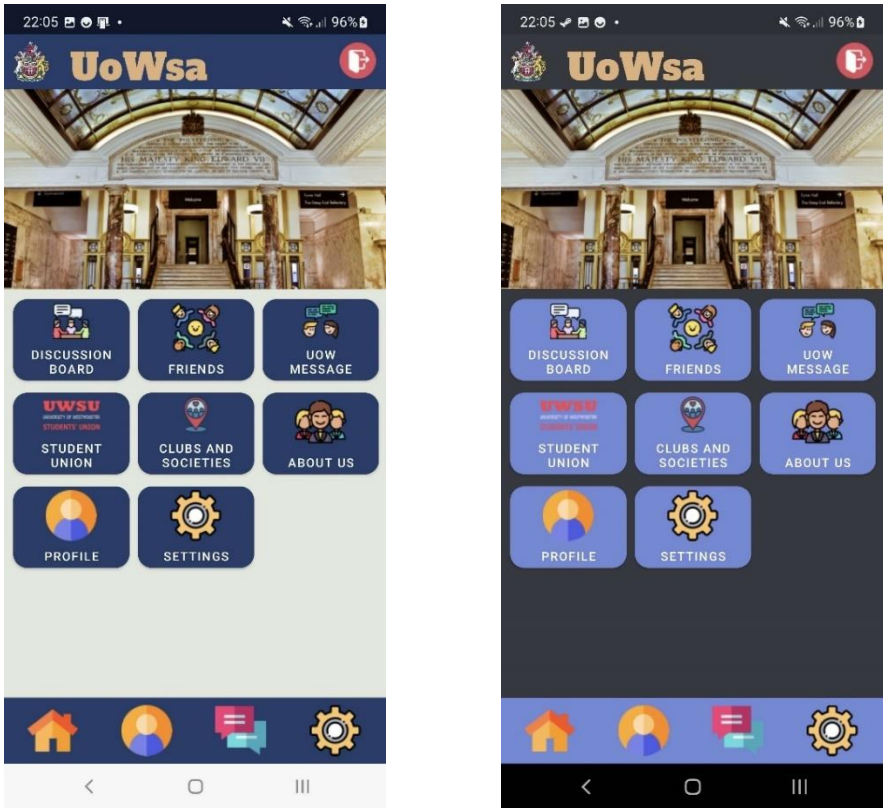


Figure 43 Prototype Home page

Figure 43 Prototype Home page illustrates the colours used in the prototype home page and it had large mix of colours in the pallet which was unappealing with a diverse number of colours used for the icons. The changes made are the icons for each button on the homepage, logout and the structure and icons for the navigation bar. The reason for changing the prototype to the Figma design was to also ensure it was visually appealing towards the target audience in University of Westminster students.

5.4.3 Figma Profile Page

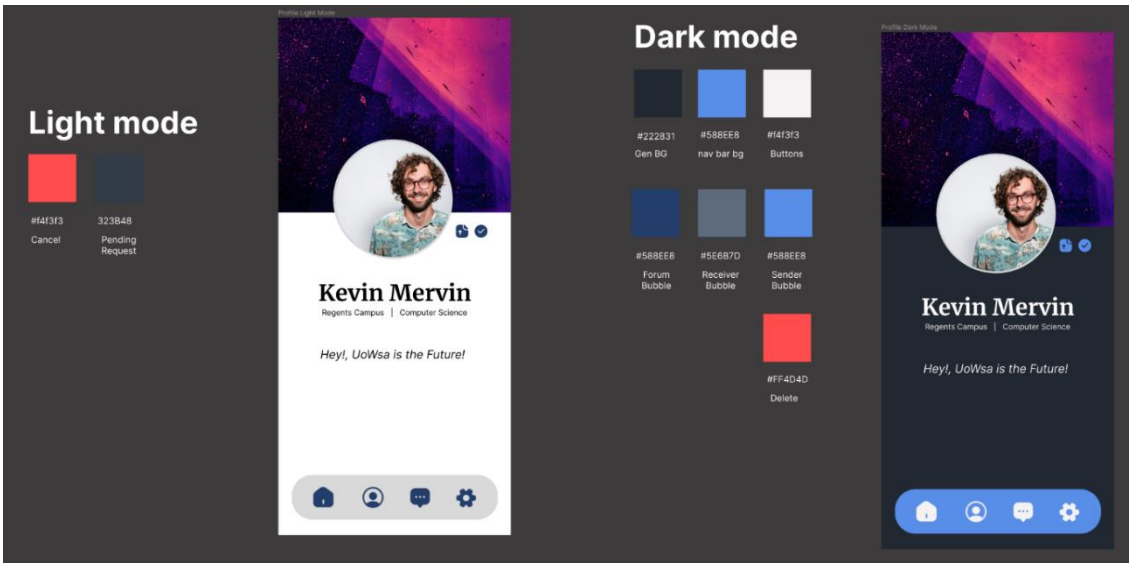


Figure 44 Profile Page

In Figure 44 Profile page, it illustrates all of the essentials for the user requirement. The page consists of the upload button where the user can change their profile image along with the confirmation button after selecting an image. The profile page includes the username and status, where the status would default to “Hey, UoWsa is the future!” if the user does not enter any status in the update profile page.

5.4.4 Figma Messaging Pages

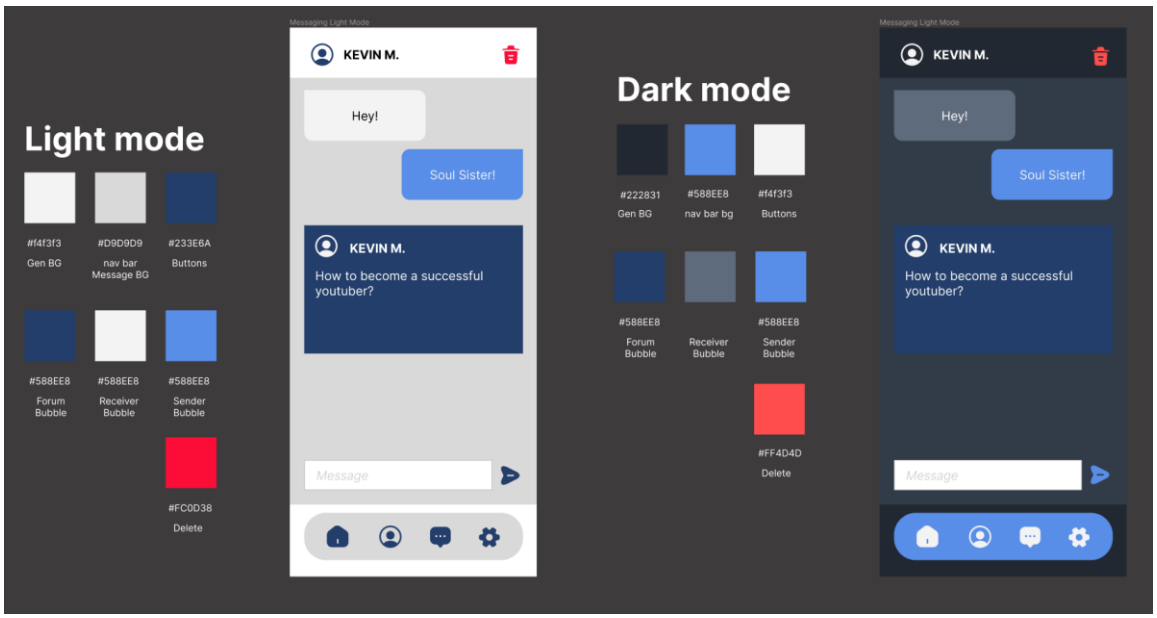


Figure 45 Messaging Page

In Figure 45 Messaging page, it illustrates all of the essentials for the user requirement. The user can read and write messages in a conversing manner with the recipient. It also provides a section on how the forum section of the application can be presented for the final design.

5.5 Use Case

The use case diagram models the relationships between the user of the application and the application system.

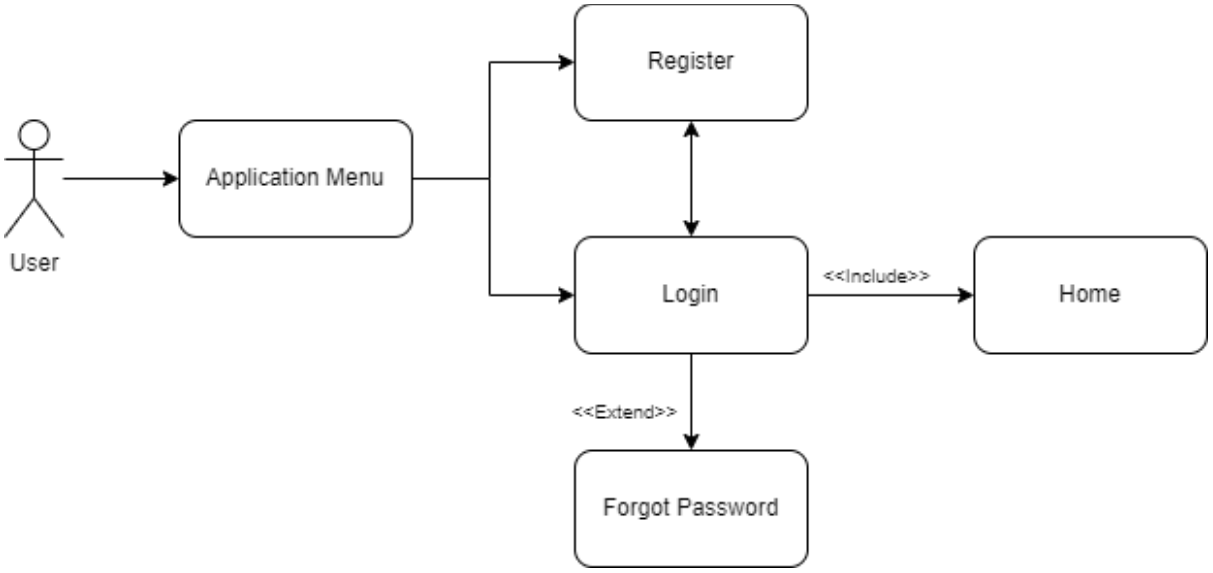


Figure 46 Use Case Diagram App Start

In Figure 46 Use Case Diagram App Start, the user launches the application into the application starting menu where there are two options given to the user. The user can choose to either login or register an account for the application. If the user chooses and succeeds in registering an account, they will be directed back to the login area. Alternatively, the user can choose to direct to the login page where the user can either press forgot my password and reset their password through email or login to the home activity.

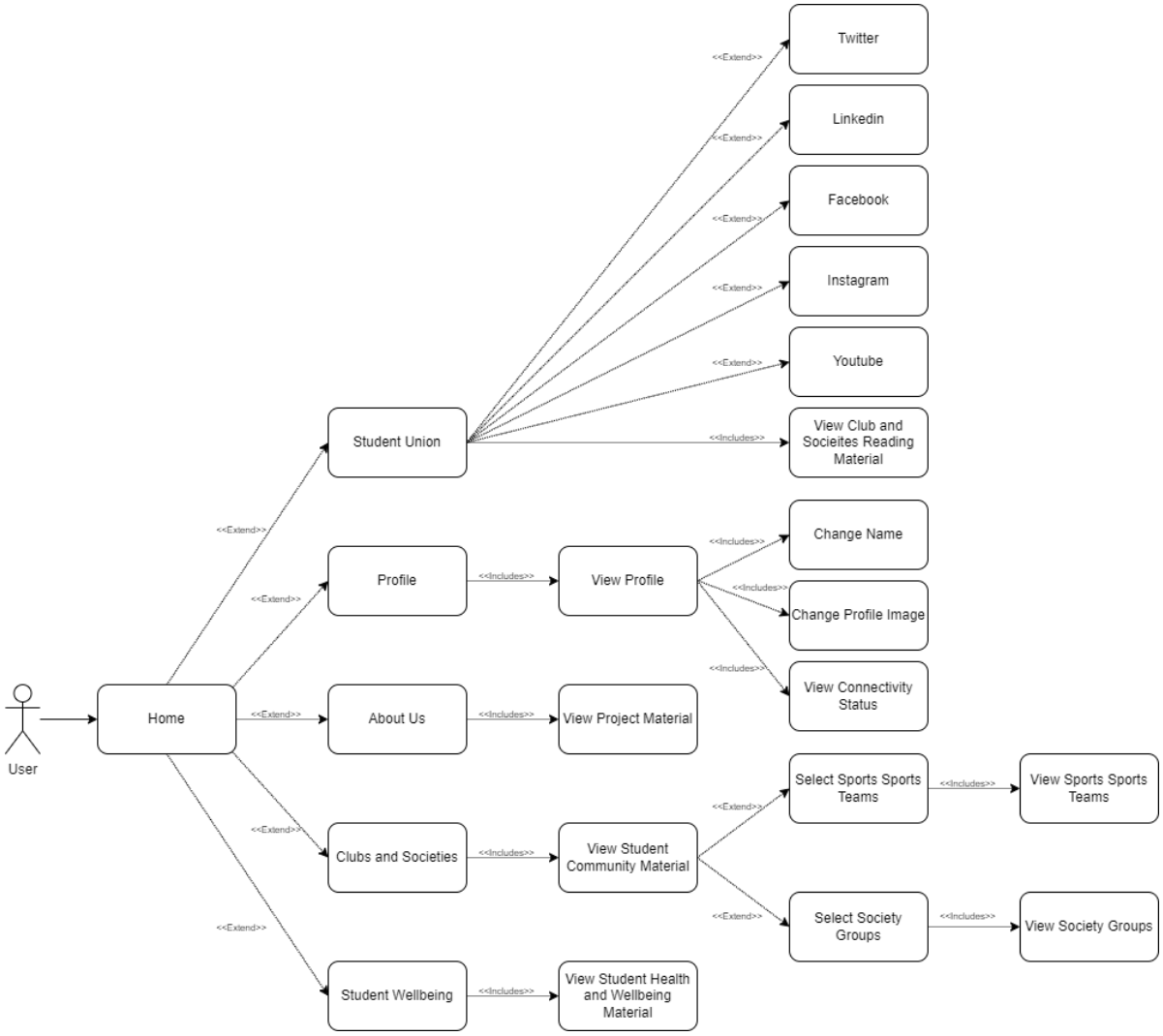


Figure 47 Use Case Diagram Home Part 1

At the Home activity, the user is able to select a total of 10 options, this includes, Discussion Board, Friends, UoW Message, Student Union, Clubs and Societies, About Us, Student Wellbeing, Profile, Settings, and Log Out. In Figure 47 Use Case Diagram Home Part 1, it illustrates a partial selection from the user which will be extended in Figure 48 Use Case Diagram Home Part 2.

In Figure 47 Use Case Diagram Home Part 1, the user can select Student Union for general information as well as an extension other social media platforms including Twitter, LinkedIn, Facebook, Instagram, and YouTube using a hyperlink image. Also, the profile activity allows the user to view and change their current profile information including name, profile image and connectivity status. Furthermore, the Clubs and Societies section of the app gives the user knowledge on the student community in place, this also is separated into two areas including the sports teams and society groups available at the University of Westminster. Moreover, the user will be able to evaluate their circumstances with the information given in the Student Wellbeing offered by the institution and third-party services. Lastly, the About Us area provides an insight to the user on the application development and planning process conducted.

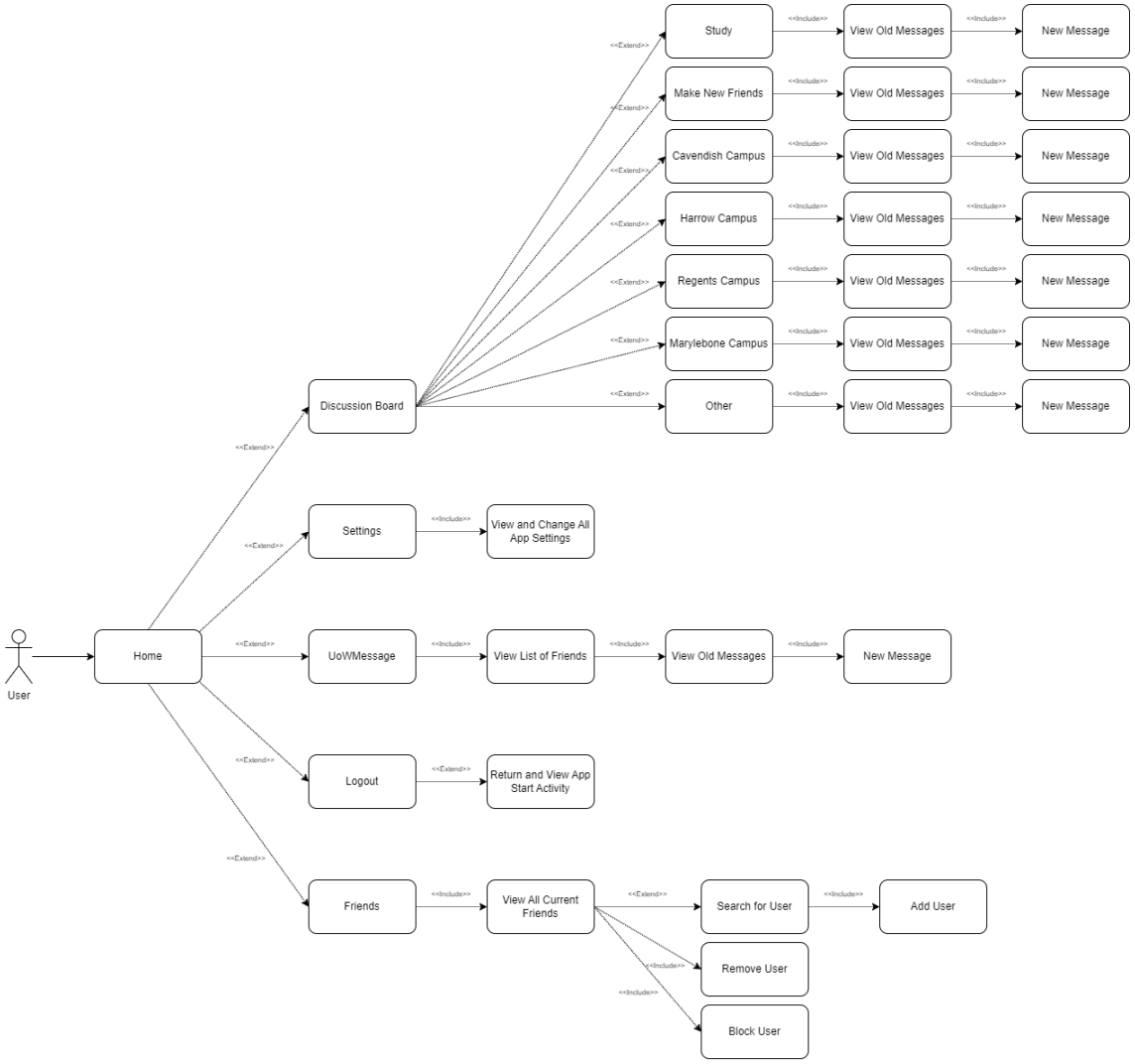


Figure 48 Use Case Diagram Home Part 2

Please note: (Figure 48 Use Case Diagram Home Part 2) is an extension to (Figure 47 Use Case Diagram Home Part 1).

In Figure 48 Use Case Diagram Home Part 2, can extend to five separate categories. Firstly, the user can select the discussion board area, where they will find seven separate categories including Study, Make new friends, Cavendish campus, Harrow campus, Regents Campus, Marylebone Campus and other. All of these options create an area for the user to view old messages and discussions and they are also able to create a new message to interact with any section of their choice. Secondly, the Settings area simply allows the user to change all app settings available to them. Thirdly, the UoW Message section provides a clarity on the list of friends added to their account, this is expanded to view their previous messages as well as creating a new message with the targeted user. Fourthly, the logout button brings the user back to the Figure 46 Use Case Diagram App Start application starting area. Lastly, the Friends area of the application gives the user control on the friends list including search for a user and add the user, remove a user, and block a user.

Use Case ID:

Table 18 Use case Application Menu

Use Case ID:	Application Menu
Description:	The user starts the application and provides a menu where the user can select Register or Login.
Primary Actor:	User
Pre-Conditions:	Application Launch
Post-Conditions:	Presents the user with an application starting menu.
Triggers:	N/A
Main Success Scenario:	1. User launches the application. 2. User is presented with the application menu.
Variations:	N/A

Table 19 Use case Register

Use Case ID:	Register
Description:	User account registration.
Primary Actor:	User
Pre-Conditions:	User has not registered with the system.
Post-Conditions:	User data is saved into the database.
Triggers:	User selects the register button.
Main Success Scenario:	1. User enters their username and password. 2. Firebase confirms their credential. 3. User is forwarded to the login area with an indicated the registration is successful.
Variations:	The user inserts invalid email or password and will not be directed to the login menu.

Table 20 Use case Login

Use Case ID:	Login
Description:	User login and authentication
Primary Actor:	User
Pre-Conditions:	User has an account within the system.
Post-Conditions:	User is directed to the home activity.
Triggers:	User selects login button.
Main Success Scenario:	1. User enters their username and password. 2. Firebase verifies the credentials. 3. User is granted access to the home activity.
Variations:	The user inserts invalid email or password and will prompt the user

	to insert the correct email and password again.

Table 21 Use case Forgot Password

Use Case ID:	Forgot Password
Description:	Password recovery request to gain access to the system.
Primary Actor:	User
Pre-Conditions:	User has an account within the system but has forgotten their password.
Post-Conditions:	User will be receiving an email to reset their password.
Triggers:	User selects forgot password submission button.
Main Success Scenario:	1. The user selects forgot password. 2. User enters registered email address. 3. Firebase sends a reset link to the email address inserted. 4. User enters a new password. 5. Firebase updates the password.
Variations:	The user enters an email unregistered.

Table 22 Use case Home

Use Case ID:	Home
Description:	The central activity in the application after login.
Primary Actor:	User
Pre-Conditions:	User is logged in.
Post-Conditions:	User is able view all the possible activities with application.
Triggers:	User is logged in.
Main Success Scenario:	1. The Home activity is displayed. 2. User interacts with the elements in the home activity.
Variations:	N/A

Table 23 Use case Student Union

Use Case ID:	Student Union
Description:	Access the student union organisation resources where all the information on how they represent the interests and welfare of all students.
Primary Actor:	User
Pre-Conditions:	User is logged in.
Post-Conditions:	User is able to view the success of the student union as well as external hyperlinks to other social

	media platforms.
Triggers:	Selects the Student Union icon button.
Main Success Scenario:	1. The Student Union information is displayed. 2. User interacts with the elements in the Student Union activity including the external social media hyperlinks.
Variations:	N/A

Table 24 Use case YouTube

Use Case ID:	YouTube
Description:	Access the student union YouTube channel.
Primary Actor:	User
Pre-Conditions:	1. The user is logged in. 2. Selects the Student Union icon button.
Post-Conditions:	User is able to view the videos uploaded to the Student Union official YouTube channel.
Triggers:	User selects the YouTube icon.
Main Success Scenario:	1. The YouTube icon is displayed. 2. The user taps the YouTube icon. 3. The app opens the YouTube hyperlink to a web browser or pre-installed YouTube app.
Variations:	N/A

Table 25 Use case Instagram

Use Case ID:	Instagram
Description:	Access the student union Instagram images and story.
Primary Actor:	User
Pre-Conditions:	1. The user is logged in. 2. Selects the Student Union icon button.
Post-Conditions:	User is able to view the images and story uploaded to the Student Union official Instagram account.
Triggers:	User selects the Instagram icon.
Main Success Scenario:	1. The Instagram icon is displayed. 2. The user taps the Instagram icon. 3. The app opens the Instagram hyperlink to a web browser or pre-installed Instagram app.
Variations:	N/A

Table 26 Use case Facebook

Use Case ID:	Facebook
Description:	Access the student union Facebook posts.
Primary Actor:	User
Pre-Conditions:	1. The user is logged in. 2. Selects the Student Union icon button.
Post-Conditions:	User is able to view the post on the Student Union official Facebook account.
Triggers:	User selects the Facebook icon.
Main Success Scenario:	1. The Facebook icon is displayed. 2. The user taps the Facebook icon. 3. The app opens the Facebook hyperlink to a web browser or pre-installed Facebook app.
Variations:	N/A

Table 27 Use case LinkedIn

Use Case ID:	LinkedIn
Description:	Access the student union LinkedIn posts.
Primary Actor:	User
Pre-Conditions:	1. The user is logged in. 2. Selects the Student Union icon button.
Post-Conditions:	User is able to view the posts on the Student Union official LinkedIn account.
Triggers:	User selects LinkedIn icon.
Main Success Scenario:	1. The LinkedIn icon is displayed. 2. The user taps the LinkedIn icon. 3. The app opens the LinkedIn hyperlink to a web browser or pre-installed LinkedIn app.
Variations:	N/A

Table 28 Use case Twitter

Use Case ID:	Twitter
Description:	Access the student union Twitter tweet / post.
Primary Actor:	User
Pre-Conditions:	1. The user is logged in. 2. Selects the Student Union icon button.
Post-Conditions:	User is able to view the posts on the Student Union official Twitter account.
Triggers:	User selects Twitter icon.
Main Success Scenario:	1. The Twitter icon is displayed. 2. The user taps the Twitter icon. 3. The app opens the Twitter hyperlink to a web browser or pre-

	installed Twitter app.
Variations:	N/A

Table 29 Use case Profile

Use Case ID:	Profile
Description:	The user can view and change their profile information.
Primary Actor:	User
Pre-Conditions:	User is logged in.
Post-Conditions:	The user is able to view and update their profile information.
Triggers:	User selects Profile icon button.
Main Success Scenario:	1. The user profile information is displayed. 2. User can update their profile information including, name, profile image and connectivity status. 3. The user save changes.
Variations:	N/A

Table 30 Use case About Us

Use Case ID:	About Us
Description:	The user can view the description of the project leader and creation of the prototype.
Primary Actor:	User
Pre-Conditions:	User is logged in.
Post-Conditions:	The user is now informed of the project leader.
Triggers:	User selects About Us icon button.
Main Success Scenario:	1. The about us information is displayed. 2. User can view information about the project leader and the creation of the prototype.
Variations:	N/A

Table 31 Use case Clubs and Societies

Use Case ID:	Clubs and Societies
Description:	The user can view a sub section of what the Student Union offers for students to engage with each other.
Primary Actor:	User
Pre-Conditions:	User is logged in.
Post-Conditions:	The user is able to view the values of Clubs and Societies as well as a sub-section of Sports Teams and Society Groups.
Triggers:	User selects Clubs and Societies icon button.
Main Success Scenario:	1. The Clubs and Societies

	information is displayed. 2. User can view the two sub-sections of Sports Teams and Society Groups. 3. The user selects Sports Teams or Society Groups button.
Variations:	N/A

Table 32 Use case Sports Teams

Use Case ID:	Sports Teams
Description:	The user can view the available Sports teams at the University of Westminster.
Primary Actor:	User
Pre-Conditions:	1. The user is logged in. 2. Selects the Clubs and Societies icon button.
Post-Conditions:	The user is now informed of the available sports teams.
Triggers:	User selects Sports team button.
Main Success Scenario:	1. The Sports Teams information is displayed. 2. User can view information on all sports teams available.
Variations:	N/A

Table 33 Use case Society Groups

Use Case ID:	Society Groups
Description:	The user can view the available Society groups at the University of Westminster
Primary Actor:	User
Pre-Conditions:	1. The user is logged in. 2. Selects the Clubs and Societies icon button.
Post-Conditions:	The user is now informed of the project leader.
Triggers:	User selects Society Groups button.
Main Success Scenario:	1. The Society Group information is displayed. 2. User can view information on all society groups available.
Variations:	N/A

Table 34 Use case Student Wellbeing

Use Case ID:	Student Wellbeing
Description:	The user can view the available student wellbeing resources offered by the university.
Primary Actor:	User
Pre-Conditions:	The user is logged in.
Post-Conditions:	The user is now informed of the

	available student wellbeing resources.
Triggers:	User selects Student Wellbeing icon button.
Main Success Scenario:	1. The Student Wellbeing information is displayed. 2. User can view information on all student wellbeing resources available.
Variations:	N/A

Table 35 Use case Discussion Board

Use Case ID:	Discussion Board
Description:	A live forum section where all students can engage with each other.
Primary Actor:	User
Pre-Conditions:	The user is logged in.
Post-Conditions:	The user can view information and select on one of the following discussion areas: 1. Study 2. Make New Friends 3. Cavendish Campus 4. Harrow Campus 5. Regents Campus 6. Marylebone Campus 7. Other
Triggers:	User selects Discussion Board icon button.
Main Success Scenario:	1. The discussion category and buttons are displayed. 2. User can
Variations:	N/A

Table 36 Use case Study

Use Case ID:	Study
Description:	An area of the application where students can engage with each other on the studying topic.
Primary Actor:	User
Pre-Conditions:	1. The user is logged in. 2. Selects the Discussion Board icon button.
Post-Conditions:	User is able to view and contribute towards a studying topic by creating a new message.
Triggers:	User selects the study button.
Main Success Scenario:	1. The previous conversation is displayed. 2. The user can view old messages involving studying. 3. The user can create a new message regarding studying.
Variations:	N/A

Table 37 Use case Make New Friends

Use Case ID:	Make New Friends
Description:	An area of the application where students can engage together and make new friends.
Primary Actor:	User
Pre-Conditions:	1. The user is logged in. 2. Selects the Discussion Board icon button.
Post-Conditions:	User is able to view and contribute towards making new friends by engaging a new message.
Triggers:	User selects the Make New Friends button.
Main Success Scenario:	1. The previous conversation is displayed. 2. The user can view old messages involving making new friends. 3. The user can create a new message regarding making new friends.
Variations:	N/A

Table 38 Use case Cavendish Campus

Use Case ID:	Cavendish Campus
Description:	An area of the application where students can engage with each other in the Cavendish Campus discussion.
Primary Actor:	User
Pre-Conditions:	1. The user is logged in. 2. Selects the Discussion Board icon button.
Post-Conditions:	User is able to view and contribute towards the Cavendish campus topic by creating a new message.
Triggers:	User selects the Cavendish Campus button.
Main Success Scenario:	1. The previous conversation is displayed. 2. The user can view old messages involving Cavendish campus. 3. The user can create a new message regarding cavendish campus.
Variations:	N/A

Table 39 Use case Harrow Campus

Use Case ID:	Harrow Campus
Description:	An area of the application where

	students can engage with each other in the Harrow Campus discussion.
Primary Actor:	User
Pre-Conditions:	1. The user is logged in. 2. Selects the Discussion Board icon button.
Post-Conditions:	User is able to view and contribute towards the Harrow Campus topic by creating a new message.
Triggers:	User selects the Harrow Campus button.
Main Success Scenario:	1. The previous conversation is displayed. 2. The user can view old messages involving Harrow Campus. 3. The user can create a new message regarding Harrow campus.
Variations:	N/A

Table 40 Use case Regents Campus

Use Case ID:	Regents Campus
Description:	An area of the application where students can engage with each other on the studying topic.
Primary Actor:	User
Pre-Conditions:	1. The user is logged in. 2. Selects the Discussion Board icon button.
Post-Conditions:	User is able to view and contribute towards the Regents Campus topic by creating a new message.
Triggers:	User selects the Regents Campus button.
Main Success Scenario:	1. The previous conversation is displayed. 2. The user can view old messages involving Regents Campus. 3. The user can create a new message regarding Regents Campus.
Variations:	N/A

Table 41 Use case Marylebone Campus

Use Case ID:	Marylebone Campus
Description:	An area of the application where students can engage with each other on the Marylebone Campus topic.
Primary Actor:	User
Pre-Conditions:	1. The user is logged in. 2. Selects the Discussion Board

	icon button.
Post-Conditions:	User is able to view and contribute towards the Marylebone Campus topic by creating a new message.
Triggers:	User selects the Marylebone Campus button.
Main Success Scenario:	1. The previous conversation is displayed. 2. The user can view old messages involving Marylebone Campus. 3. The user can create a new message regarding Marylebone Campus.
Variations:	N/A

Table 42 Use case Other

Use Case ID:	Other
Description:	An area of the application where students can engage with each other without relaying to a specific category of topics.
Primary Actor:	User
Pre-Conditions:	1. The user is logged in. 2. Selects the Discussion Board icon button.
Post-Conditions:	User is able to view and contribute towards any or off topic chat by creating a new message.
Triggers:	User selects the Other button.
Main Success Scenario:	1. The previous conversation is displayed. 2. The user can view old messages. 3. The user can create a new message.
Variations:	N/A

Table 43 Use case Settings

Use Case ID:	Settings
Description:	The user views and updates any available in app settings.
Primary Actor:	User
Pre-Conditions:	The user is logged in.
Post-Conditions:	User can adjust and save in app settings.
Triggers:	User selects the Settings icon button.
Main Success Scenario:	1. The previous conversation is displayed. 2. The user can adjust in app settings. 3. The user can save changes.
Variations:	N/A

Table 44 Use case UoW Message

Use Case ID:	UoW Message
Description:	The user sends and receives messages to / from other users.
Primary Actor:	User
Pre-Conditions:	The user is logged in.
Post-Conditions:	User is able to send, and view received messages from the contact.
Triggers:	User selects the UoW Message icon button.
Main Success Scenario:	1. The user can see previous messages. 2. The user can compose new messages. 3. The message is sent and received by the targeted user.
Variations:	N/A

Table 45 Use case Friends

Use Case ID:	Friends
Description:	The user is able to view, search, block and remove friends
Primary Actor:	User
Pre-Conditions:	The user is logged in.
Post-Conditions:	User is able to view and contribute towards the studying topic by creating a new message.
Triggers:	User selects the Friends icon button.
Main Success Scenario:	1. The friends list is displayed. 2. The user search for a new friend. 3. The user can block or remove an existing user in their friends list.
Variations:	N/A

Table 46 Use case Search for User

Use Case ID:	Search for User
Description:	The user can search for another user and have an option to add and send a friend request.
Primary Actor:	User
Pre-Conditions:	1. The user is logged in. 2. Selects the Friends icon button.
Post-Conditions:	User is able to view and contribute towards the studying topic by creating a new message.
Triggers:	User selects the study button.
Main Success Scenario:	1. The previous conversation is displayed.

	2. The user can view old messages involving studying. 3. The user can create a new message regarding studying.
Variations:	N/A

Table 47 Use case Logout

Use Case ID:	Logout
Description:	The user is log out of the application and return to the starting app activity.
Primary Actor:	User
Pre-Conditions:	The user is logged in.
Post-Conditions:	User session is terminated, and the user no longer has access to the account until they log back in.
Triggers:	User selects the logout button.
Main Success Scenario:	1. User is prompted to confirm the logout action. 2. User confirms the logout prompt. 3. The user is logged out and returns to the starting app activity.
Variations:	The user cancels the logout prompt.

5.6 Risk Assessment

The risk assessment consists of the Risk, Impact, Solution and Potential Risk. The potential risk will be represented as follows:

(L) - Low

(M) - Medium

(H) – High

Table 48 Risk Assessment Table

Risk	Impact	Solution	Potential Risk
The app does not function as intended (errors and bugs).	The user could be slow, cannot register or login and possibly crash and unable to open the application.	Test the application regularly to ensure there are no existing errors in the application.	**(H)**
The app disconnects from the database.	The user will not be able to login or signup successfully or unable to send and receive messages.	Ensure the Firebase SDK is applied within the app project and the connection to the database is tested regularly.	**(H)**
The user interface is not supported for all android devices.	The user may find buttons, images, text positioned correctly and hard to read or buttons behind another object in the app.	Spend more time to ensure all buttons, images and text are constraint correctly and test the application for any incorrect visual alignments.	**(H)**
The user is unable to add, remove and block users.	The user will be unable to use direct messaging.	Ensure the database is connected correctly and the functionality is in place by using multiple account testing between each other.	**(H)**
The user cannot send messages in the discussion board / forum section of the app.	The user will not be able to message in the forum.	Ensure the database is connected correctly and have the correct functions in place.	**(H)**
The admin cannot delete messages in the discussion board / forum	The admin has no control over the application.	Ensure the database is connected and the admin has separate and administrative	**(H)**

section of the app.		functionalities to conventional users to ensure the application covers ethical concerns.	
The user profile information is unlisted.	The user cannot see their profile information.	Ensure the appropriate methods are in place and the database is connected to the user profile activity.	**(M)**
Hyperlink buttons and image buttons to external websites does not function.	The user presses any hyperlink button with no functionality.	Ensure the URL is directed to the button.	**(M)**
Information for Student Union, Clubs and Societies and Student Wellbeing is insufficient.	The user will not gain sufficient knowledge about the programmes available in Student Union, Clubs and Societies and Student Wellbeing.	Research and apply sufficient information in the activity to ensure the information is well explained.	**(L)**
The application is unappealing to users.	The user will not have an enjoyable experience.	Bring attention to detailing the application with essentials and desirables along with consistency of an engaging visual presentation.	**(L)**

6. Tools and Implementation

6.1 Tools

In this section, all tools that were used throughout the mobile development will be documented. This includes the programming language used, implementation aspects of design as well as how Google Firebase was applied into the project.

6.1.1 Android Studio

Android Mobile applications can be developed using many other programming software including, IntelliJ, Eclipse, VSCode, React Native and many more. However, Android Studio still remains as the best and primary programming software to implement Android Apps. Android Studio is completely and officially supported by Google, the use of Database and other tools for this app requires many Google services, therefore it was easily the best choice to conduct prototypes and the completed project.

6.1.2 Kotlin

Once the project was finalised in developing an Android Mobile Application, there was a choice between Java or Kotlin for programming language. Due to the substantial benefits in Kotlin, it was clear to acquire all the resources and technological skills for this project. Kotlin provides functional programming support, which allows developers to create readable, concise, and maintainable code. Furthermore, Kotlin is integrated with Java interoperability, therefore Java libraries and frameworks can be used in Kotlin. Finally, Kotlin Syntax are created with further advancements in readability, this gives developers a well-defined solution to any errors at hand.

6.1.3 Themes

Android Studio supports two types of themes for light and dark mode, AppCompact Themes and Material Design Themes. The differences between the two are, AppCompact is backwards compatible and supports older versions, whereas Material Design is a newer technology which was built on-top of AppCompact and it also includes pre-defined light and dark mode variants upon building on colour resources. In this project, Material Design was used, due to the adaptability and simplistic structure to create a light and dark mode for the project.

6.1.4 XML

XML Extensible Markup Language is used to store and transport data. It was designed for readability for both human and machine. XML is extensively used for layouts for each page within the application, drawable shapes and icons, manifest file data controls on permissions, package name and version, and values for colours and themes. All of these XML processes was used during development of the application.

6.1.5 Circle Image View and Glide

Circle Image View was used heavily to allow users to view profile images, this implementation is not a tool which is pre-installed into the Android Studio platform, therefore the use of “de.hdodenhof:circleimageview.3.1.0” was synced as a library dependency. Whilst there are predefined images, the user can upload the image into the database. In order for the circle image view to be read in a simplistic and efficient manner from the database, the use of Glide was implemented. Glide is an image loading and caching library which provides support for displaying images into a circular view.

6.1.6 Firebase Software Development Kit

Firebase Software Development Kit SDK provides additional libraries which the developer can select and use within Android Studio. This includes Firebase Authentication, Realtime Database and Storage which were used for this project. In Figure 49 Firebase SDK, it illustrates all of the Google Firebase libraries integrated with Android Studio. Overall, Firebase SDK provides a range of powerful tools to assist the development of the project with factors of scalability in the future and simplifies the development process to build a successful android mobile application.

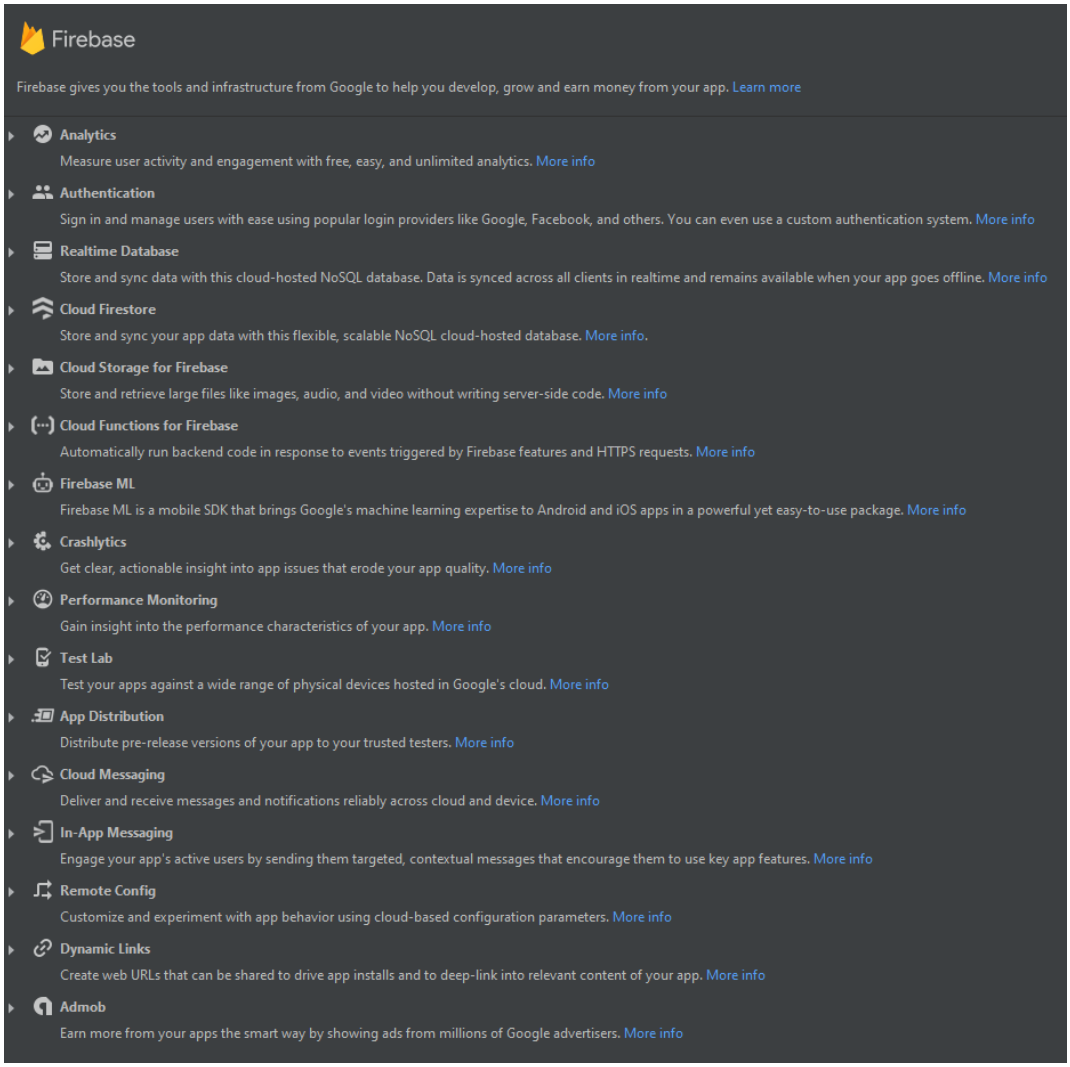


Figure 49 Firebase SDK

6.1.7 Firebase Authentication

Firebase Authentication provides native, additional providers and custom providers for user authentication within the mobile application. In this project, only email and password are the authentication process required for the user to login to their account. In Figure 50 Firebase Authentication Sign-In Method, it illustrates the possibilities of additional providers such as Google, Facebook, Twitter, and many more additional providers, however only Email / Password is the selected choice for this project.

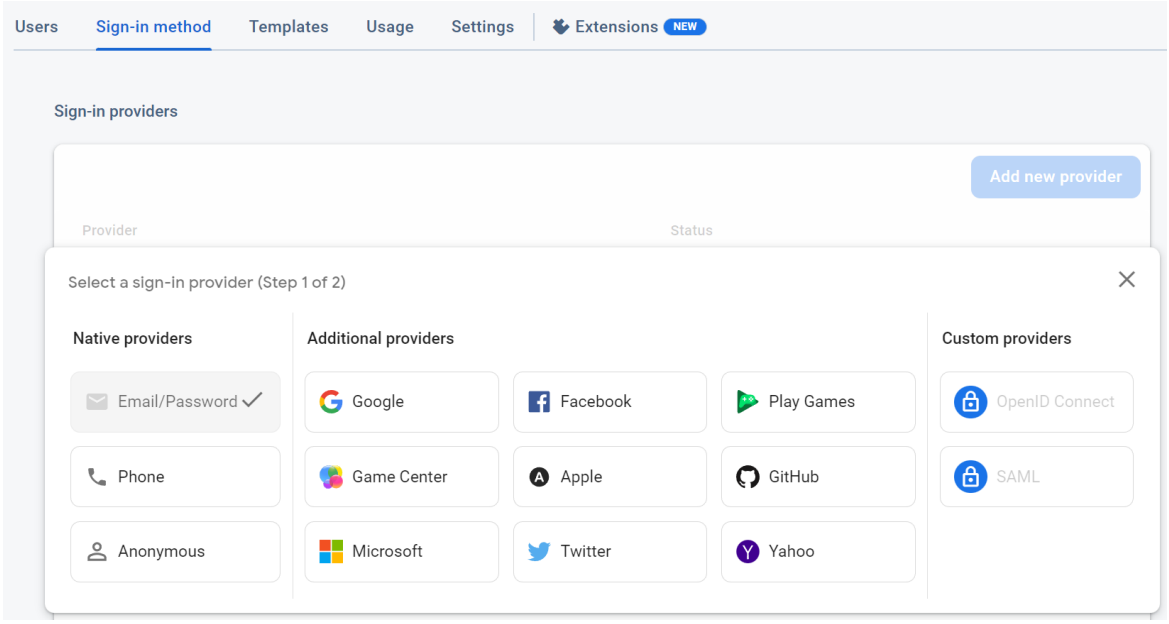


Figure 50 Firebase Authentication Sign-In Method

When a user signs up to UoWsa, their email address is listed along with the date they created the account as well as the last time they've signed into the account. The use of User ID UID is to allow simple cross reference to identify each individual user in the Firebase Realtime Database. This is illustrated in Figure 51 Firebase Authentication Users.

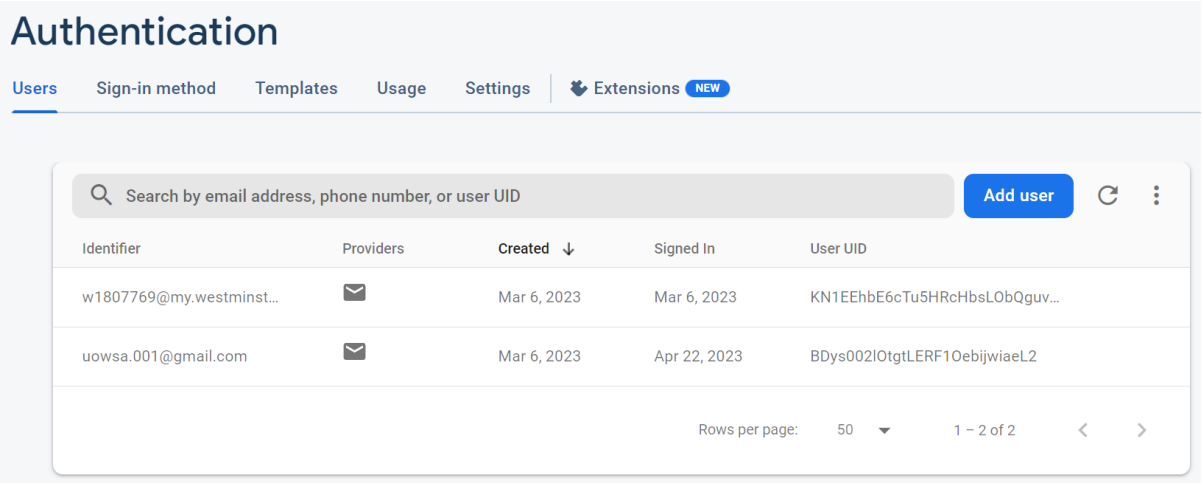


Figure 51 Firebase Authentication Users

In Figure 52 Firebase Authentication Password Recovery Template it illustrates the model used when the user enters the password recovery process. When the user inserts their email address, this template will be used and sent to the user's email address for password reset.

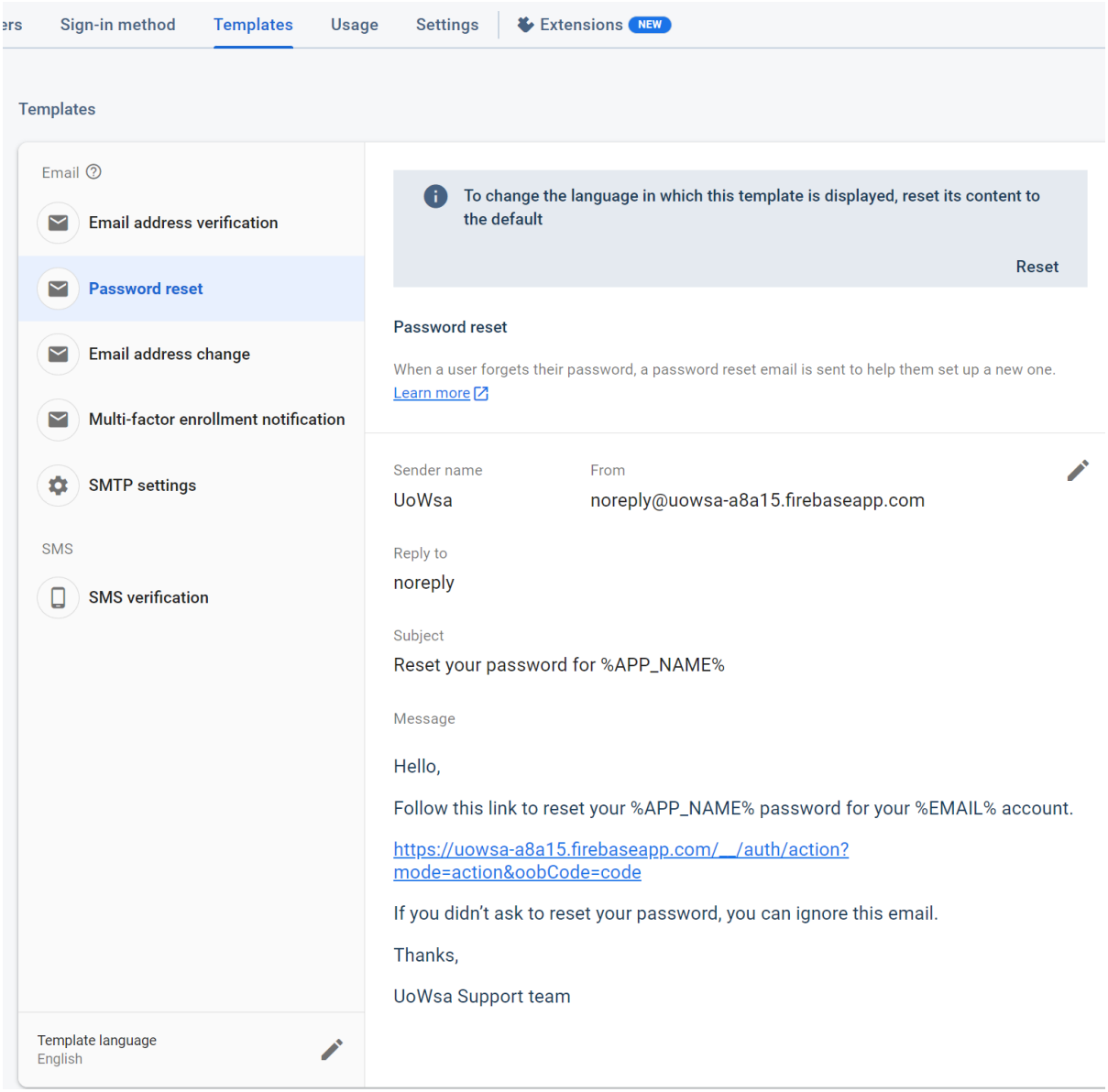


Figure 52 Firebase Authentication Password Recovery Template

6.1.8 Firebase Realtime Database

In Figure 53 Firebase Realtime Database it illustrates all of the components within the database as well as the users tab expanded. Depending on how each user interacts within the rules set in the developer's code. This database is a NoSQL database.

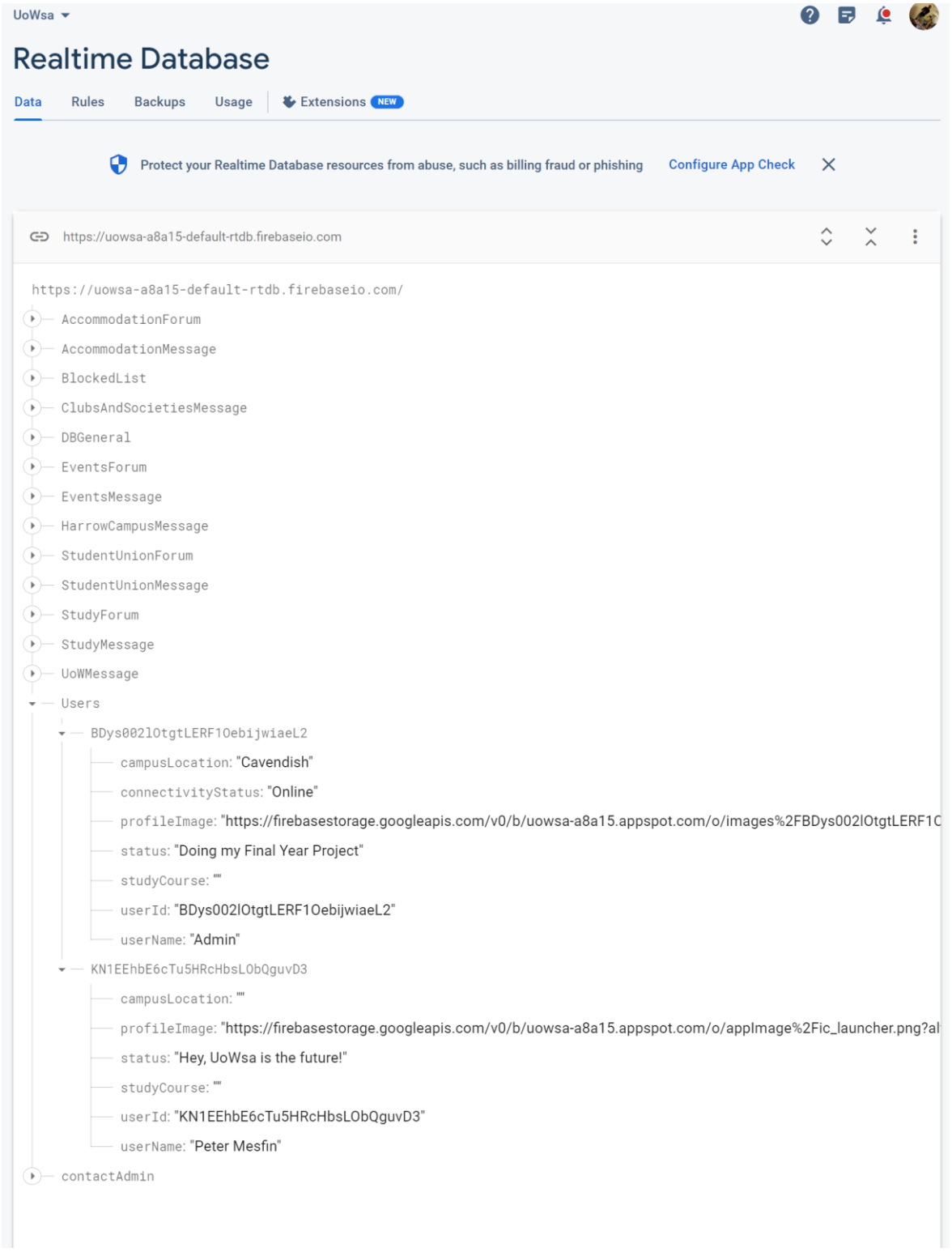


Figure 53 Firebase Realtime Database

6.1.9 Firebase Storage

The Firebase Storage is predominantly used for storing file formats. In this project, profile images are used which then has to be stored and read. In Figure 53 Firebase Realtime database in the user's section indicates a key "profileImage" and value with a specific URL link. For example, in the key "profileImage" includes the value "https://firebasestorage.googleapis.com/v0/b/uowsa-a8a15.appspot.com/o/images%2FBDys002lOtgtLERF1OebijwiaeL2?alt=media&token=05e8f6ba-f2ca-4dd2-91c0-4cee78750cab". In Figure 54 Firebase Storage, it illustrates the name of each profile image with the users UID from the authentication. The unique identifier can then refer each profile image to its unique URL with the users imbedded UID. The URL link example also reveals the users UID "BDys002lOtgtLERF1OebijwiaeL2".

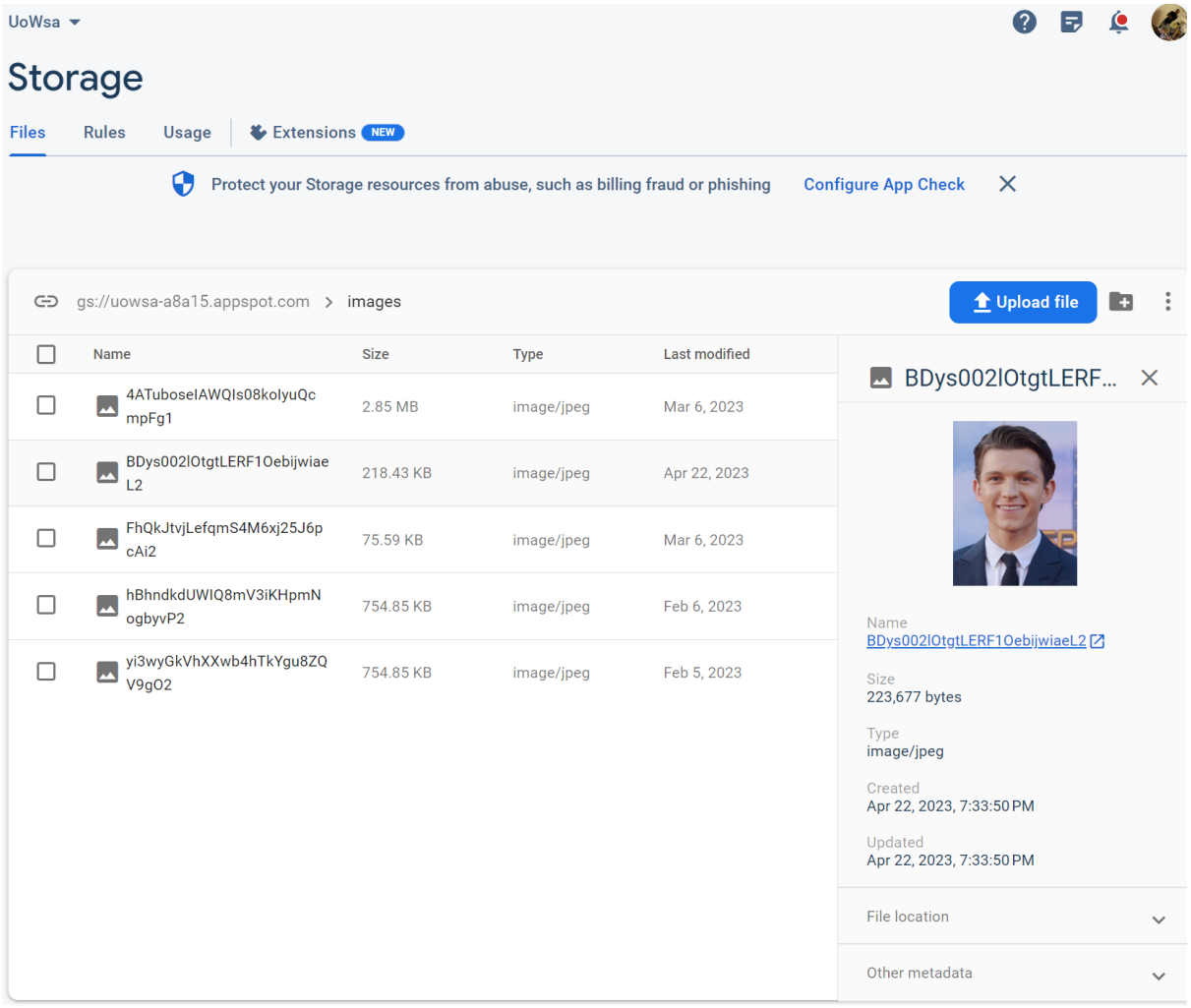


Figure 54 Firebase Storage

6.1.10 Firebase and Google Analytics

There are several types of analytics which are available with Firebase and Google API and Services. Firebase allows the developer to observe and make necessary adjustments to the UoWsa service based on the usage, in Figure 55 Firebase Authentication Usage it demonstrates how many active users on a daily and monthly basis.

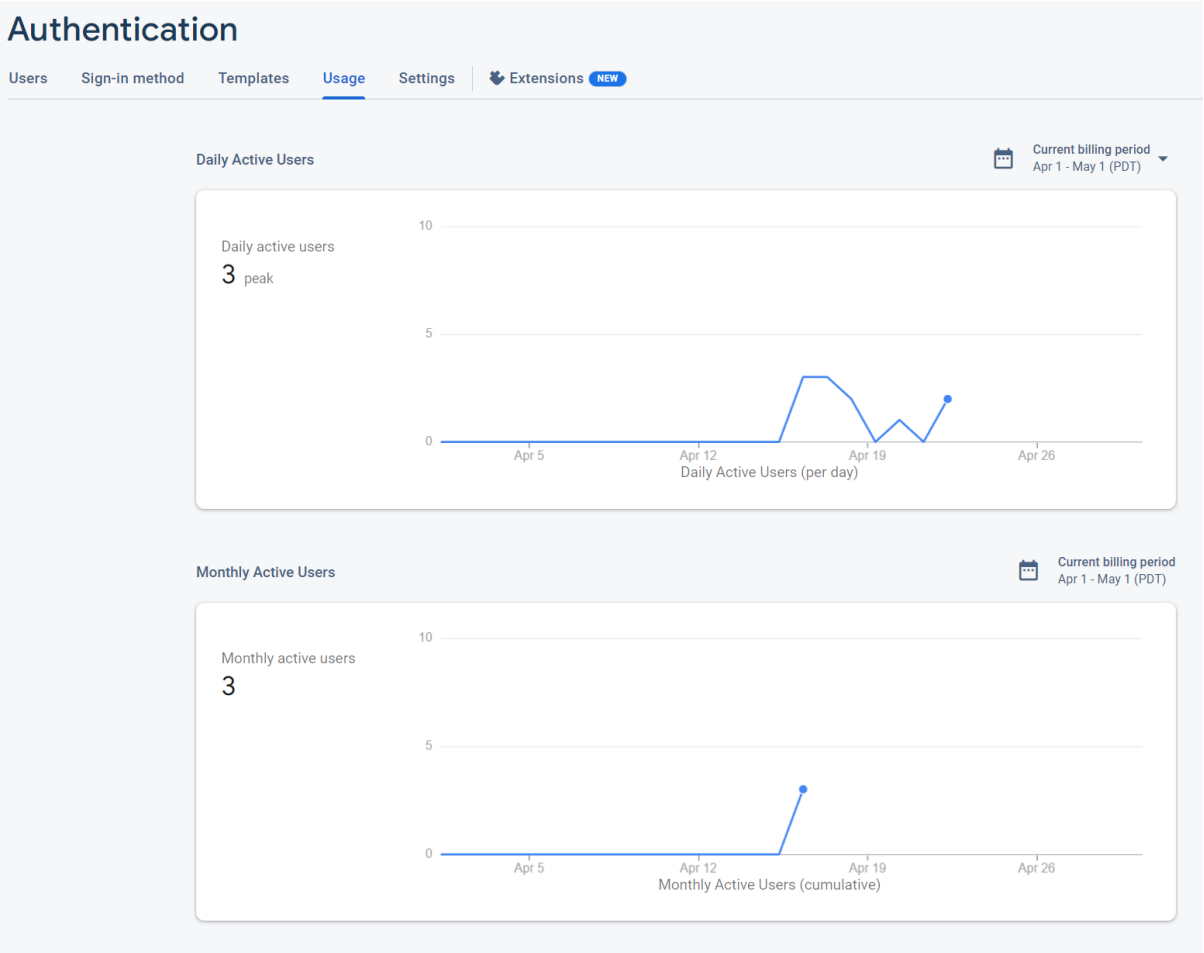


Figure 55 Firebase Authentication Usage

In Figure 56 Firebase Realtime Database Usage displays the number of real time connections to the database, how much data is stored on the database and the number of downloads made from the database including connection protocol and SSL encryption overhead.

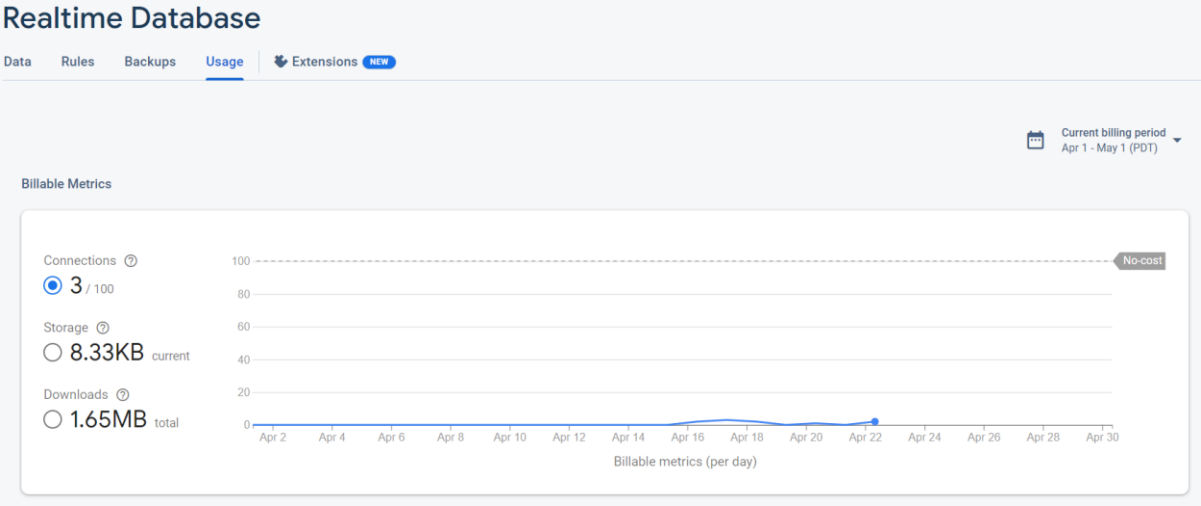


Figure 56 Firebase Realtime Database Usage

Figure 57 Firebase Storage Usage provides data on the bytes stored, which is total of how much storage space was used, the object count, the number of files in existence, bandwidth sent, the number of bytes sent from the client and requests, the total of requests made by the client.

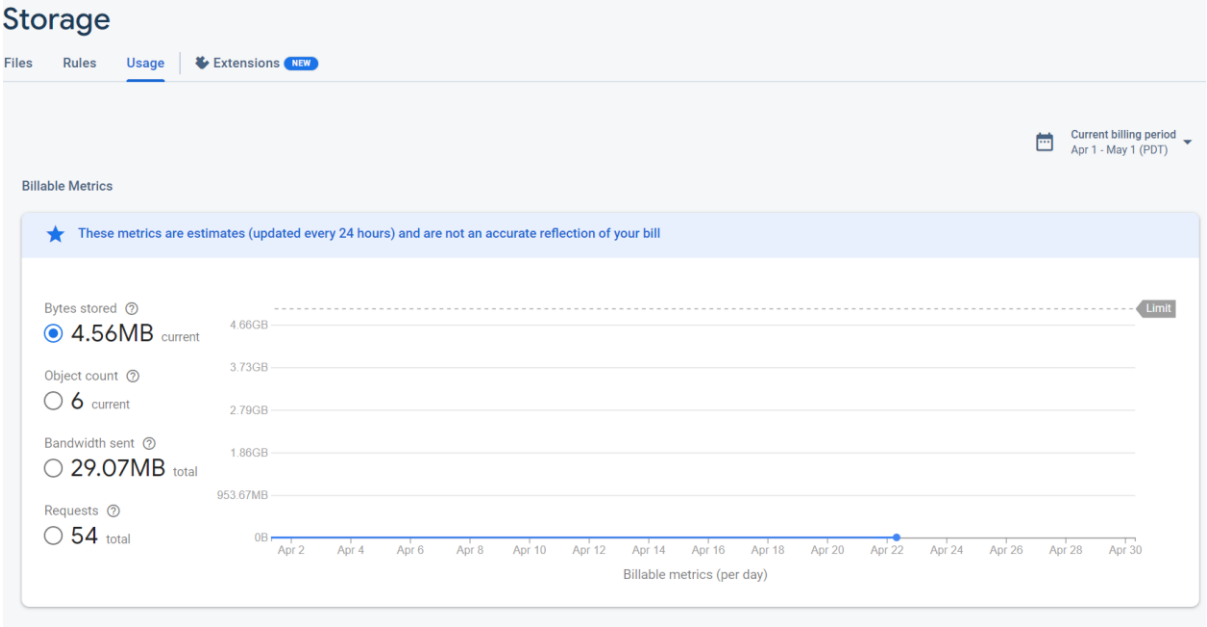


Figure 57 Firebase Storage Usage

Google API and services also offers analytics in the mobile application developed using Firebase illustrated in Figure 58 Google API and Services. For example, the developer can track the number of requests for a specific area of the project such as,

Cloud Storage for Firebase API. Figure 59 Google Services and Analytics covers information including, requests errors and latency. This dataset can provide a significant detail for overall quality of life updates for the application in the future.

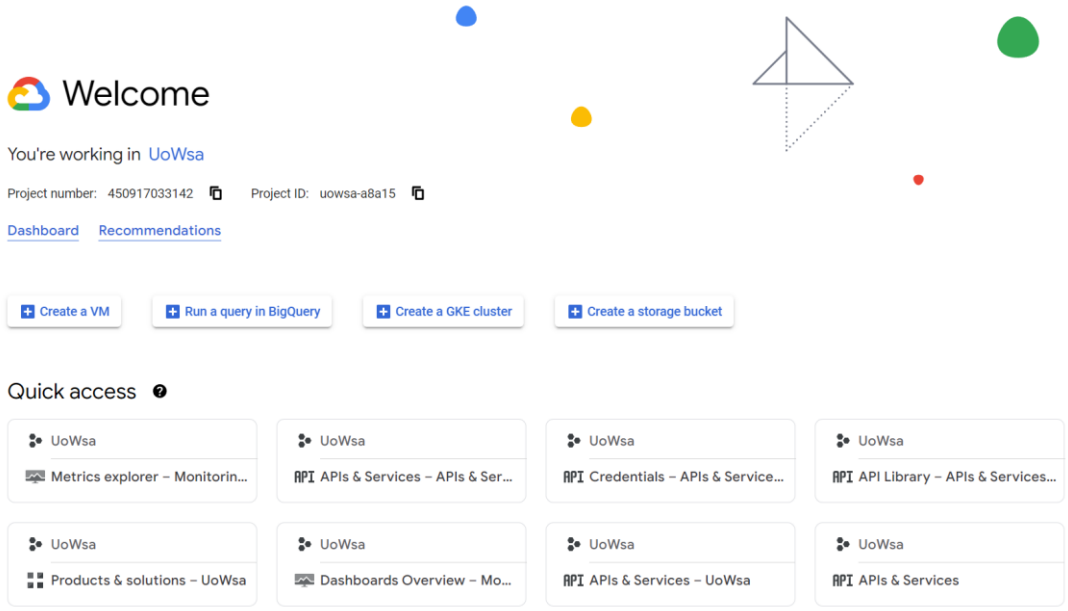


Figure 58 Google API and Services

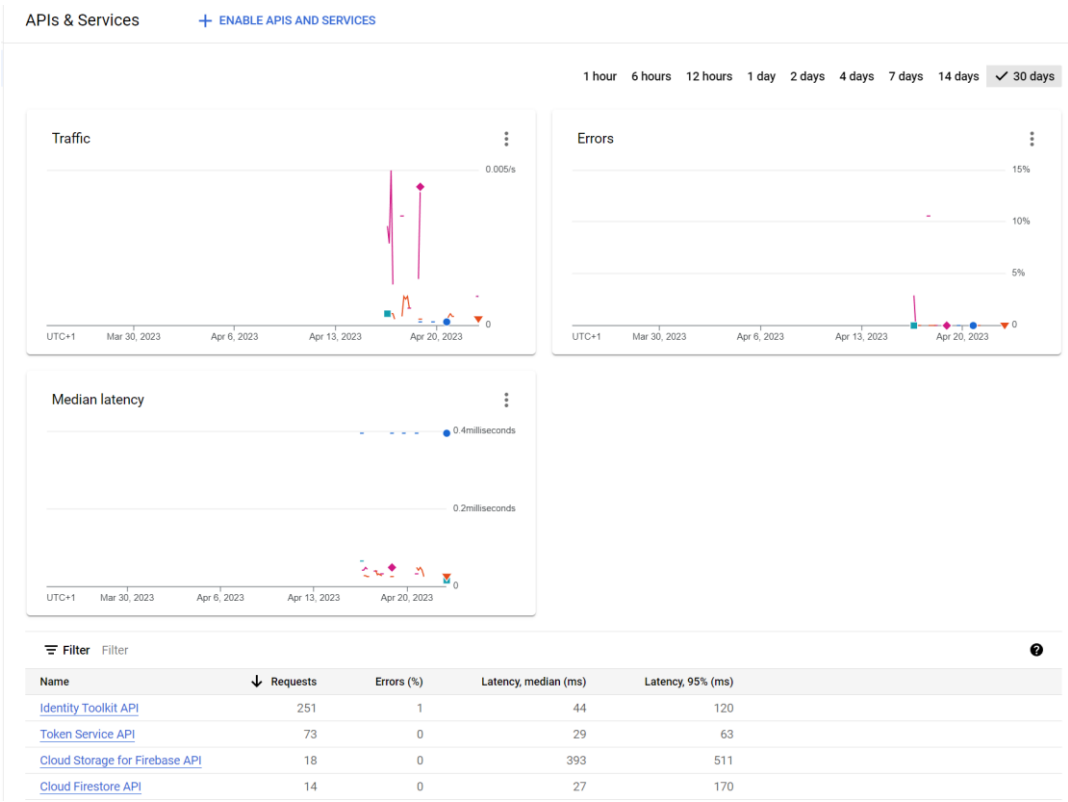


Figure 59 Google Services and Analytics

6.2 Skills

This section covers the technical skills which have been learnt through the University of Westminster Computer Science module.

6.2.1 Previous Skills

- Android Studio and Kotlin

Android Studio was introduced in the Level 5 Mobile Application Development module where I learnt the fundamentals of creating activities and using XML to visualise the design of the application. Kotlin is also a programming language I found similarities in Java where I spent my time in learning in Level 4 Software development II and Level 5 Object-Oriented Programming classes.

- Modelling Diagrams

I have modelled diagrams including concentric diagram, use-case diagram, class diagram, flowchart, and Entity Relationship Diagram. The Level 5 Client-Server Architectures, Level 5 Software Development Group Project as well as this project has given me an extensive understanding on the importance of preparation in a project.

- User Experience UX and Ethical Issues

The significance of UX designs and ethical issues came into my academic knowledge during the Level 5 Human Interaction and User Experience. This class provided psychological aspects when designing a front-end interface for the mass public, therefore I used the same approach of diversity and unbiased implementation to create this project.

6.2.2 Newly Acquired Skills

- Clip Studio Paint

  Self-created logos and other aspects of the mobile application including images and logos were created using the Clip Studio Paint software. At first, I found it extremely difficult to place images and logos inside the activity as images and logos could vary with pixel densities, therefore I had to learn

about how to resize and export images appropriately for this project (*Clip Studio Tips, n.d.*).

- Android Studio Drawable

Drawable allows me to inflate an image or shape created with XML. This is a difficult learning curve, as I found it difficult to create shapes using XML. After some research (*Grec, 2012*), I developed my skills in creating more shapes to for my throwaway prototype and prototype.

- Physical Device Testing

I ran into issues testing the functionality as I was developing the product with an emulator implemented within Android Studio, which I learnt in Level 5 Mobile Application Development. I had to accommodate by finding alternatives using an old Android mobile phone after brief research on process of connecting a physical device (*Android Developers, n.d.*).

- Firebase Authentication

The use of this authentication method provides an easy-to-use SDK to authenticate users for the application. This includes a traditional email and password signup, phone number or connecting your accounts with search engine and social media platforms such as, Google, Twitter, Facebook, Microsoft among many other (*Firebase, n.d.*). Furthermore, the advantages of Firebase Authentication for developers are the simplistic design including the choices of security implementation. I have tested some of these methods in throwaway prototypes. It has been assured the data analytics collected from the Firebase console falls under jurisdiction of the GDPR as long as the users are EU residents (*Bass, 2022*).

- Firebase Realtime Database

The NoSQL structure database has not been taught during my time so far at the university and this was a great opportunity to test and develop using a database structure which I am not familiar with. The great functional features such as real-time synchronisation, scalability and security are the main reasons why it was a no brainer to learn for this project.

- Firebase Storage

Firebase Storage serves to store user-related content in this project. The speed and performance with low latency along with its compatibility with other existing libraries such as, authentication and database give Firebase Storage a simplistic integration across all of the Firebase services.

- Colours and Themes XML

Colours and Themes XML are the colours used for both light and dark mode. During the Level 5 Mobile Application Development we were taught how to hardcode colours to specific buttons or text, however the use of an XML file to store colours and call for the specific colour code was not taught. Therefore, I became more knowledgeable on the use of colours for different aspects of the application as well as differentiate the colours used for light mode and dark mode.

- Figma

I previously learnt about Google sites from the Level 6 Informational Driven Enterprise Entrepreneurship and Axure 10 from the Level 5 Human Interaction and User Experience class for Mock-ups and wireframes. However, I have learnt using Figma for swift and precise colour coding and shapes to design the pages on the mobile application. Figma was a simplistic learning curve I've learnt from reading online and YouTube tutorials, it provides clear indication for implementation along with the design components used.

- Circle Image View

Android Studio provides basic structures on how images are presented using drawable resources, however shaping the view of the image is not integrated on the platform. Using third party libraries such as "de.hodenhod:circleimageview:3.1.0" into the project Gradle file provided knowledge on how to use external libraries for the project.

- Android Manifest

Android Manifest is an essential file integrated on Android Operating System. I have learnt briefly of its capabilities in the Level 5 Mobile Application Development module, however the use of configurations such as permissions to the internet and other implementations such as orientation and fixing size of display became an essential of how content is presented to the user in the project.

6.3 Implementation

The implementation is the code and outcomes of how the final design are implemented and illustrated. The Dark mode is demonstrated with a Google Pixel 6 Pro emulator and the Light mode is demonstrated with a Google Pixel 4 XL emulator.

6.3.1 Implementation - Start up and Authentication

The Implementation Start up and Authentication illustrates all of the pages where the user starts the application and how the user creates or login to their account. The implementation of Terms of Service as well as Forgot my password will also be included in this section. Figure 60 App icon demonstrates if the user selects the UoWsa app icon from their mobile phones the app will begin and direct the user to Figure 61 UoWsa App Start Up.

Figure 60 App Icon

Figure 61 UoWsa App Start-up

```
class MainActivity : AppCompatActivity() {

    private lateinit var auth: FirebaseAuth;

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        setContentView(R.layout.activity_main)

//          ...
//          Initialize Firebase Auth
        auth = Firebase.auth

        val login = findViewById<Button>(R.id.loginBtn)
        val signUp = findViewById<Button>(R.id.signUpBtn)
        val tos = findViewById<Button>(R.id.toS)

        login.setOnClickListener { it: View!
            val intentLogin = Intent( packageContext: this, LoginActivity::class.java)
            startActivity(intentLogin)
        }

        signUp.setOnClickListener { it: View!
            val intentSignUp = Intent( packageContext: this, SignUpActivity::class.java)
            startActivity(intentSignUp)
        }

        tos.setOnClickListener { it: View!
            val intentSignUp = Intent( packageContext: this, TermsOfServiceActivity::class.java)
            startActivity(intentSignUp)
        }
    }
}
```

Figure 62 UoWsa App Start-up code

The user will be presented with a page with three separate buttons, “Login”, “Sign Up” and “Terms of Service” illustrated in Figure 61 UoWsa App Start-up. In Figure 62 UoWsa App Start-up code it demonstrates how each button is linked to the XML layout. The .setOnClickListener function will operate once the user selects the button on the page, and the Intent function inside the .setOnClickListener Opens a new page targeted to a specific page indicated for either LoginActivity, SignUpActivity or TermsOfServiceActivity.

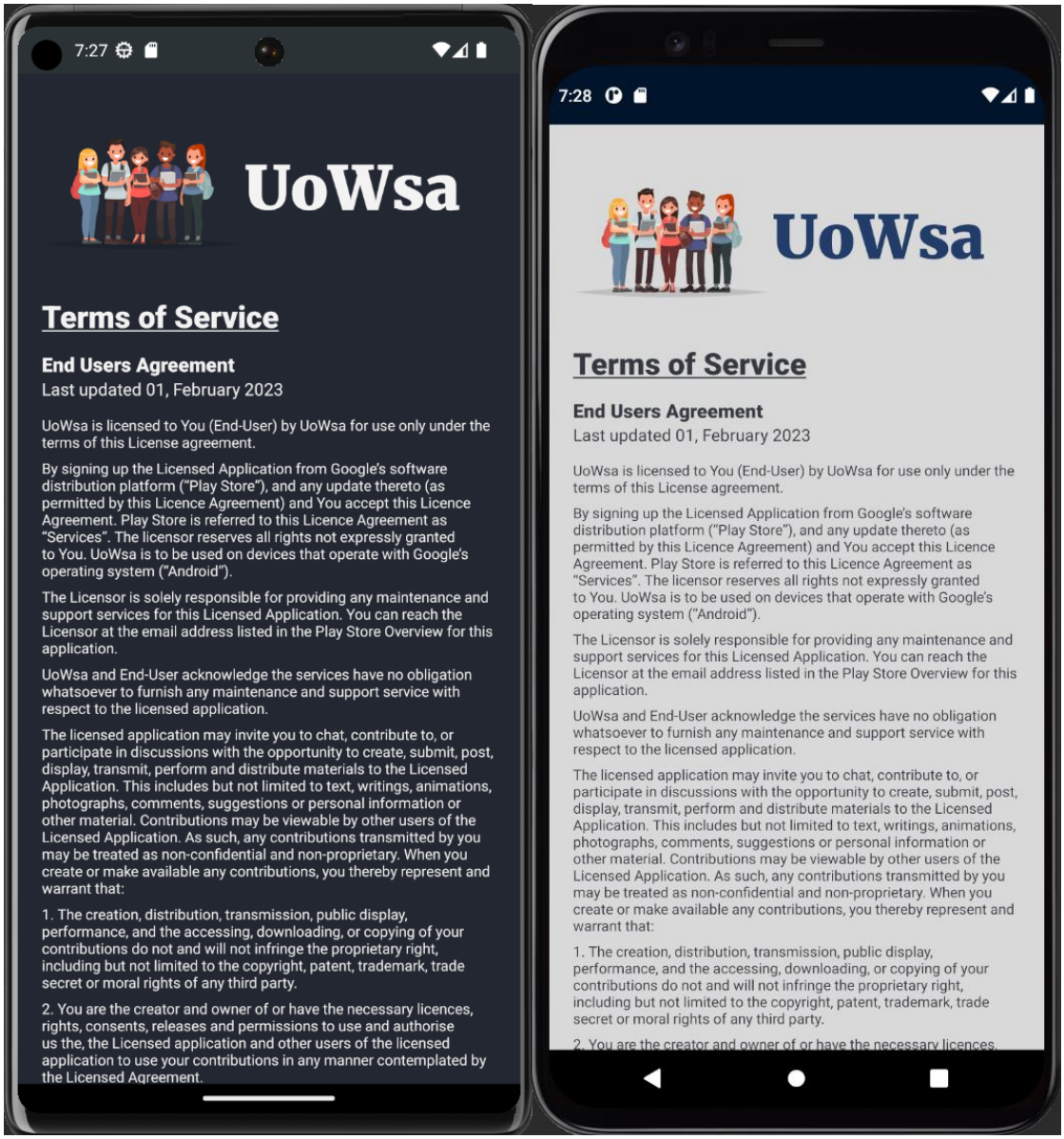


Figure 63 UoWsa Terms of Service

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

    <ScrollView
        android:layout_width="match_parent"
        android:layout_height="match_parent"
        android:fillViewport="true">

        <androidx.constraintlayout.widget.ConstraintLayout
            android:layout_width="match_parent"
            android:layout_height="wrap_content"
            android:paddingBottom="20dp">

            <ImageView
                android:id="@+id/tosImage"
                android:layout_width="170dp"
                android:layout_height="150dp"
                android:layout_marginStart="16dp"
                android:layout_marginTop="16dp"
                android:src="@drawable/ic_uowsalogo"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toTopOf="parent" />

            <TextView
                android:id="@+id/UoWsa"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginTop="60dp"
                android:fontFamily="@font/merriweather_black"
                android:text="UoWsa"
                android:textColor="?colorAccent"
                android:textSize="50sp"
                app:layout_constraintStart_toEndOf="@+id/tosImage"
                app:layout_constraintTop_toTopOf="parent" />

            <TextView
                android:id="@+id/textView"
                android:layout_width="wrap_content"
                android:layout_height="wrap_content"
                android:layout_marginStart="20dp"
                android:layout_marginTop="16dp"
                android:fontFamily="sans-serif-black"
                android:text="@string/tos"
                android:textColor="?colorPrimary"
                android:textSize="25sp"
                app:layout_constraintStart_toStartOf="parent"
                app:layout_constraintTop_toBottomOf="@+id/tosImage" />

            <TextView
                android:id="@+id/textView15"
```

Figure 64 Terms of Service XML layout

```
<resources>
    <string name="app_name">UoWsa</string>
    <string name="login">Login</string>
    <string name="logIn">Log In</string>
    <string name="signup">Sign Up</string>
    <string name="signIn">Sign In</string>
    <string name="Email">Email</string>
    <string name="Password">Password</string>
    <string name="conPassword">Confirm Password</string>
    <string name="alreadyRegisteredLogin">Already registered, Login</string>
    <string name="forgotPassword">Forgot My Password</string>
    <string name="dontHaveAnAccount">Don\'t have an account ?</string>
    <string name="alreadyHaveAnAccount">Already have an account ?</string>
    <string name="passwordReset">Password Reset</string>
    <string name="reset">Reset</string>
    <string name="studentWellbeing">STUDENT WELLBEING</string>
    <string name="discussionBoard">DISCUSSION BOARD</string>
    <string name="friends">FRIENDS</string>
    <string name="uowMessage">UoW MESSAGE</string>
    <string name="studentUnion">STUDENT UNION</string>
    <string name="clubsAndSocieties">Clubs and Societies</string>
    <string name="aboutUS">ABOUT US</string>
    <string name="settings">SETTINGS</string>
    <string name="profile">Profile</string>
    <string name="home">Home</string>
    <string name="uowsaProject">The UoWsa Project</string>
    <string name="username">Username</string>
    <string name="innerLayer">Inner Layer: University of Westminster Social Application (UoWsa).</string>
    <string name="layer1">Layer 1: The direct stakeholders who are involved with the creation and development of the project
    <string name="layer1p">The developer Christopher Wong, supervisor Markos Mentzelopoulos and selected students from the U
    <string name="layer2">Layer 2: The continuous stakeholders who maintains the application when the project is complete.</
    <string name="layer2p">Once the application is completed, the administrators take control of the in-app affairs as well
    <string name="layer3">Layer 3: External stakeholders</string>
    <string name="layer3p">External stakeholders are individuals or groups outside this project who has interests or concerr
```

Figure 65 Strings.xml

In Figure 63 Terms of Service illustrates the rules set by UoWsa for users using the application. As every layout is hardcoded the Kotlin file does not have any code for the page to do anything, however the use of “Strings.xml” is applied for the listed text views to store all the string data where the user can view the hardcoded text illustrated in Figure 65 Strings.xml.

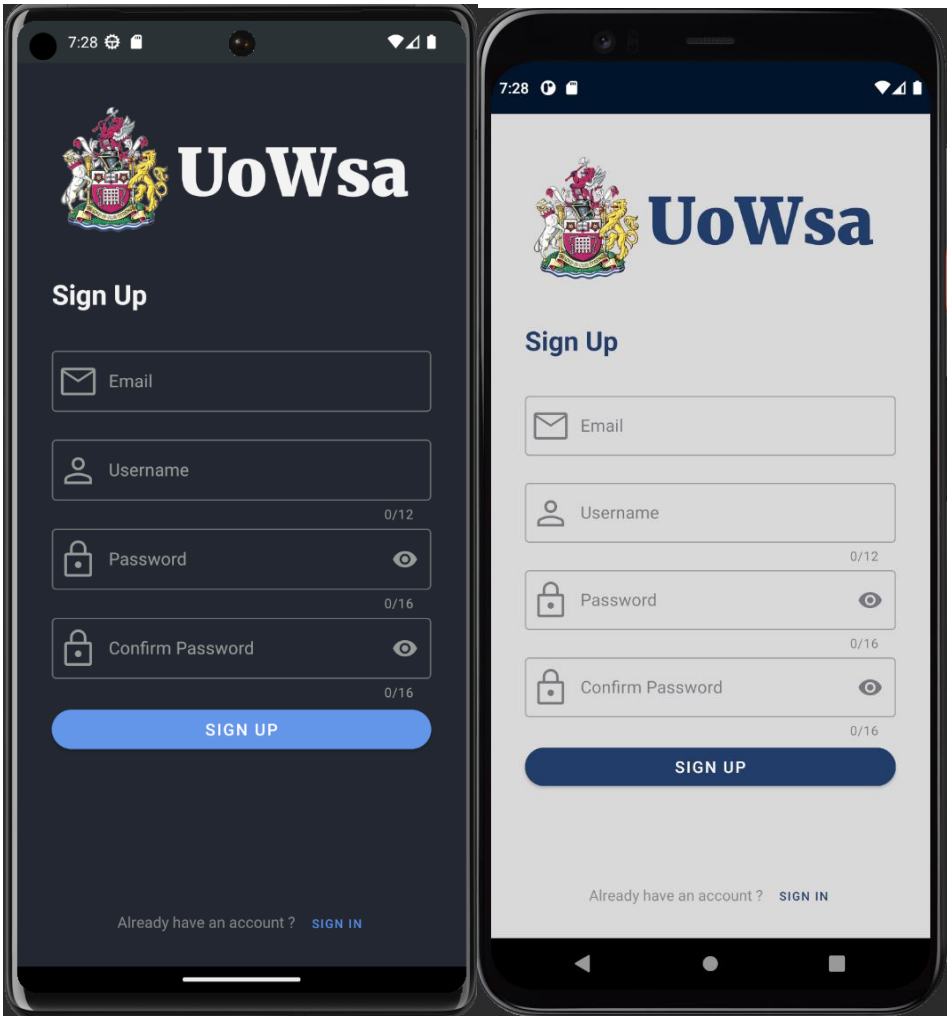


Figure 66 Sign Up

```
signUpBtn.setOnClickListener { it: View!
    if (email.text.toString().trim().isEmpty()) {
        Toast.makeText( context: this@SignUpActivity, text: "Please insert an email", Toast.LENGTH_SHORT)
            .show()
    }
    //email validation unnecessary, as the input type is set to email in the xml layout file.
    val emailTxt = email.text.toString()
    val passwordTxt = password.text.toString()
    val usernameTxt = username.text.toString()
    val conPasswordTxt = conPassword.text.toString()
    val usernameLowerCase = username.text.toString().lowercase()

    if (email.text.toString().trim().isEmpty()) {
        Toast.makeText( context: this@SignUpActivity, text: "Please insert an email.", Toast.LENGTH_SHORT).show()
    }else if(!Patterns.EMAIL_ADDRESS.matcher(emailTxt).matches()){
        Toast.makeText( context: this@SignUpActivity, text: "The email inserted is not a defined email address.", Toast.LENGTH_SHORT).show()
    }else if(usernameTxt.isEmpty()){
        Toast.makeText( context: this@SignUpActivity, text: "Please insert a username", Toast.LENGTH_SHORT).show()
    }else if(usernameLowerCase.contains( other: "admin")){
        Toast.makeText( context: this@SignUpActivity, text: "This username $usernameTxt is reserved, please enter a new username.", Toast.LENGTH_SHORT).show()
    }else if(usernameTxt.length < 4 || usernameTxt.length > 12) {
        Toast.makeText( context: this@SignUpActivity, text: "Please enter a username between 4 - 12 characters.", Toast.LENGTH_SHORT).show()
    }else if (passwordTxt.isEmpty()) {
        Toast.makeText( context: this@SignUpActivity, text: "Please insert a Password", Toast.LENGTH_SHORT).show()
    }else if (passwordTxt.length < 8 || passwordTxt.length > 16){
        Toast.makeText( context: this@SignUpActivity, text: "Please insert a Password between 8 - 16 characters.", Toast.LENGTH_SHORT).show()
    }else if (conPasswordTxt.isEmpty()){
        Toast.makeText( context: this@SignUpActivity, text: "The confirm password is empty.", Toast.LENGTH_SHORT).show()
    }else if(passwordTxt != conPassword.text.toString().trim()){
        Toast.makeText( context: this@SignUpActivity, text: "The confirm password inserted is incorrect.", Toast.LENGTH_SHORT).show()
    } else{
        registerUser(usernameTxt, emailTxt, passwordTxt)
    }
}
```

Figure 67 Sign Up Validation Code

In Figure 66 Sign up illustrates four separate text fields where the user must follow the rules set in the sign-up validation process shown in Figure 67 Sign Up Validation Code. Firstly, once the user enters all the information and selects the sign-up button, the code will then activate. The text fields listed email, username, password and confirm password will be converted to a string and begins the validation process using conditional statements.

If email, username, password and confirm password is empty, a message will indicate which of the text fields are empty.

If the email address is not a valid email address using the Kotlin built in email validation, a message will indicate the email inserted is not a defined email address.

If the username is equal to “admin” after converting the string to lowercase, a message will appear indicating the username is reserved.

If the password is less than 8 or more than 16 characters, a message will indicate the password must be 8 – 16 characters.

If the password and confirm password is not identical, a message will indicate the confirm password inserted is incorrect.

If the user already has an account, they could simply select the “Already have an account? Sign In” button.

Once the user passes valid details into the sign-up process, Firebase Realtime database will be initiated for that specific user with their userId, username, empty profile image, status (predefined as “Hey, UoWsa is the future!”), empty study course, empty campus location and connectivity set as Offline. Once this process is complete, the user will be directed to the login page. If an error occurs, a message will be indicated, “Failed to Register”.

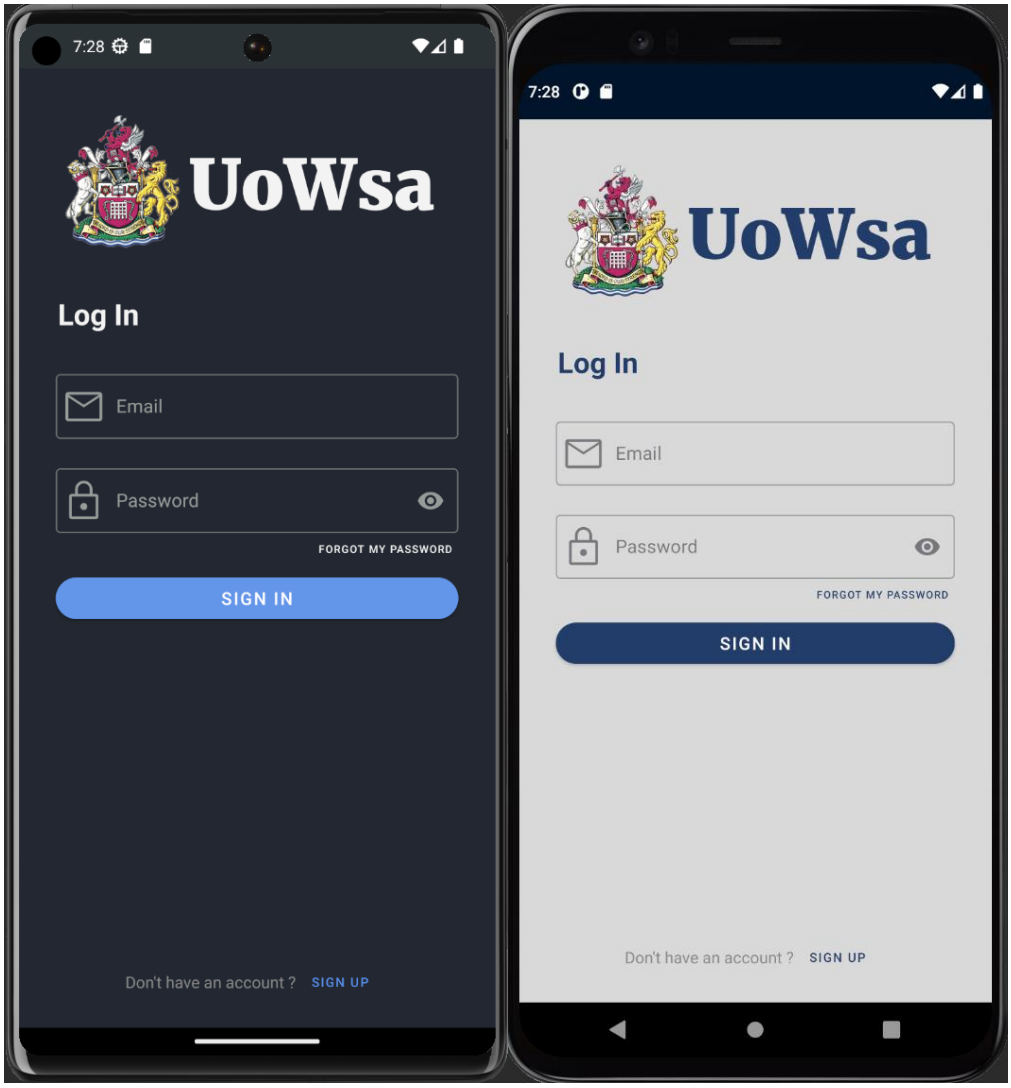


Figure 68 Login

```
auth = Firebase.auth

forgotPassword.setOnClickListener { it: View!
    val intentHome = Intent( packageContext: this, ForgotPasswordActivity::class.java)
    startActivity(intentHome)
}

registerBtn.setOnClickListener { it: View!
    val intentHome = Intent( packageContext: this, SignUpActivity::class.java)
    startActivity(intentHome)
}

signInBtn.setOnClickListener { it: View!
    if (email.text.toString().trim().isEmpty()) {
        Toast.makeText( context: this@LoginActivity, text: "Please insert an email", Toast.LENGTH_SHORT).show()
    } else if (password.text.toString().trim().isEmpty()) {
        Toast.makeText( context: this@LoginActivity, text: "Please insert a password", Toast.LENGTH_SHORT).show()
    }else{
        auth.signInWithEmailAndPassword(email.text.toString().trim(), password.text.toString().trim()).addOnCompleteListener { task ->
            if (task.isSuccessful){
                Toast.makeText( context: this@LoginActivity, text: "You've signed in successfully", Toast.LENGTH_SHORT).show()
                val intentHome = Intent( packageContext: this, HomeActivity::class.java)
                intentHome.flags = Intent.FLAG_ACTIVITY_NEW_TASK or Intent.FLAG_ACTIVITY_CLEAR_TASK
                intentHome.putExtra( name: "user_id", auth.currentUser!!.uid)
                val database = FirebaseDatabase.getInstance().getReference( path: "Users").child(auth.currentUser!!.uid).child( pathString: "connectivityStatus").setValue("Online")
                startActivity(intentHome)
                finish()
            }else{
                Toast.makeText( context: this@LoginActivity, text: "Failed to Login", Toast.LENGTH_SHORT).show()
            }
        }
    }
}
```

Figure 69 Login Code

In Figure 68 includes two text fields for email address and password followed by a “Forgot My Password” and “Sign In” button. The button at the bottom, “Don’t have an account? Sign Up” directs the user to the Sign-up page illustrated in Figure 66 Sign Up. Figure 69 Login Code illustrates the code behind the login validation process. If the email and password text fields are empty, a message will indicate whether the email and password fields are empty. If the user enters the wrong email or password, a message will indicate “Failed to Login”. If the user enters the email and password correctly using Google Firebase Authentication, the user will be logged in and directed to the home page. FLAG_ACTIVITY_NEW_TASK or Intent.FLAGACTIVITY _CLEAR_TASK will ensure if the user cannot go back to the sign in page using the back button on their Android Mobile phone. In order to go back to the sign in, the user must use the logout button. Once the user logs into their account, their connectivity status will be set as Online in the database.

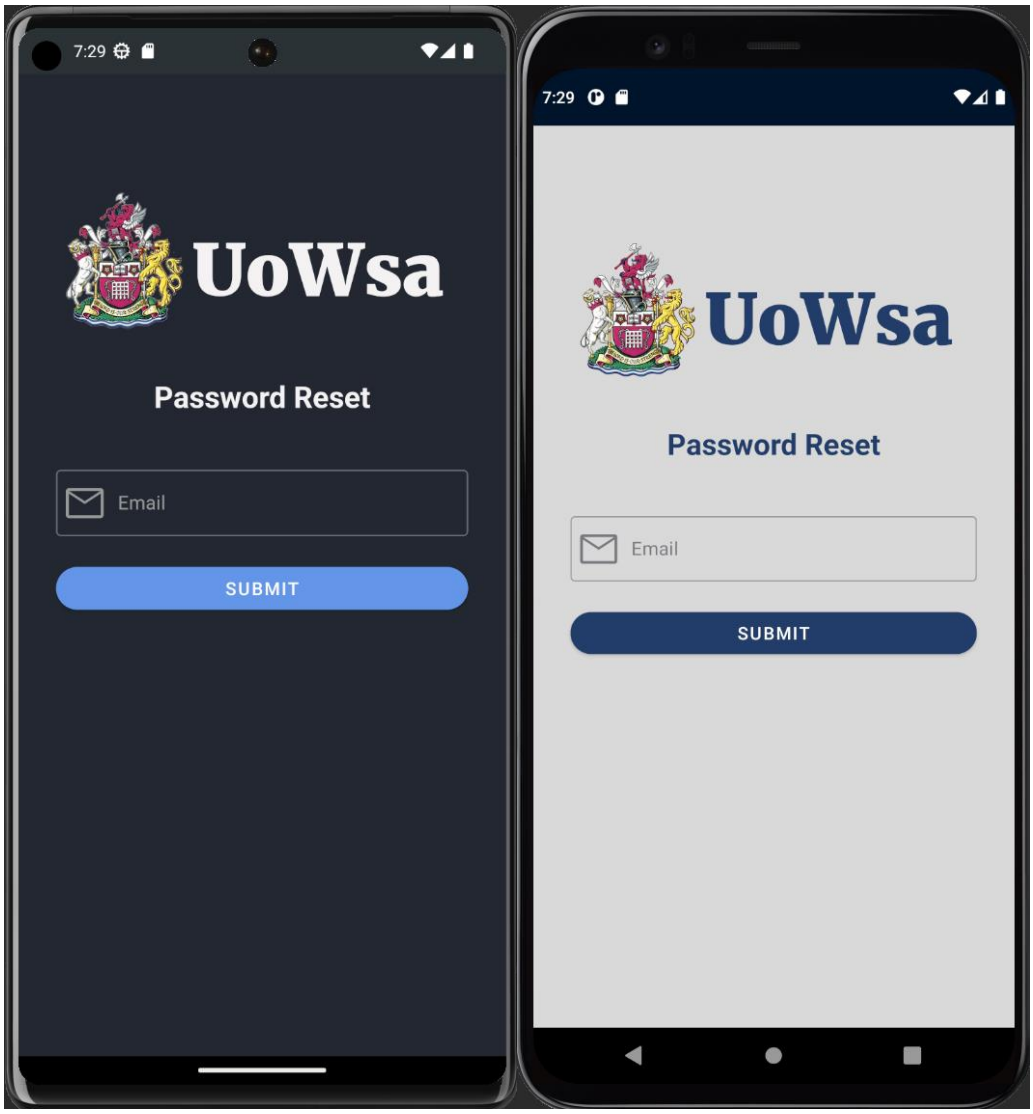


Figure 70 Password Recovery

```
val email = findViewById<EditText>(R.id.email)
val resetBtn = findViewById<Button>(R.id.resetBtn)

auth = Firebase.auth

resetBtn.setOnClickListener { it: View!
    if (email.text.toString().trim().isEmpty()){
        Toast.makeText( context: this@ForgotPasswordActivity,  text: "Please insert an email", Toast.LENGTH_SHORT).show()
    }else{
        auth.sendPasswordResetEmail(email.text.toString().trim()).addOnCompleteListener{task ->
            if (task.isSuccessful){
                Toast.makeText( context: this@ForgotPasswordActivity,  text: "Email sent successfully to reset your password", Toast.LENGTH_SHORT).show()
                finish()
            }else{
                Toast.makeText( context: this@ForgotPasswordActivity, task.exception!!.message.toString(), Toast.LENGTH_SHORT).show()
                finish()
            }
        }
    }
}
```

Figure 71 Password Recovery Code

When the user selects “Forgot My Password” illustrated in Figure 68 Login, the user will be directed to the password recovery page shown in Figure 70. A singular text field and Reset button will send an email to the user’s email address to reset their password. Figure 71 ensures the user passes validation safety checks.

If the email text field is empty, a message will indicate “Please insert an email”.

If the user enters a valid email address under Firebase Authentication, an email will be sent to the user based on the template in Figure 52 Firebase Authentication Password Recovery Template. Once the user completes this process, the user will be directed to the login page. An email will be sent to the user illustrated in Figure 72 Password Reset Email. Once the user clicks on the link inside the email the user a screen will pop up to prompt the user to enter a new password, this is illustrated in Figure 73 Reset Your Password.

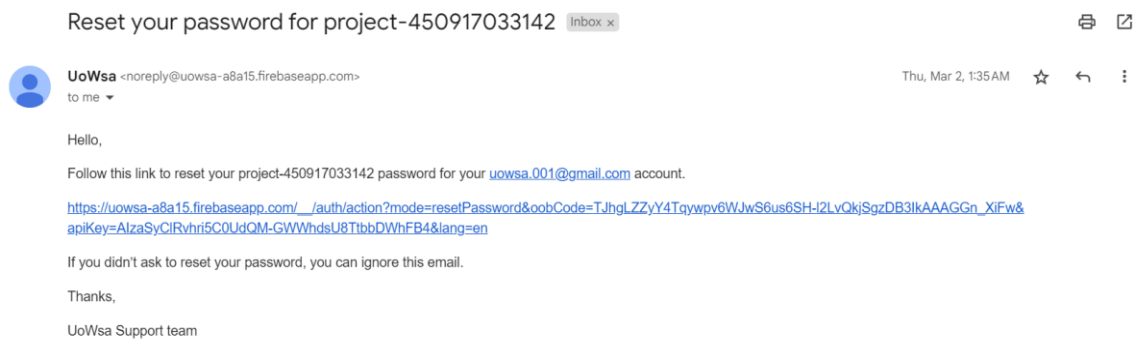


Figure 72 Password Reset Email

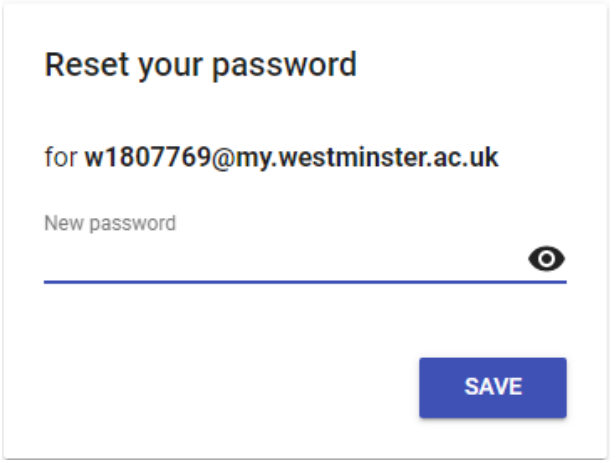


Figure 73 Reset Your Password

6.3.2 Implementation – Home and Navigation Menu

The home page is the central hub of the mobile application. It allows the user to branch out to other areas of the application easily, however a navigation bar is also included with four buttons for productivity and better UI and UX design. Home, Profile, UoW Message and Settings navigation menu icon buttons are used throughout every single page after user login. These are the four buttons which is believed to be the most essential for the application for users to complete their task swiftly.

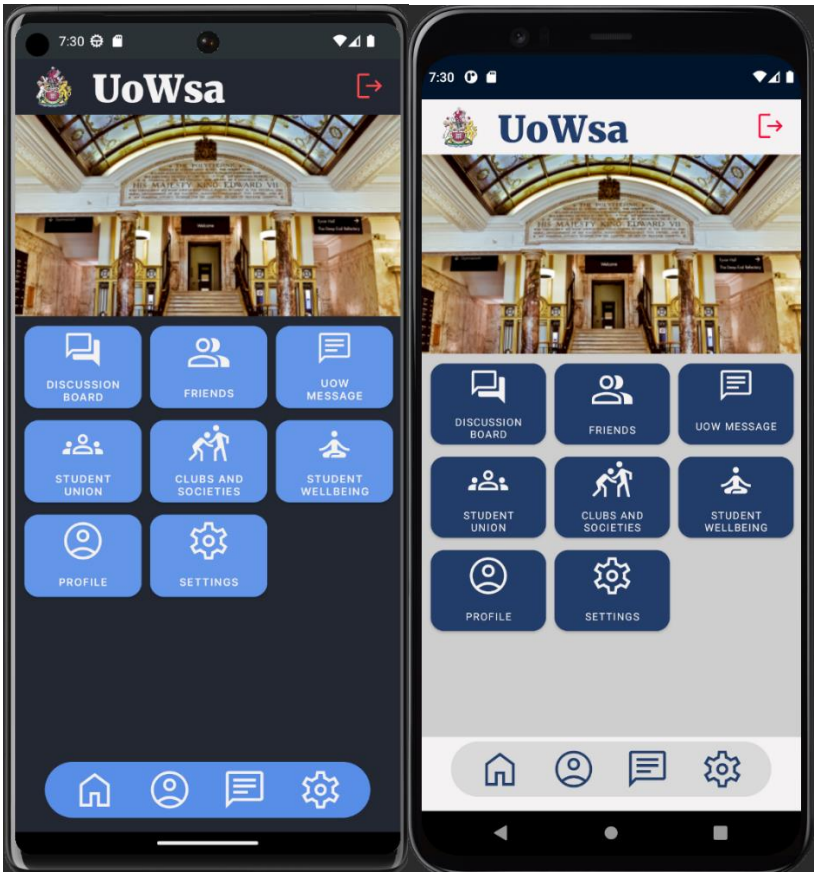


Figure 74 Home Page and Navigation Menu

```
val discussionBoard = findViewById<Button>(R.id.discussionBoard)
val friends = findViewById<Button>(R.id.friends)
val uowMessage = findViewById<Button>(R.id.uowMessage)
val studentUnion = findViewById<Button>(R.id.studentUnion)
val clubsAndSociety = findViewById<Button>(R.id.clubsAndSocieties)
val studentWellBeing = findViewById<Button>(R.id.studentWellbeing)
val profile = findViewById<Button>(R.id.profile)
val settings = findViewById<Button>(R.id.settings)

val homeBtn = findViewById<ImageButton>(R.id.navHomeBtn)
val profileBtn = findViewById<ImageButton>(R.id.navProfileBtn)
val uowMessageBtn = findViewById<ImageButton>(R.id.navUowMessage)
val settingsBtn = findViewById<ImageButton>(R.id.navSettings)

val logoutBtn = findViewById<ImageButton>(R.id.logoutBtn)

discussionBoard.setOnClickListener { it: View!
    val intentDiscussionBoard = Intent( packageContext: this, DiscussionBoardActivity::class.java)
    startActivity(intentDiscussionBoard)
}

friends.setOnClickListener { it: View!
    val intentFriends = Intent( packageContext: this, FriendsActivity::class.java)
    startActivity(intentFriends)
}

uowMessage.setOnClickListener { it: View!
    val intentUoWMessage = Intent( packageContext: this, UoWMessageActivity::class.java)
    startActivity(intentUoWMessage)
}

studentUnion.setOnClickListener { it: View!
    val intentStudentUnion = Intent( packageContext: this, StudentUnionActivity::class.java)
    startActivity(intentStudentUnion)
}

clubsAndSociety.setOnClickListener { it: View!
    val intentClubsAndSocieties = Intent( packageContext: this, StudentCommunityActivity::class.java)
    startActivity(intentClubsAndSocieties)
}

studentWellBeing.setOnClickListener { it: View!
    val intentClubsAndSocieties = Intent( packageContext: this, StudentWellbeingActivity::class.java)
    startActivity(intentClubsAndSocieties)
}

profile.setOnClickListener { it: View!
    val intentProfile = Intent( packageContext: this, ProfileActivity::class.java)
    startActivity(intentProfile)
}

settings.setOnClickListener { it: View!
    val intentSettings = Intent( packageContext: this, SettingsActivity::class.java)
    startActivity(intentSettings)
}
```

Figure 75 Home Page Code

```
homeBtn.setOnClickListener{ it: View!
    val navIntentHome = Intent( packageContext: this, HomeActivity::class.java)
    startActivity(navIntentHome)
}

profileBtn.setOnClickListener { it: View!
    val navIntentProfile = Intent( packageContext: this, ProfileActivity::class.java)
    startActivity(navIntentProfile)
}

uowMessageBtn.setOnClickListener { it: View!
    val navIntentUoWMessage = Intent( packageContext: this, UoWMessageActivity::class.java)
    startActivity(navIntentUoWMessage)
}

settingsBtn.setOnClickListener { it: View!
    val navIntentSettings = Intent( packageContext: this, SettingsActivity::class.java)
    startActivity(navIntentSettings)
}

logoutBtn.setOnClickListener { it: View!
    val builder: AlertDialog.Builder = AlertDialog.Builder( context: this, R.style.DialogTheme)
    val dialog: AlertDialog = builder.setTitle("Logout")
        .setMessage("Are you sure you want to logout?")
        .setPositiveButton( text: "Cancel") {
                dialog, _ ->
            dialog.dismiss()
        }
        .setNegativeButton( text: "Logout") { dialog, _ ->
            val database = FirebaseDatabase.getInstance().getReference( path: "Users").child(auth.currentUser!!.uid).child( pathString: "connectivityStatus").setValue("Offline")
            signOut()
            dialog.dismiss()
        }
        .create()
    dialog.show()

    dialog.getButton(AlertDialog.BUTTON_POSITIVE).setTextColor(Color.BLACK)
    dialog.getButton(AlertDialog.BUTTON_NEGATIVE).setTextColor(Color.RED)
}

imageExists()
```

Figure 76 Navigation Menu and Logout Dialog Code

```
private fun imageExists() {
    val user: FirebaseUser? = auth.currentUser
    val userId: String = user!!.uid

    val databaseImage = FirebaseDatabase.getInstance().getReference( path: "Users").child(userId).child( pathString: "profileImage")
    databaseImage.get().addOnSuccessListener { it: DataSnapshot!
        val imageExist = it.value.toString()
        if (imageExist == ""){
            val storageRef = FirebaseStorage.getInstance().getReference( location: "appImage").child( pathString: "ic_launcher.png")
            storageRef.downloadUrl.addOnSuccessListener {task->
                val url = task.toString()
                databaseReference = FirebaseDatabase.getInstance().getReference( path: "Users").child(userId)
                databaseReference.child( pathString: "profileImage").setValue(url)
            }
        }
    }
}

private fun signOut(){
    auth.signOut()
    val intentHome = Intent( packageContext: this, MainActivity::class.java)
    intentHome.flags = Intent.FLAG_ACTIVITY_NEW_TASK or Intent.FLAG_ACTIVITY_CLEAR_TASK
    startActivity(intentHome)
    finish()
}
```

Figure 77 Home Page Profile Image Code

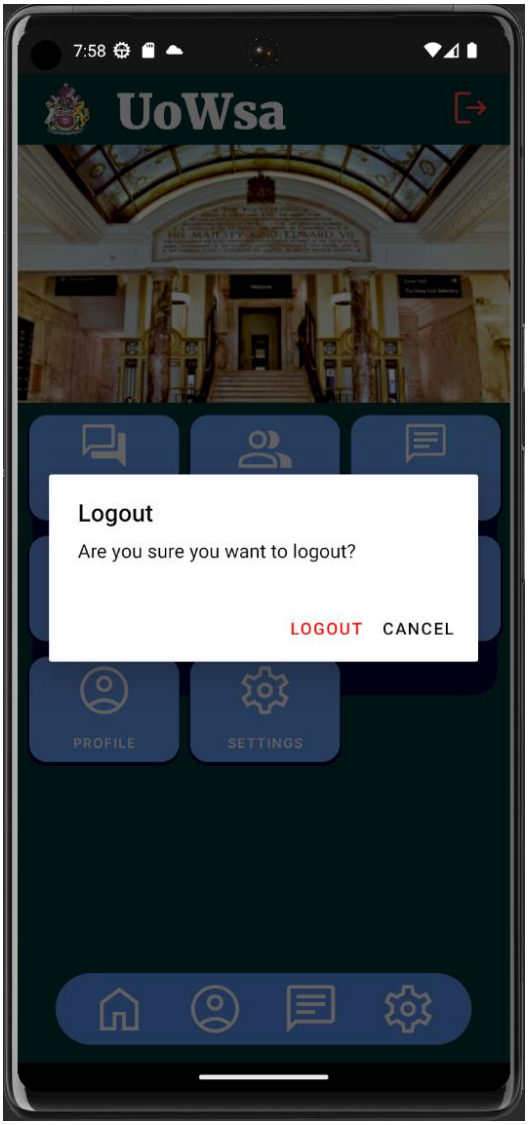


Figure 78 Logout Dialog

Figure 75 Home Page Code shows all of the buttons directing the user to other areas of the application depending on which button the user selects along with the “Logout” icon button on the top right. Figure 76 Navigation Menu and Figure 78 Logout Dialog Code shows all of the four buttons which the user can select to direct the user to their destination from any page of the application. The user can choose whether to logout with the function signout(), where it will also instruct the database to set the value of the user’s connectivity status as “Offline” or cancel using the dialog.dismiss() function.

When the user creates an account, they do not have any profile images, therefore the imageExists() function checks if the user has inserted a profile image previously in Figure 77 Home Page Profile Image Code, if the user has never uploaded their own profile image, a pre-defined temporary image will be set into the database for the user.

6.3.3 Implementation – User Profile

The user profile is where the user can view the information they have set from login or the update profile section of the application. The information includes, profile picture, username, status, campus location and field of study.

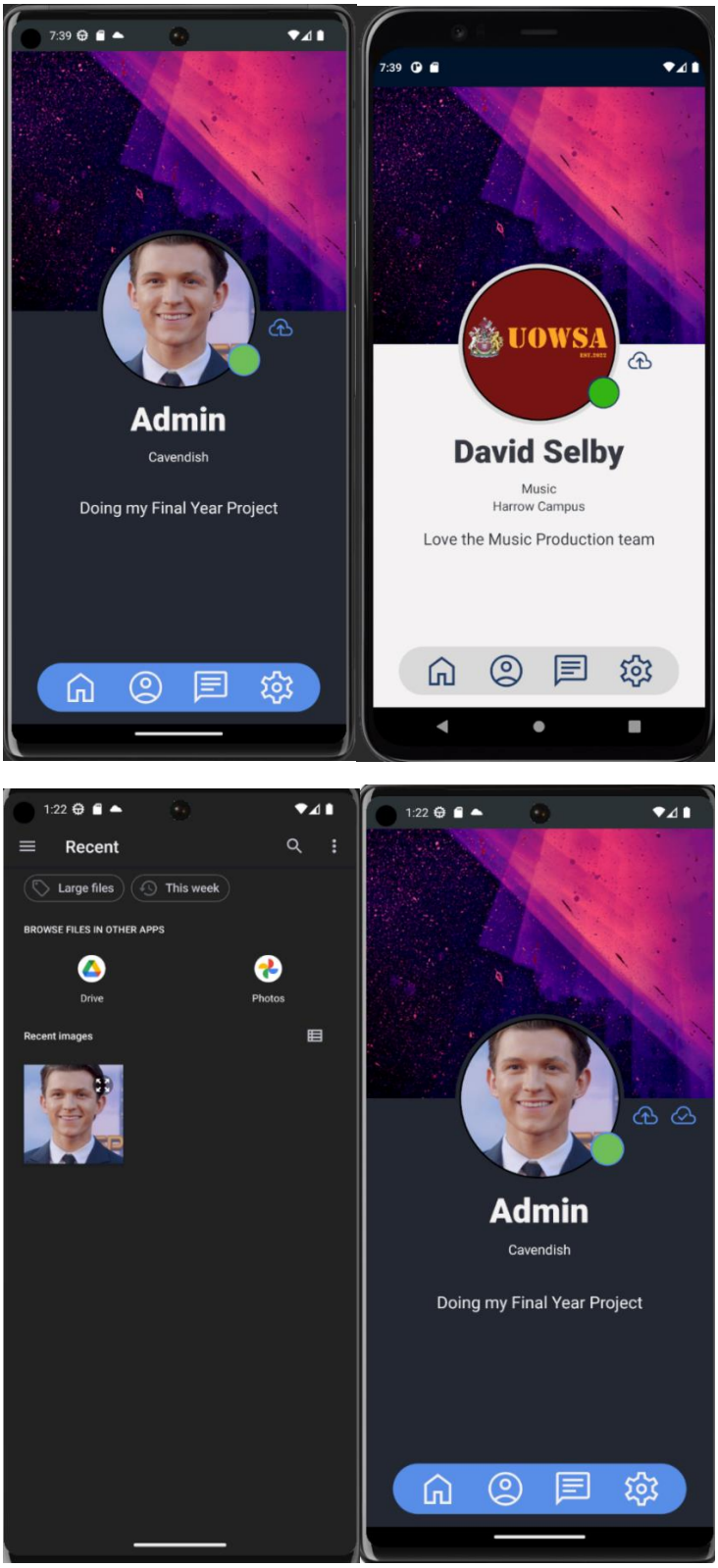


Figure 79 User Profile

```
val user: FirebaseUser? = auth.currentUser
val userId: String = user!!.uid
val connectivityStatus = findViewById<TextView>(R.id.connectivityStatus)
val databaseRefresh = FirebaseDatabase.getInstance().getReference( path: "Users").child(userId)
databaseRefresh.child( pathString: "connectivityStatus").get().addOnSuccessListener { it: DataSnapshot!
    val connectivityStatusInfo = it.value.toString()
    if (connectivityStatusInfo == "Online"){
        connectivityStatus.setBackgroundResource(R.drawable.connectivity_status_online)
    }else{
        connectivityStatus.setBackgroundResource(R.drawable.connectivity_status_offline)
    }
}

profileImage = findViewById(R.id.profileImage)
val editImageBtn = findViewById<ImageButton>(R.id.editImageBtn)
usernameView = findViewById(R.id.namePlaceholder)
statusMessageView = findViewById(R.id.statusMessage)
uploadBtn = findViewById(R.id.uploadBtn)
optional1 = findViewById(R.id.optional1)
optional2 = findViewById(R.id.optional2)

val navHome = findViewById<ImageButton>(R.id.navHomeBtn)
val navProfile = findViewById<ImageButton>(R.id.navProfileBtn)
val navMessage = findViewById<ImageButton>(R.id.navUowMessage)
val navSettings = findViewById<ImageButton>(R.id.navSettings)

readImage()
readUsername()
readStatus()
readOptionals()

editImageBtn.setOnClickListener { it: View!
    selectImage()
}

uploadBtn?.setOnClickListener { it: View!
    clearDatabaseImage()
    uploadImage()
    if (uploadBtn?.isEnabled == true) {
        uploadBtn?.visibility = View.INVISIBLE
    }
}
```

Figure 80 User Profile Code

When the user enters the user account profile page the readImage(), readUsername(), readStatus() and readOptionals are automatically activated to ensure those are the only automatic functions that will be executed immediately once the user lands on the page.

```
private fun readImage() {
    val user: FirebaseUser? = auth.currentUser
    val userId: String = user!!.uid
    val storageRef = FirebaseStorage.getInstance().getReference( location: "images")
        .child(FirebaseAuth.getInstance().currentUser!!.uid)

    val databaseImage =
        FirebaseDatabase.getInstance().getReference( path: "Users").child(userId).child( pathString: "profileImage")
            .get().addOnSuccessListener { it: DataSnapshot!
                val imageExist = it.value.toString()
                if (imageExist == "") {
                    profileImage?.setImageResource(R.mipmap.ic_launcher)
                } else if (imageExist.contains(userId)) {
                    val file = File.createTempFile( prefix: "tempFileImage", suffix: "jpg")
                    storageRef.getFile(file).addOnSuccessListener { it: FileDownloadTask.TaskSnapshot!
                        val bitmap = BitmapFactory.decodeFile(file.absolutePath)
                        profileImage?.setImageBitmap(bitmap)
                    }
                } else {
                    val tempStorageRef = FirebaseStorage.getInstance().getReference( location: "appImage")
                        .child( pathString: "ic_launcher.png")
                    val file = File.createTempFile( prefix: "tempFileImage", suffix: "jpg")
                    tempStorageRef.getFile(file).addOnSuccessListener { it: FileDownloadTask.TaskSnapshot!
                        val bitmap = BitmapFactory.decodeFile(file.absolutePath)
                        profileImage?.setImageBitmap(bitmap)
                    }
                }
            }
}
```

Figure 81 Read Profile Image Code

The image read would be the default image set if the user selects their profile page for the very first time. The default image was set from Figure 77 Home Page Profile Image Code. Regardless, the image would be loaded using the path of the database using a media token to connect the Firebase Storage with the client application. Every time the user enters the profile page, readImage() function will download the image into the Circle Image View shown in Figure 79 User Profile.

The readImage() function reads the user's profile image from Firebase storage and displays into the circle image view. It then checks if the user has an existing profile image using the values from Firebase Realtime database. The function temporarily stores the downloaded image. It then decodes it using a Bitmap and sets the Bitmap as the circle image view. If the user doesn't have an image at all, a safety check is used to replace an empty image area to the App Icon.

```
private fun selectImage() {
    val intent = Intent()
    intent.type = "image/*"
    intent.action = Intent.ACTION_GET_CONTENT

    startActivityForResult(intent, requestCode: 100)
}

override fun onActivityResult(requestCode: Int, resultCode: Int, data: Intent?) {
    super.onActivityResult(requestCode, resultCode, data)

    if (requestCode == 100 && resultCode == RESULT_OK) {
        uploadImageUri = data?.data!!
        profileImage?.setImageURI(uploadImageUri)
        Log.d( tag: "tag", uploadImageUri.toString())
        uploadBtn?.visibility = View.VISIBLE
    }
}
```

Figure 82 Upload Image to User Profile Code

When the user selects the upload button, the user will be directed to their local photo gallery where they can upload their image and view the image from Figure 79 User Profile. Figure 82 Upload Image to User Profile Code illustrates the process of opening the photo gallery and setting the image into the Circle Image View.

When the user opens their photo library from the selectImage() function, the startActivityForResult() function will create an intent and request a code of 100 to start the activity and wait for a result. The onActivityResult() function will then be activated and checks if the request code is equal to 100 set from the startActivityForResult(). If the result returns “Result_OK” it retrieves the URI from the selected image and sets the image to the circle Image View.

```
private fun clearDatabaseImage() {
    val user = auth.currentUser?.uid.toString()
    val storageRef = FirebaseStorage.getInstance().getReference( location: "images")
        .child(FirebaseAuth.getInstance().currentUser!!.uid)

    //Delete image if there are any before uploading a new image.
    storageRef.delete().addOnSuccessListener { it: Void!
        Toast.makeText( context: this@ProfileActivity, text: "Replacing Image...", Toast.LENGTH_SHORT)
            .show()
    }

}

private fun uploadImage() {
    val user = auth.currentUser?.uid.toString()
    Log.d( tag: "tag", user)
    val storageRef = FirebaseStorage.getInstance().getReference( location: "images")
        .child(FirebaseAuth.getInstance().currentUser!!.uid)

    val userId = auth.currentUser?.uid!!

    databaseReference = FirebaseDatabase.getInstance().getReference( path: "Users").child(userId)

    val uploadTask = storageRef.putFile(uploadImageUri)
    uploadTask.continueWith { it: Task<UploadTask.TaskSnapshot!>
        storageRef.downloadUrl
    }.addOnCompleteListener { it ->
        if (it.isSuccessful) {
            it.result.addOnSuccessListener { task ->
              val hashMap:HashMap<String,String> = HashMap()
                Log.d( tag: "tag", uploadImageUri.toString())
                val url = task.toString()
              hashMap["profileImage"] = url
              Log.d("tag", url)
              databaseReference.updateChildren(hashMap as Map<String, Any>)
                databaseReference.child( pathString: "profileImage").setValue(url)
                    .addOnCompleteListener(this) { it: Task<Void!>
                        if (it.isSuccessful) {
                            Toast.makeText(
                                context: this,
                                text: "Image is successfully uploaded",
                                Toast.LENGTH_SHORT
                            ).show()
                        }
                    }
            }
        }
    }
```

Figure 83 Confirm Upload Image Code

Once the user selects the image from their local gallery, the confirm upload button will appear. If the user does not select the confirm button and enters a new page the progress to change their profile image will not be saved. The user must confirm the changes using the confirm upload button to replace their previous image.

Figure 83 Confirm Upload Image Code shows how the current image is being removed by deleting the image URL from the database using the clearDatabaseImage() based on the path set for the user. The uploadImage() function updates the URL set in the database and calls for the corresponding image file from the storage. It then uses the putFile() method with the storage reference to start the upload process.

Once the upload is completed, the continueWith() function on Upload Task to the URL of the uploaded image. The onSucessListener gets the URL of the uploaded image and updates the profile image for the user.

```
private fun readUsername() {
    val user: FirebaseUser? = auth.currentUser
    val userId: String = user!!.uid
    Log.d( tag: "tag", userId)

    val databaseUsers = FirebaseDatabase.getInstance().getReference( path: "Users")
        .child(FirebaseAuth.getInstance().currentUser!!.uid).child( pathString: "userName")
    Log.d( tag: "tag", databaseUsers.toString())

    databaseUsers.addValueEventListener(object : ValueEventListener {
        override fun onDataChange(snapshot: DataSnapshot) {
            val username = snapshot.getValue(String::class.java).toString()
            Log.d( tag: "tag", username)
            usernameView?.setText(username)
        }

        override fun onCancelled(error: DatabaseError) {
            Toast.makeText(
                 context: this@ProfileActivity,
                 text: "Error retrieving username...",
                Toast.LENGTH_SHORT
            ).show()
        }
    })
}
```

Figure 84 Read Username Code

```
private fun readStatus() {
    val user: FirebaseUser? = auth.currentUser
    val userId: String = user!!.uid
    Log.d( tag: "tag", userId)

    val databaseStatus = FirebaseDatabase.getInstance().getReference( path: "Users")
        .child(FirebaseAuth.getInstance().currentUser!!.uid).child( pathString: "status")
    Log.d( tag: "tag", databaseStatus.toString())

    databaseStatus.addValueEventListener(object : ValueEventListener {
        override fun onDataChange(snapshot: DataSnapshot) {
            val status = snapshot.getValue(String::class.java).toString()
            Log.d( tag: "tag", status)
            statusMessageView?.setText(status)
        }

        override fun onCancelled(error: DatabaseError) {
            Toast.makeText(
                 context: this@ProfileActivity,
                 text: "Error retrieving user status...",
                Toast.LENGTH_SHORT
            ).show()
        }
    })
}
```

Figure 85 Read Status Code

```
private fun readOptionals() {
    val user: FirebaseUser? = auth.currentUser
    val userId: String = user!!.uid
    Log.d( tag: "tag", userId)

    val databaseStudyCourse = FirebaseDatabase.getInstance().getReference( path: "Users")
        .child(FirebaseAuth.getInstance().currentUser!!.uid).child( pathString: "studyCourse")
    Log.d( tag: "tag", databaseStudyCourse.toString())

    databaseStudyCourse.addValueEventListener(object : ValueEventListener {
        override fun onDataChange(snapshot: DataSnapshot) {
            val studyCourse = snapshot.getValue(String::class.java).toString()
            Log.d( tag: "tag", studyCourse)
            if (studyCourse == "") {
                val databaseCampusLocation1 =
                    FirebaseDatabase.getInstance().getReference( path: "Users")
                        .child(FirebaseAuth.getInstance().currentUser!!.uid)
                        .child( pathString: "campusLocation")
                Log.d( tag: "tag", databaseCampusLocation1.toString())

                databaseCampusLocation1.addValueEventListener(object : ValueEventListener {
                    override fun onDataChange(snapshot: DataSnapshot) {
                        val campusLocation =
                            snapshot.getValue(String::class.java).toString()
                        Log.d( tag: "tag", campusLocation)
                        optional1?.setText(campusLocation)
                        optional1?.visibility = View.VISIBLE
                    }

                    override fun onCancelled(error: DatabaseError) {
                    }

                })
            } else {
                optional1?.setText(studyCourse)
                optional1?.visibility = View.VISIBLE
                val databaseCampusLocation =
                    FirebaseDatabase.getInstance().getReference( path: "Users")
                        .child(FirebaseAuth.getInstance().currentUser!!.uid)
                        .child( pathString: "campusLocation")
                Log.d( tag: "tag", databaseCampusLocation.toString())

                databaseCampusLocation.addValueEventListener(object : ValueEventListener {
                    override fun onDataChange(snapshot: DataSnapshot) {
                        val campusLocation =
                            snapshot.getValue(String::class.java).toString()
                        Log.d( tag: "tag", campusLocation)
                        optional2?.setText(campusLocation)
                        optional2?.visibility = View.VISIBLE
                    }

                    override fun onCancelled(error: DatabaseError) {
                    }
                })
            }
```

Figure 86 Read Optionals Code

In Figure 84 Read Username Code, Figure 85 Read Status Code, Figure 86 Read Optionals Code consists of three private functions to retrieve data from the database.

readUsername() function users the current logged-in user's unique identifier UID from firebase authentication and relates to the username from Firebase Realtime database. The retrieved username is then converted into text view for display.

The readStatus() function works similarly with the readUsername() function, however it addresses the status text to a separate text view.

The readOptionals() function retrieves two optional information, studyCourse and campusLocation. If the user has not updated their study course in the update profile section, campus location data will take the text view area allocated for study course.

6.3.4 Implementation – Settings (Update Profile, Blocked Users, Administrators, Contact Us, Delete Account, Logout)

This section of the application is divided into three categories: account, help and privacy. The account category comprises options such as updating the user profile, unblocking users, and contact administrators. The help category includes features like contacting support, accessing the help center, reviewing the terms of service, and reading about the company. The privacy category includes function such as, deleting the account and logging out. In this subchapter, we will illustrate how to implement these different sections.

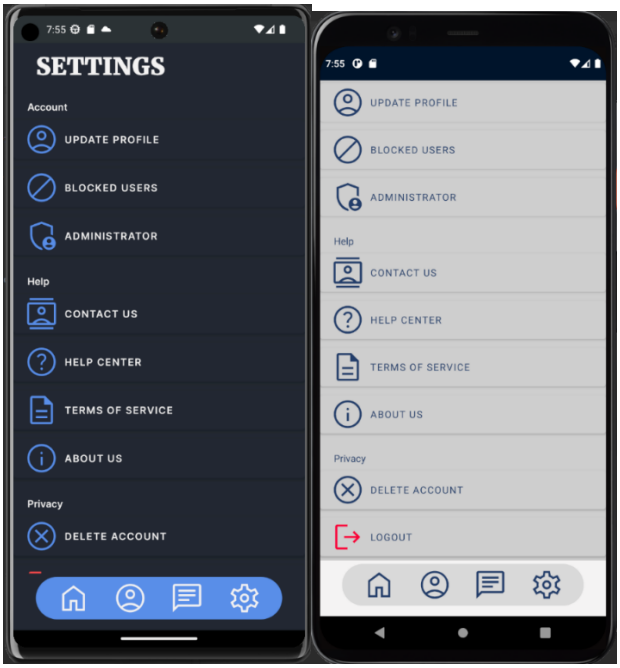


Figure 87 Settings

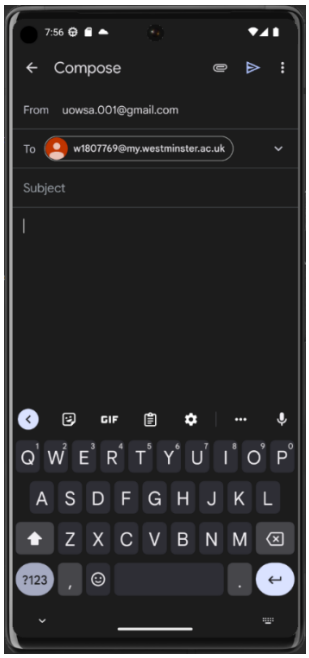


Figure 88 Contact Us

```
helpCenter.setOnClickListener { it: View!
    val intentHome = Intent( packageContext: this, HelpCenterActivity::class.java)
    startActivity(intentHome)
}

admin.setOnClickListener{ it: View!
    val intentHome = Intent( packageContext: this, ContactAdminActivity::class.java)
    startActivity(intentHome)
}

contactUs.setOnClickListener { it: View!
    contactEmail()
}

aboutUs.setOnClickListener { it: View!
    val intentHome = Intent( packageContext: this, AboutUsActivity::class.java)
    startActivity(intentHome)
}

tos.setOnClickListener { it: View!
    val intentHome = Intent( packageContext: this, ToSActivity::class.java)
    startActivity(intentHome)
}

blockedList.setOnClickListener { it: View!
    val intentHome = Intent( packageContext: this, BlockedListActivity::class.java)
    startActivity(intentHome)
}

deleteAccount.setOnClickListener { it: View!
    val intentHome = Intent( packageContext: this, DeleteAccountActivity::class.java)
    startActivity(intentHome)
      deleteUoWsaAccount()
}

logout.setOnClickListener { it: View!
    val builder: AlertDialog.Builder = AlertDialog.Builder( context: this, R.style.DialogTheme)
    val dialog: AlertDialog = builder.setTitle("Logout")
        .setMessage("Are you sure you want to logout?")
        .setPositiveButton( text: "Cancel") {
                dialog, _ ->
            dialog.dismiss()
        }
        .setNegativeButton( text: "Logout") { dialog, _ ->
            val database = FirebaseDatabase.getInstance().getReference( path: "Users").child(auth.currentUser!!.uid).child( pathString: "connectivityStatus").setValue("Offline")
            signOut()
            dialog.dismiss()
        }
        .create()
    dialog.show()

    dialog.getButton(AlertDialog.BUTTON_POSITIVE).setTextColor(Color.BLACK)
    dialog.getButton(AlertDialog.BUTTON_NEGATIVE).setTextColor(Color.RED)
}
```

Figure 89 Settings Code

```
private fun contactEmail(){
    val emailIntent = Intent(
        Intent.ACTION_SENDTO, Uri.fromParts(
            scheme: "mailto", ssp: "w1807769@my.westminster.ac.uk", fragment: null
        )
    )
    emailIntent.putExtra(Intent.EXTRA_SUBJECT, value: "Subject")
    emailIntent.putExtra(Intent.EXTRA_TEXT, value: "Body")
    startActivity(Intent.createChooser(emailIntent, title: "Send email..."))
}

private fun signOut(){
    auth.signOut()
    val intentHome = Intent( packageContext: this, MainActivity::class.java)
    intentHome.flags = Intent.FLAG_ACTIVITY_NEW_TASK or Intent.FLAG_ACTIVITY_CLEAR_TASK
    startActivity(intentHome)
    finish()
}
```

Figure 90 Contact Us & Logout Code

Figure 89 Settings code relates to the Figure 87 Settings, it is a listed options where user can select to using OnClickListener. The Alert dialog builder function relates to a dialog box which will ask to confirm with the user if they wish to logout or cancel illustrated in Figure 78 Logout as the logout function exist in both the settings and home page. Furthermore, the signOut() calls for the function in Figure 90 Contact Us & Logout Code. When the user logs out it uses the function auth.signOut() to ensure Firebase Authentication is notified the user has logout of the application. This will then direct the user back to the starting app screen in 61 UoWsa App Start-up.

Figure 88 Contact us illustrates the target email address automatically inserted with the user able to add the subject and context to construct the email for support. Figure 90 illustrates the exact email address it targets, and it automatically opens their mailing application on the user's device.

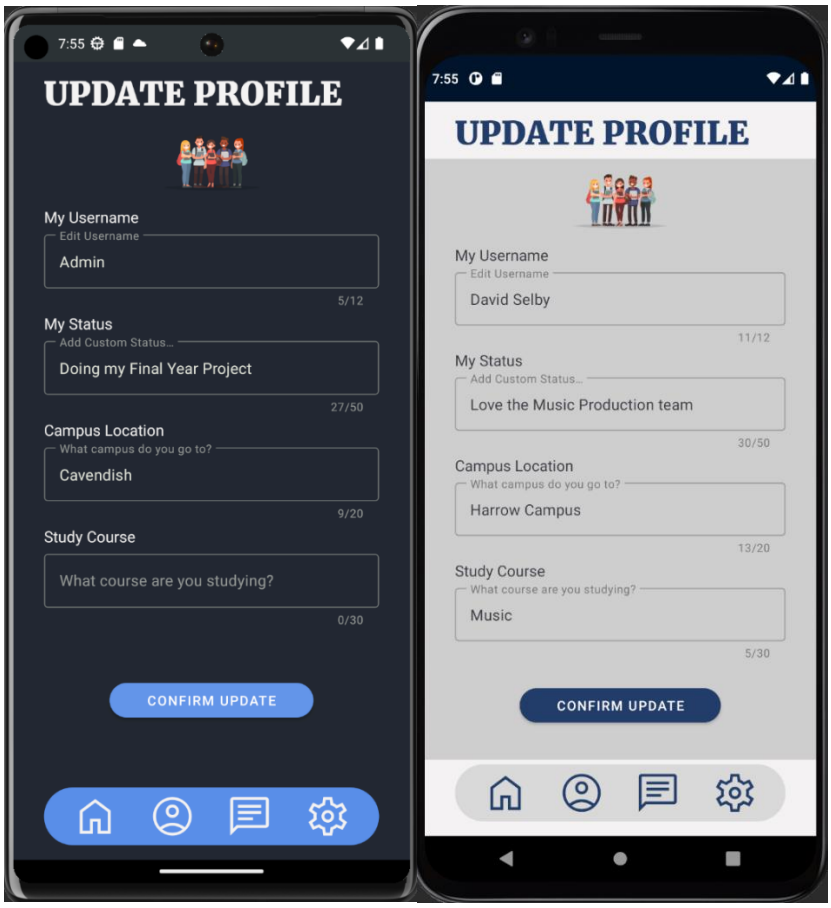


Figure 91 Update Profile

```
val usernameData = FirebaseDatabase.getInstance().getReference( path: "Users").child(userId).child( pathString: "userName")
usernameData.get().addOnSuccessListener { it: DataSnapshot!
    val name = it.value.toString()
    Log.d( tag: "tag", name)
    usernameSetText.setText(name)
}

val statusData = FirebaseDatabase.getInstance().getReference( path: "Users").child(userId).child( pathString: "status")
statusData.get().addOnSuccessListener { it: DataSnapshot!
    val statusInfo = it.value.toString()
    Log.d( tag: "tag", statusInfo)
    if (statusInfo != "Hey, UoWsa is the future!"){
        statusSetText.setText(statusInfo)
    }
}

val campusLocationData = FirebaseDatabase.getInstance().getReference( path: "Users").child(userId).child( pathString: "campusLocation")
campusLocationData.get().addOnSuccessListener { it: DataSnapshot!
    val campusLocationInfo = it.value.toString()
    Log.d( tag: "tag", campusLocationInfo)
    if (campusLocationInfo != ""){
        campusLocationSetText.setText(campusLocationInfo)
    }
}

val studyCourseData = FirebaseDatabase.getInstance().getReference( path: "Users").child(userId).child( pathString: "studyCourse")
studyCourseData.get().addOnSuccessListener { it: DataSnapshot!
    val studyCourseDataInfo = it.value.toString()
    Log.d( tag: "tag", studyCourseDataInfo)
    if (studyCourseDataInfo != ""){
        studyCourseSetText.setText(studyCourseDataInfo)
    }
}
```

Figure 92 Update Profile Pre-defined Text Code

```
updateBtn.setOnClickListener{ it: View!
    val usernameLowerCase = username.text.toString().lowercase()
    val usernameTxt = username.text.toString()
    val campusLocationTxt = campusLocation.text.toString()
    val studyCourseTxt = studyCourse.text.toString()
    if (username.text.toString().isEmpty()){
        Toast.makeText( context: this, text: "Please enter a username", Toast.LENGTH_SHORT).show()
    }else if (usernameTxt.length < 4 || usernameTxt.length > 12){
        Toast.makeText( context: this@UpdateProfileActivity, text: "Please enter a username between 4 - 12 characters", Toast.LENGTH_SHORT).show()
    } else if (usernameLowerCase == "admin"){

            Toast.makeText( context: this@UpdateProfileActivity, text: "This username $usernameTxt is reserved, please enter a new username.", Toast.LENGTH_SHORT).show()
    }else
    {
        var statusTxt = status.text.toString()
        if (status.text.toString().isEmpty()){
            statusTxt = "Hey, UoWsa is the future!"
            updateData(usernameTxt, statusTxt, campusLocationTxt, studyCourseTxt)
        }else if (statusTxt.length > 50){
            Toast.makeText( context: this@UpdateProfileActivity, text: "Your status is over 50 characters", Toast.LENGTH_SHORT).show()
        }else{
            updateData(usernameTxt, statusTxt, campusLocationTxt, studyCourseTxt)
        }
    }
}
```

Figure 93 Edit Profile Conditions Code

```
private fun setUsername(username: String){
    val user: FirebaseUser? = auth.currentUser
    val userId: String = user!!.uid
    databaseReference = FirebaseDatabase.getInstance().getReference( path: "Users").child(userId).child( pathString: "userName")

    databaseReference.setValue(username).addOnCompleteListener(this){ it: Task<Void!>
        if (it.isSuccessful){
            Toast.makeText( context: this,  text: "Username Updated Successfully", Toast.LENGTH_SHORT).show()
        }else{
            Toast.makeText( context: this,  text: "Username failed to Update", Toast.LENGTH_SHORT).show()
        }
    }
}

private fun setStatus(status: String){
    val user: FirebaseUser? = auth.currentUser
    val userId: String = user!!.uid
    databaseReference = FirebaseDatabase.getInstance().getReference( path: "Users").child(userId).child( pathString: "status")

    databaseReference.setValue(status).addOnCompleteListener(this){ it: Task<Void!>
        if (it.isSuccessful){
            Toast.makeText( context: this,  text: "Status Updated Successfully", Toast.LENGTH_SHORT).show()
        }else{
            Toast.makeText( context: this,  text: "Status failed to Update", Toast.LENGTH_SHORT).show()
        }
    }
}

private fun setCampusLocation(campusLocation: String){
    val user: FirebaseUser? = auth.currentUser
    val userId: String = user!!.uid
    databaseReference = FirebaseDatabase.getInstance().getReference( path: "Users").child(userId).child( pathString: "campusLocation")

    databaseReference.setValue(campusLocation).addOnCompleteListener(this){ it: Task<Void!>
        if (it.isSuccessful){

        }else{
            Toast.makeText( context: this,  text: "Campus Location failed to Update", Toast.LENGTH_SHORT).show()
        }
    }
}

private fun setStudyCourse(studyCourse: String){
    val user: FirebaseUser? = auth.currentUser
    val userId: String = user!!.uid
    databaseReference = FirebaseDatabase.getInstance().getReference( path: "Users").child(userId).child( pathString: "studyCourse")

    databaseReference.setValue(studyCourse).addOnCompleteListener(this){ it: Task<Void!>
        if (it.isSuccessful){

        }else{
            Toast.makeText( context: this,  text: "Study Course failed to Update", Toast.LENGTH_SHORT).show()
        }
    }
}
```

Figure 94 Update Username, Status, Campus Location and Study Course Code

Figure 91 Update Profile allows the user to review their user account profile details where the user can change existing information including, username, status, campus location and study course. Figure 92 Update Profile Pre-defined Text Code illustrates how each element is obtained from the database and automatically filled into the textview. Figure 93 runs through conditional statements similarly to the registering process, as there are certain requirements which must be met in order to update the data on the database.

When the user selects the confirm update button, it will run through all of the functions listed in Figure 94 Update Username, Status, Campus Location and Study Course Code. All of these functions have the same purpose to take each parameter to corresponding data in the database for the currently user logged in. If the user is successful or unsuccessful in making changes to their profile, a message will be displayed with updated successfully or failed to update to the corresponding update.

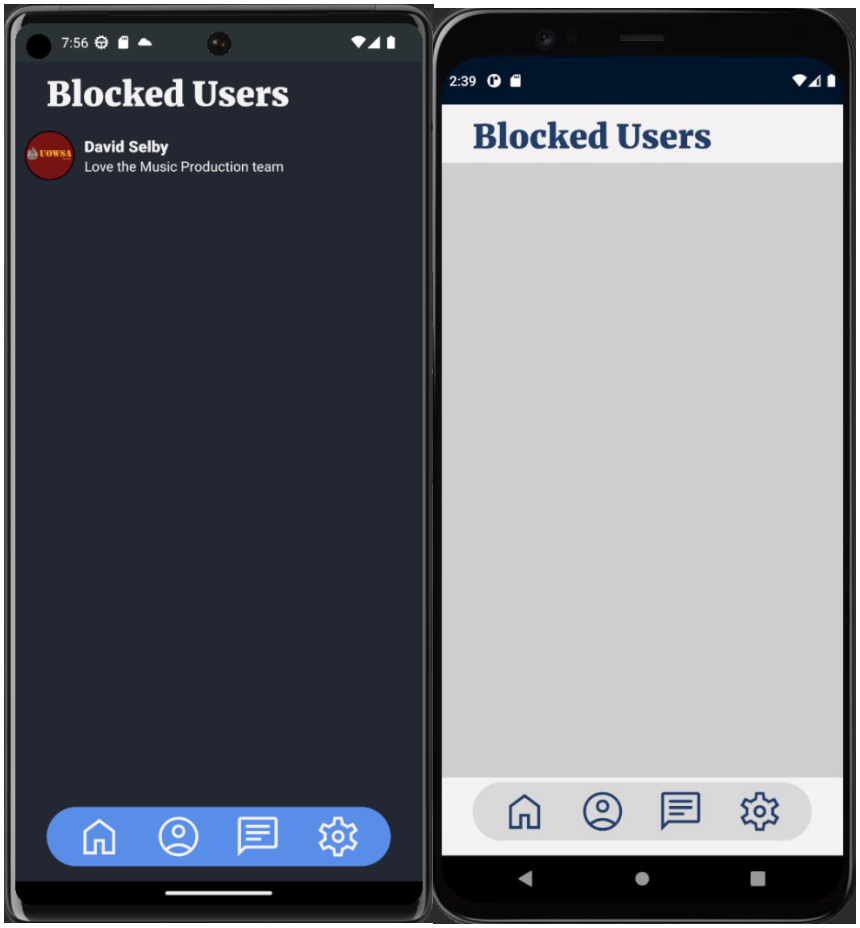
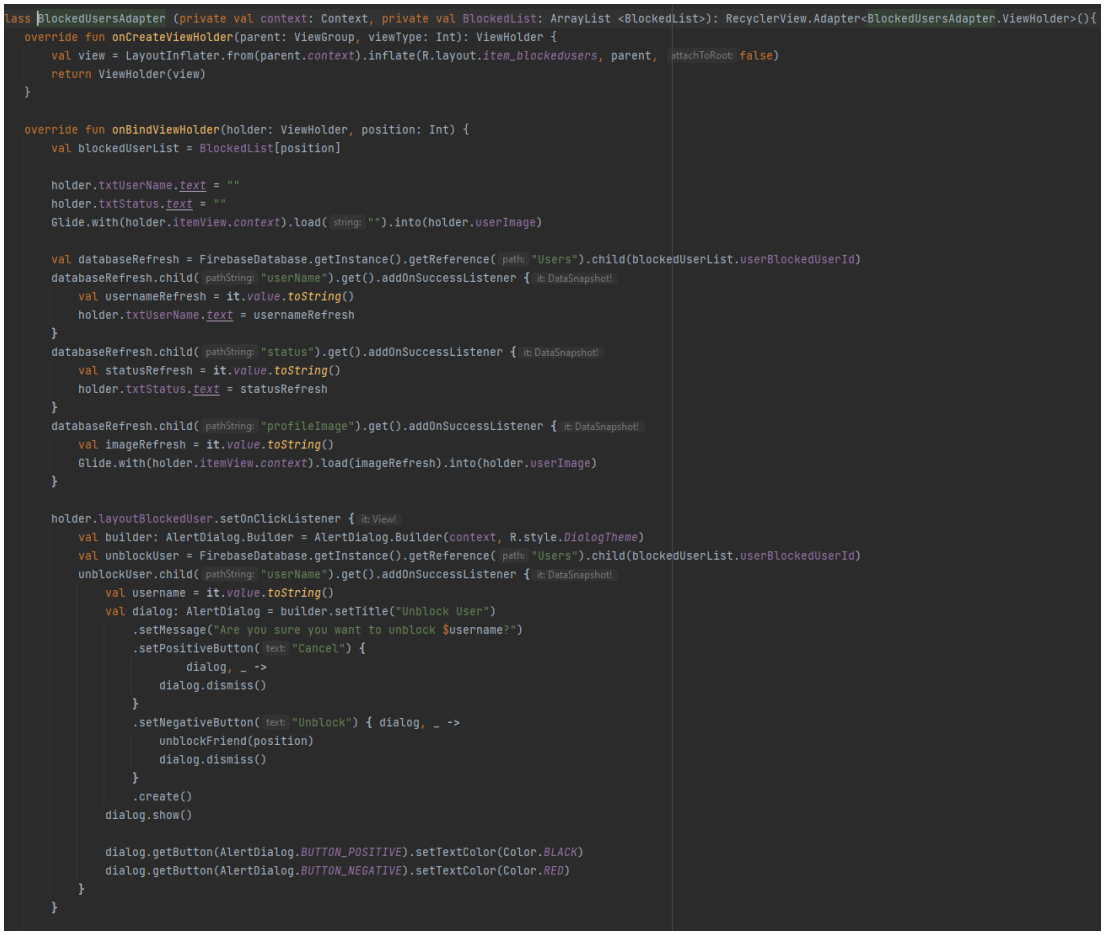


Figure 95 Blocked Users

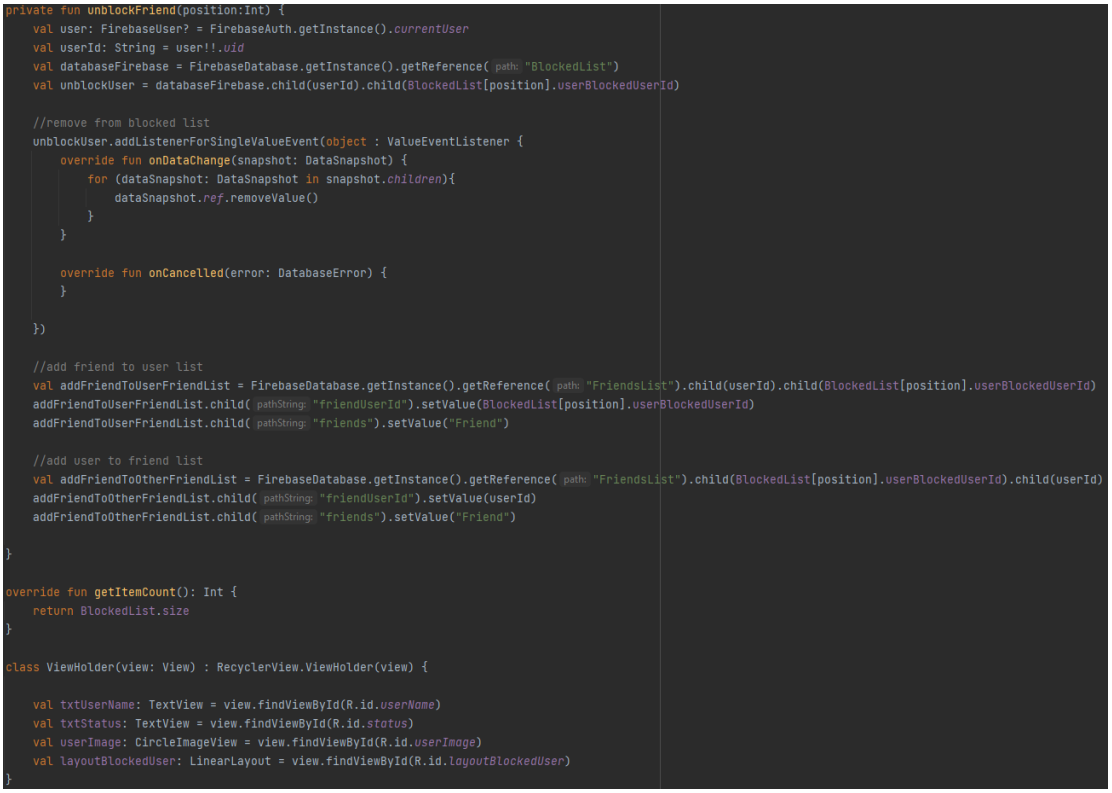
```
    blockedUsersRecyclerView?.layoutManager = LinearLayoutManager( context: this, RecyclerView.VERTICAL,  reverseLayout: false)

    getBlockedList()
}

private fun getBlockedList() {
    val user: FirebaseUser? = auth.currentUser
    val userId: String = user!!.uid

    databaseReference = FirebaseDatabase.getInstance().getReference( path: "BlockedList").child(userId)

    databaseReference.addValueEventListener(object : ValueEventListener {
        override fun onCancelled(error: DatabaseError) {
            Toast.makeText(applicationContext, error.message, Toast.LENGTH_SHORT).show()
        }

        override fun onDataChange(snapshot: DataSnapshot) {
            blockedList.clear()

            for (dataSnapShot: DataSnapshot in snapshot.children) {
                val blocked = dataSnapShot.getValue(BlockedList::class.java)
                Log.d( tag: "tag12", blocked.toString())
                blockedList.add(blocked!!)
            }

            val friendsListAdapter = BlockedUsersAdapter( context: this@BlockedListActivity, blockedList)
            Log.d( tag: "tag13", friendsListAdapter.toString())
            blockedUsersRecyclerView?.adapter = friendsListAdapter
            Log.d( tag: "tag22", blockedUsersRecyclerView.toString())
        }
    })
}
```

Figure 96 List of Blocked Users Code



```
lass BlockedUsersAdapter (private val context: Context, private val BlockedList: ArrayList <BlockedList>): RecyclerView.Adapter<BlockedUsersAdapter.ViewHolder>(){
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {
        val view = LayoutInflater.from(parent.context).inflate(R.layout.item_blockedusers, parent,  attachToRoot: false)
        return ViewHolder(view)
    }

    override fun onBindViewHolder(holder: ViewHolder, position: Int) {
        val blockedUserList = BlockedList[position]

        holder.txtUserName.text = ""
        holder.txtStatus.text = ""
        Glide.with(holder.itemView.context).load( string: "").into(holder.userImage)

        val databaseRefresh = FirebaseDatabase.getInstance().getReference( path: "Users").child(blockedUserList.userBlockedUserId)
        databaseRefresh.child( pathString: "userName").get().addOnSuccessListener { it: DataSnapshot!
            val usernameRefresh = it.value.toString()
            holder.txtUserName.text = usernameRefresh
        }
        databaseRefresh.child( pathString: "status").get().addOnSuccessListener { it: DataSnapshot!
            val statusRefresh = it.value.toString()
            holder.txtStatus.text = statusRefresh
        }
        databaseRefresh.child( pathString: "profileImage").get().addOnSuccessListener { it: DataSnapshot!
            val imageRefresh = it.value.toString()
            Glide.with(holder.itemView.context).load(imageRefresh).into(holder.userImage)
        }

        holder.layoutBlockedUser.setOnClickListener { it: View!
            val builder: AlertDialog.Builder = AlertDialog.Builder(context, R.style.DialogTheme)
            val unblockUser = FirebaseDatabase.getInstance().getReference( path: "Users").child(blockedUserList.userBlockedUserId)
            unblockUser.child( pathString: "userName").get().addOnSuccessListener { it: DataSnapshot!
                val username = it.value.toString()
                val dialog: AlertDialog = builder.setTitle("Unblock User")
                    .setMessage("Are you sure you want to unblock $username?")
                    .setPositiveButton( text: "Cancel") {
                            dialog, _ ->
                        dialog.dismiss()
                    }
                    .setNegativeButton( text: "Unblock") { dialog, _ ->
                        unblockFriend(position)
                        dialog.dismiss()
                    }
                    .create()
                dialog.show()

                dialog.getButton(AlertDialog.BUTTON_POSITIVE).setTextColor(Color.BLACK)
                dialog.getButton(AlertDialog.BUTTON_NEGATIVE).setTextColor(Color.RED)
            }
        }
```

Figure 97 List of Blocked Users Recycler View Code



```
private fun unblockFriend(position:Int) {
    val user: FirebaseUser? = FirebaseAuth.getInstance().currentUser
    val userId: String = user!!.uid
    val databaseFirebase = FirebaseDatabase.getInstance().getReference( path: "BlockedList")
    val unblockUser = databaseFirebase.child(userId).child(BlockedList[position].userBlockedUserId)

    //remove from blocked list
    unblockUser.addListenerForSingleValueEvent(object : ValueEventListener {
        override fun onDataChange(snapshot: DataSnapshot) {
            for (dataSnapshot: DataSnapshot in snapshot.children){
                dataSnapshot.ref.removeValue()
            }
        }

        override fun onCancelled(error: DatabaseError) {
        }

    })

    //add friend to user list
    val addFriendToUserFriendList = FirebaseDatabase.getInstance().getReference( path: "FriendsList").child(userId).child(BlockedList[position].userBlockedUserId)
    addFriendToUserFriendList.child( pathString: "friendUserId").setValue(BlockedList[position].userBlockedUserId)
    addFriendToUserFriendList.child( pathString: "friends").setValue("Friend")

    //add user to friend list
    val addFriendToOtherFriendList = FirebaseDatabase.getInstance().getReference( path: "FriendsList").child(BlockedList[position].userBlockedUserId).child(userId)
    addFriendToOtherFriendList.child( pathString: "friendUserId").setValue(userId)
    addFriendToOtherFriendList.child( pathString: "friends").setValue("Friend")

}

override fun getItemCount(): Int {
    return BlockedList.size
}

class ViewHolder(view: View) : RecyclerView.ViewHolder(view) {

    val txtUserName: TextView = view.findViewById(R.id.userName)
    val txtStatus: TextView = view.findViewById(R.id.status)
    val userImage: CircleImageView = view.findViewById(R.id.userImage)
    val layoutBlockedUser: LinearLayout = view.findViewById(R.id.layoutBlockedUser)
}
```

Figure 98 Unblock User Code

Figure 95 Blocked Users illustrates if 1 user is blocked the other user cannot see themselves being blocked, however the friends list reflects and they're no longer in each other's friends list. The use of Recycler View to in a linear layout was chosen over the grid layout for better UX design, this can be viewed in Figure 96 List of Blocked Users Code.

The recycler view uses an adapter along with a view holder to hold each blocked user iterated in the list. Figure 97 shows how the information of the blocked users profile image, username and status is displayed.

To unblock the user, it can be done using the onClickListener function, when the user selects the specific user, it will open the Alert Dialog, the embedded function inside unblockFriend() can then be evaluated in Figure 98 Unblock User Code, where the system removes the user who is in the blocked list and adds them back to the friends list the data structure and path.

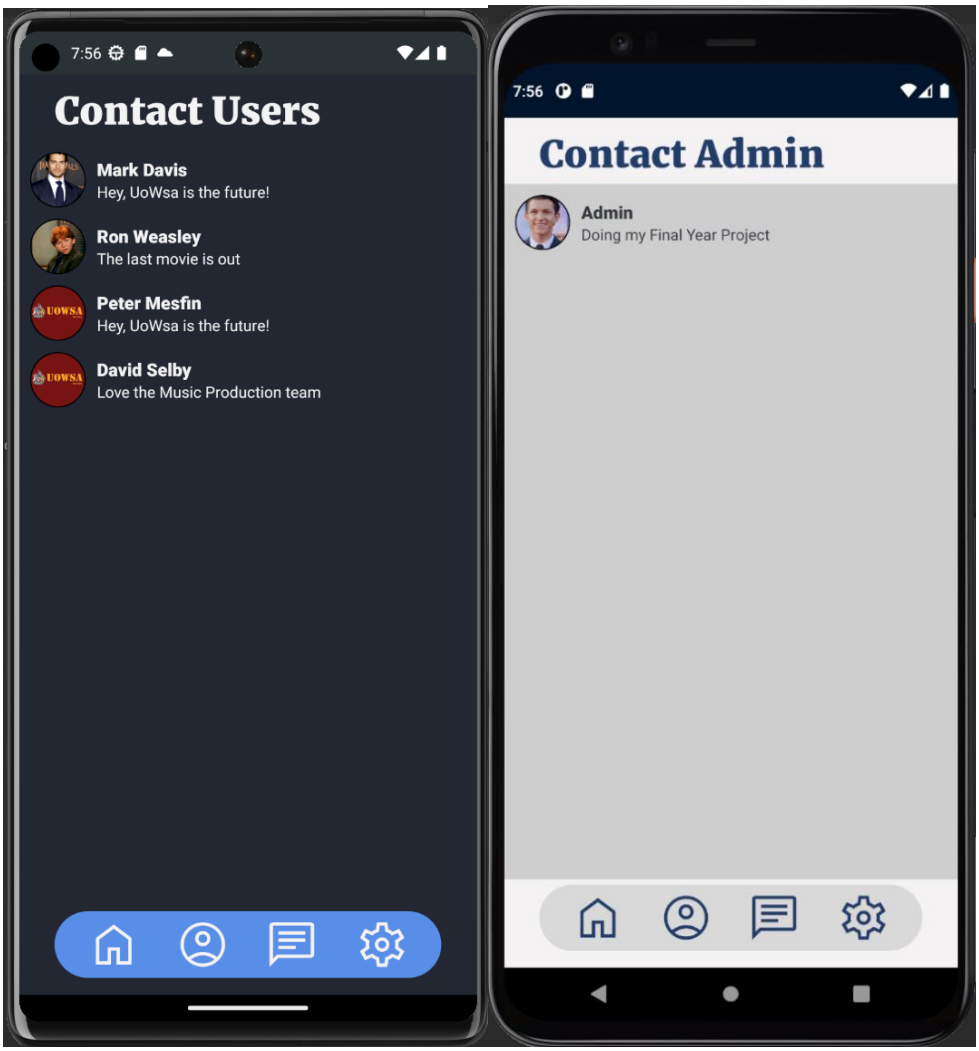


Figure 99 Contact Admin / User

```
contactAdminRecyclerView = findViewById(R.id.contactAdminRecyclerView)
contactAdminRecyclerView?.layoutManager = LinearLayoutManager( context: this, RecyclerView.VERTICAL,  reverseLayout: false)

val contactAdmin = findViewById<TextView>(R.id.contactAdmin)

val user: FirebaseUser? = auth.currentUser
val userId: String = user!!.uid
val userPath = FirebaseDatabase.getInstance().getReference( path: "Users").child(userId).child( pathString: "userName")
userPath.get().addOnSuccessListener { it: DataSnapshot!
    val name = it.value.toString()
    val nameLower = name.lowercase()
    Log.d( tag: "tag", name)
    Log.d( tag: "taglower", nameLower)
    if (nameLower == "admin"){
        contactAdmin.setText("Contact Users")
        contactAdmin.visibility = View.VISIBLE
    }else{
        contactAdmin.setText("Contact Admin")
        contactAdmin.visibility = View.VISIBLE
    }
}

verifyUser()
```

Figure 100 Contact Admin / User Code

```
private fun verifyUser() {
    val user: FirebaseUser? = auth.currentUser
    val userId: String = user!!.uid

    val databaseName = FirebaseDatabase.getInstance().getReference( path: "Users").child(userId).child( pathString: "userName")
    databaseName.addListenerForSingleValueEvent(object : ValueEventListener{
        override fun onDataChange(snapshot: DataSnapshot) {
            val name = snapshot.value.toString()
            val nameLower = name.lowercase()
            if (nameLower == "admin"){
                getUserList()
            }else{
                getAdminList()
            }
        }

        override fun onCancelled(error: DatabaseError) {
        }

    })
}
```

Figure 101 Contact Admin / User Identity Check Code

Figure 99 illustrates the how if the user is an admin, it will show contact user as a title, alternatively if the user is a standard user, it will show contact admin. Figure 100 Contact Admin / User Code takes the current user's username and verify if they're an allocated admin. Username admin is not available to any user and cannot be changed in the update profile section. In order to verify this, Figure 101 Contact Admin / User Identity Check Code illustrates if the user is a verified admin with their username. It doesn't matter if the username is random with uppercase or lower case to avoid detection, because the process takes the username in lower case and if it equals to "admin" in lowercase, their identity can be verified as an admin on the platform.

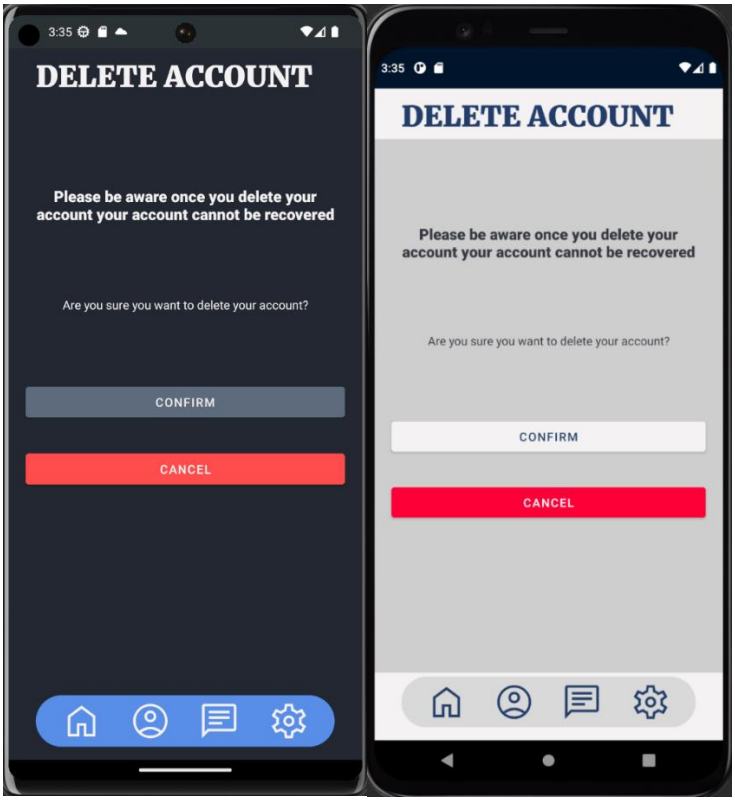


Figure 102 Delete Account

```
private fun deleteUoWsaAccount() {
val user = Firebase.auth.currentUser!!
user.delete().addOnCompleteListener { task ->
    if (task.isSuccessful) {
        Log.d( tag: "TAG",  msg: "User account deleted.")
    }
}

val userStorage = auth.currentUser?.uid.toString()
val picName = "img"
val filename = userStorage + picName
val storageRef = FirebaseStorage.getInstance().getReference( location: "images").child(FirebaseAuth.getInstance().currentUser!!.uid).child(filename)

storageRef.delete().addOnSuccessListener { it: Void!
    Log.d( tag: "tag",  msg: "User Storage Deleted")
}.addOnFailureListener { it: Exception
    Log.d( tag: "tag",  msg: "Unable to delete user storage")
}

val userId: String = user.uid
databaseReference = FirebaseDatabase.getInstance().getReference( path: "Users").child(userId)
databaseReference.removeValue()

val intentMain = Intent( packageContext: this, MainActivity::class.java)
Toast.makeText( context: this@DeleteAccountActivity,  text: "Your account has now been closed", Toast.LENGTH_SHORT).show()
intentMain.flags = Intent.FLAG_ACTIVITY_NEW_TASK or Intent.FLAG_ACTIVITY_CLEAR_TASK
startActivity(intentMain)
finish()
```

Figure 103 Delete Account Code

Figure 102 Delete account illustrates how the account is deleted with two buttons confirm and cancel. If the user selects the confirm button, Figure 103 Delete Account Code shows the deleteUoWsaAccount() function process. It starts with removing the values associated with the user account from Firebase storage, where the users profile image is stored. It then removes all the values from the user in the database. This will then direct the user to the account start up page and the user cannot use the back button to go back to the delete account page to assure there are no legal, ethical, and social issues.

6.3.5 Implementation – Friends

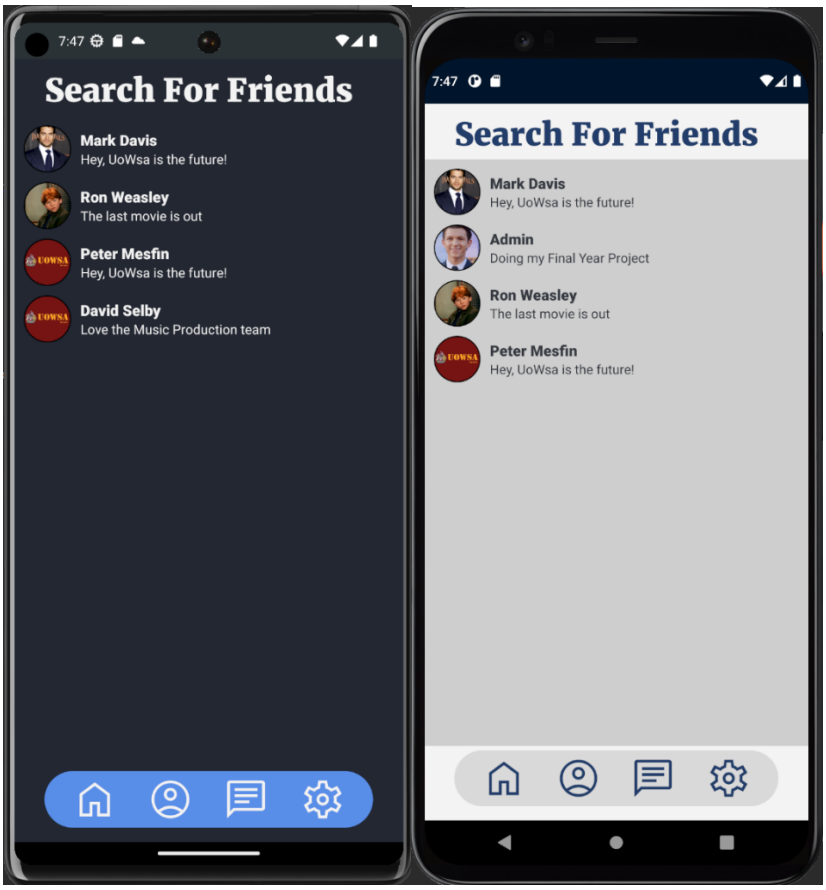


Figure 104 Search for Friends

```
private fun getUserList() {
    val user: FirebaseUser? = auth.currentUser
    val userId: String = user!!.uid

    databaseReference = FirebaseDatabase.getInstance().getReference( path: "Users")

    databaseReference.addValueEventListener(object : ValueEventListener {
        override fun onCancelled(error: DatabaseError) {
            Toast.makeText(applicationContext, error.message, Toast.LENGTH_SHORT).show()
        }

        override fun onDataChange(snapshot: DataSnapshot) {
            userList.clear()

            for (dataSnapShot: DataSnapshot in snapshot.children) {
                val users = dataSnapShot.getValue(User::class.java)
                if (users!!.userId != userId) {
                    userList.add(users)
                }
            }

            val userAdapter = UserAdapter( context: this@SearchFriendsActivity, userList)

            searchForFriendsRecyclerView?.adapter = userAdapter
        }

    })
}
```

Figure 105 Users List Code

```
class UserAdapter (private val context: Context, private val userList: ArrayList <User>) : RecyclerView.Adapter<UserAdapter.ViewHolder>(){
    override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {
        val view = LayoutInflater.from(parent.context).inflate(R.layout.item_user, parent, attachToRoot: false)
        return ViewHolder(view)
    }

    override fun onBindViewHolder(holder: ViewHolder, position: Int) {
        val user = userList[position]
        holder.txtUserName.text = user.userName
        holder.txtStatus.text = user.status
        Glide.with(holder.itemView.context).load(user.profileImage).into(holder.userImage)

        holder.layoutUser.setOnClickListener { it: View!
            val intent = Intent(context, OtherUserActivity::class.java)
            intent.putExtra( name: "userId",user.userId)
            intent.putExtra( name: "userName",user.userName)
            intent.putExtra( name: "profileImage", user.profileImage)
            intent.putExtra( name: "status", user.status)
            intent.putExtra( name: "studyCourse", user.studyCourse)
            intent.putExtra( name: "campusLocation", user.campusLocation)
            context.startActivity(intent)
        }
    }

    override fun getItemCount(): Int {
        return userList.size
    }

    class ViewHolder(view: View) : RecyclerView.ViewHolder(view) {

        val txtUserName: TextView = view.findViewById(R.id.userName)
        val txtStatus: TextView = view.findViewById(R.id.status)
        val userImage: CircleImageView = view.findViewById(R.id.userImage)
        val layoutUser: LinearLayout = view.findViewById(R.id.layoutUser)
    }
```

Figure 106 Users List Adapter Code

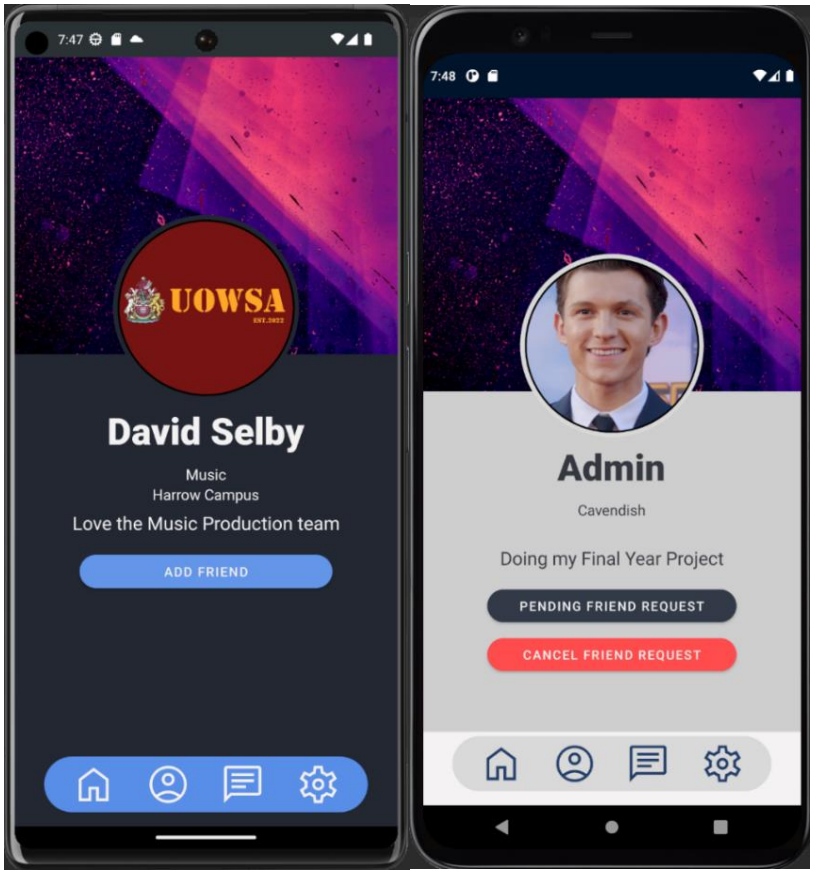


Figure 107 Add Friend

When the user selects the friends' button from the home page, the user will be directed to the friends' page where the user can search for friends illustrated in Figure 104 Search for friends. In Figure 105 Users List code uses a loop and conditional statement to ensure it gathers the whole list of users that exists on the platform; however, it also includes every user except the current user in the conditional statement. The list will then require an adapter and recycler view to show the list on the device. The list of intent to be transferred to the target intent to add friend is illustrated in Figure 106 Users List Adapter code. This list of userId, username, profileImage, status, studyCourse and campusLocation will then be transferred to the next activity when the user selects the specific user using the onClickListener() function illustrated in Figure 107 Add Friend.

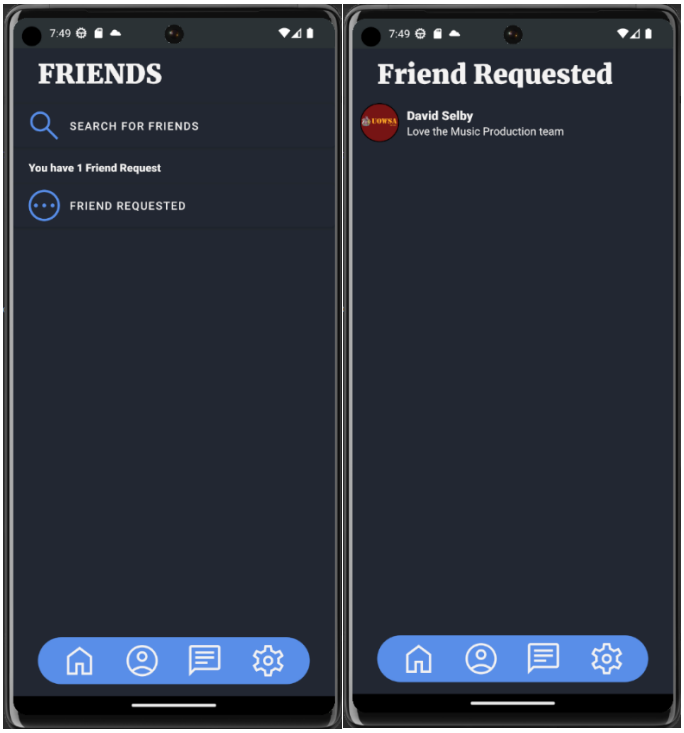


Figure 108 Friend Request

```
private fun friendRequestCounter(){
    val user: FirebaseUser? = auth.currentUser
    val userId: String = user!!.uid
    val counter = FirebaseDatabase.getInstance().getReference( path: "FriendRequest").child(userId)
    counter.addValueEventListener(object : ValueEventListener{
        @SuppressLint("SetTextI18n")
        override fun onDataChange(snapshot: DataSnapshot) {
            val count = snapshot.childrenCount
            Log.d( tag: "tagcount", count.toString())
            if (count < 2){
                val countFR = count.toString()
                Log.d( tag: "tagcount2", countFR)
                friendRequestCount?.setText("You have $countFR Friend Request")
            }else{
                val countFRs = count.toString()
                Log.d( tag: "tagcount1", countFRs)
                friendRequestCount?.setText("You have $countFRs Friend Requests")
            }
        }

        override fun onCancelled(error: DatabaseError) {
        }

    })
}
```

Figure 109 Friend Request Counter Code

```
private fun getRequestedFriendsList() {
    val user: FirebaseUser? = auth.currentUser
    val userId: String = user!!.uid

    databaseReference = FirebaseDatabase.getInstance().getReference( path: "FriendRequest").child(userId)

    databaseReference.addValueEventListener(object : ValueEventListener {
        override fun onCancelled(error: DatabaseError) {
            Toast.makeText(applicationContext, error.message, Toast.LENGTH_SHORT).show()
        }

        override fun onDataChange(snapshot: DataSnapshot) {
            friendRequestList.clear()

            for (dataSnapShot: DataSnapshot in snapshot.children) {
                val friendRequest = dataSnapShot.getValue(FriendRequest::class.java)
                Log.d( tag: "tag12", friendRequest.toString())
                friendRequestList.add(friendRequest!!)
            }

            val friendRequestAdapter = FriendRequestAdapter( context: this@FriendRequestActivity, friendRequestList)
            Log.d( tag: "tag15", friendRequestAdapter.toString())
            friendRequestRecyclerView?.adapter = friendRequestAdapter
            Log.d( tag: "tag16", friendRequestRecyclerView.toString())
        }
    })
}
```

Figure 110 Friend Request Code

Figure 107 illustrates how a user can add a friend and when the user can add a user and when that action is complete the user can see the friend request is pending or they can cancel the request. On the other user's device, the user can view the friend request counter in Figure 108 Friend Request have 1 friend requested and Figure 109 Friend Request Counter Code runs through the database to find how many existing friends request the user has using the childrenCount function. This follows with a conditional statement if the count is less than two the text field shown on the device will be "You have 1 Friend Request", on the other hand if there are more than 2 it will show "You have (exact number of friend request) Friend Requests". The friend list will then use the recycler view adapter indicated in Figure 110 Fiend Request Code to view the list of friend request.

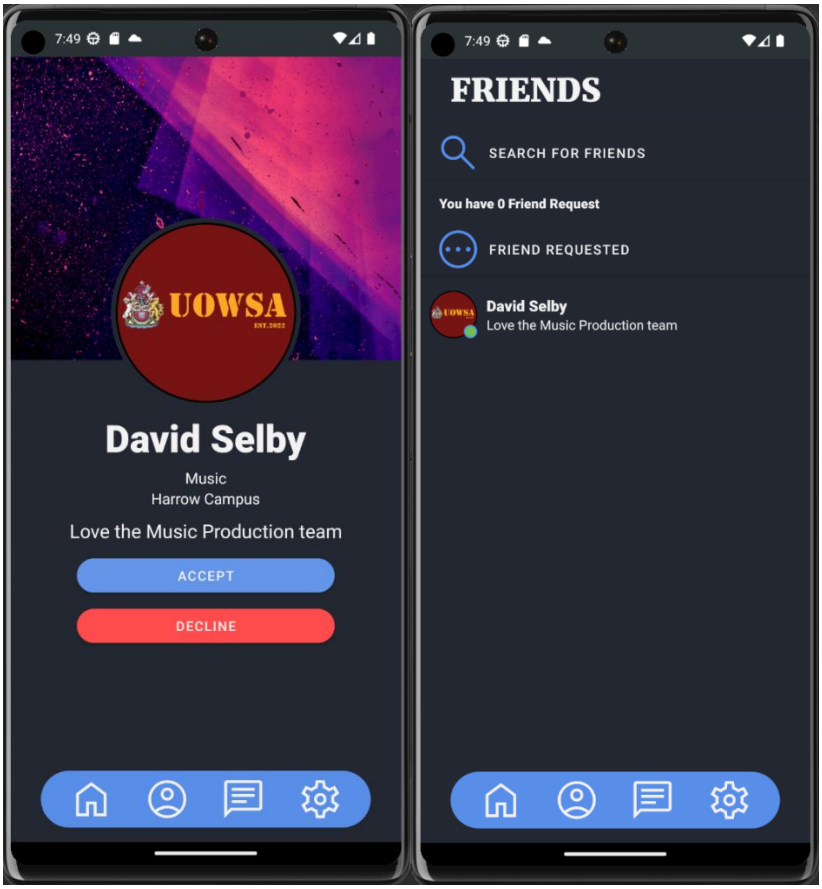


Figure 111 6.63 Accept / Decline Friend Request

```
private fun friendRequestAccepted(){
    val user: FirebaseUser? = auth.currentUser
    val userId: String = user!!.uid
    val friendRequestAccepted = FirebaseDatabase.getInstance().getReference( path: "FriendRequest").child(userId).child(senderUserId!!).child( pathString: "requestAccepted")

    friendRequestAccepted.setValue("Accepted").addOnCompleteListener(this) { it: Task<Void!> ->
        if (it.isSuccessful){
            Toast.makeText( context: this,  text: "Friend Added Successfully", Toast.LENGTH_SHORT).show()
            val acceptedRequest = FirebaseDatabase.getInstance().getReference( path: "FriendRequest").child(userId).child(senderUserId!!).child( pathString: "requestAccepted")
            acceptedRequest.addListenerForSingleValueEvent(object : ValueEventListener{
                override fun onCancelled(error: DatabaseError) {

                }
                override fun onDataChange(snapshot: DataSnapshot) {
                    val accepted = snapshot.getValue(String::class.java).toString()
                    if (accepted == "Accepted"){

                        databaseReference = FirebaseDatabase.getInstance().getReference( path: "FriendRequest").child(userId).child(senderUserId!!)
                        databaseReference.removeValue()
                    }
                }
            })
            val intentFriendRequest = Intent( packageContext: this, FriendsActivity::class.java)
            intentFriendRequest.flags = Intent.FLAG_ACTIVITY_NEW_TASK or Intent.FLAG_ACTIVITY_CLEAR_TASK
            startActivity(intentFriendRequest)
            finish()
        }
    }
}

private fun declineFriendList() {
    val user: FirebaseUser? = auth.currentUser
    val userId: String = user!!.uid

    databaseReference = FirebaseDatabase.getInstance().getReference( path: "FriendRequest").child(userId).child(senderUserId!!)
    databaseReference.removeValue()

    val intentFriendRequest = Intent( packageContext: this, FriendRequestActivity::class.java)
    intentFriendRequest.flags = Intent.FLAG_ACTIVITY_NEW_TASK or Intent.FLAG_ACTIVITY_CLEAR_TASK
    startActivity(intentFriendRequest)
    finish()
}
```

Figure 112 Accept / Decline Friend Request Code

```
private fun addToFriendList() {
    val user: FirebaseUser? = auth.currentUser
    val userId: String = user!!.uid

    if(receiverFriendUserId == userId){
        val addFriendToUserFriendList = FirebaseDatabase.getInstance().getReference( path: "FriendsList").child(userId).child(senderUserId!!)

        addFriendToUserFriendList.child( pathString: "friendUserId").setValue(senderUserId!!)

        addFriendToUserFriendList.child( pathString: "friends").setValue("Friend")

        val addFriendToOtherFriendList = FirebaseDatabase.getInstance().getReference( path: "FriendsList").child(senderUserId!!).child(userId)

        addFriendToOtherFriendList.child( pathString: "friendUserId").setValue(userId)

        addFriendToOtherFriendList.child( pathString: "friends").setValue("Friend")

        friendRequestAccepted()

    }
}
```

Figure 113 Add to Friends List Code

Figure 111 Accept / Decline Friend Request illustrates the accept and decline friend request as well as what happens if the user selects the accept request button. Figure 112 Accept / Decline Friend Request shows the differences of accepting and declining a friend request with the acceptFriendRequest() and declineFriendRequest() function. When the user accepts a friend request it deletes the friend request and add the user to friends list with all the values required using the addToFriendList() function in Figure 113 Add to Friends List Code. When the user decides to decline the friend request, the friend request values on the database will just be deleted.

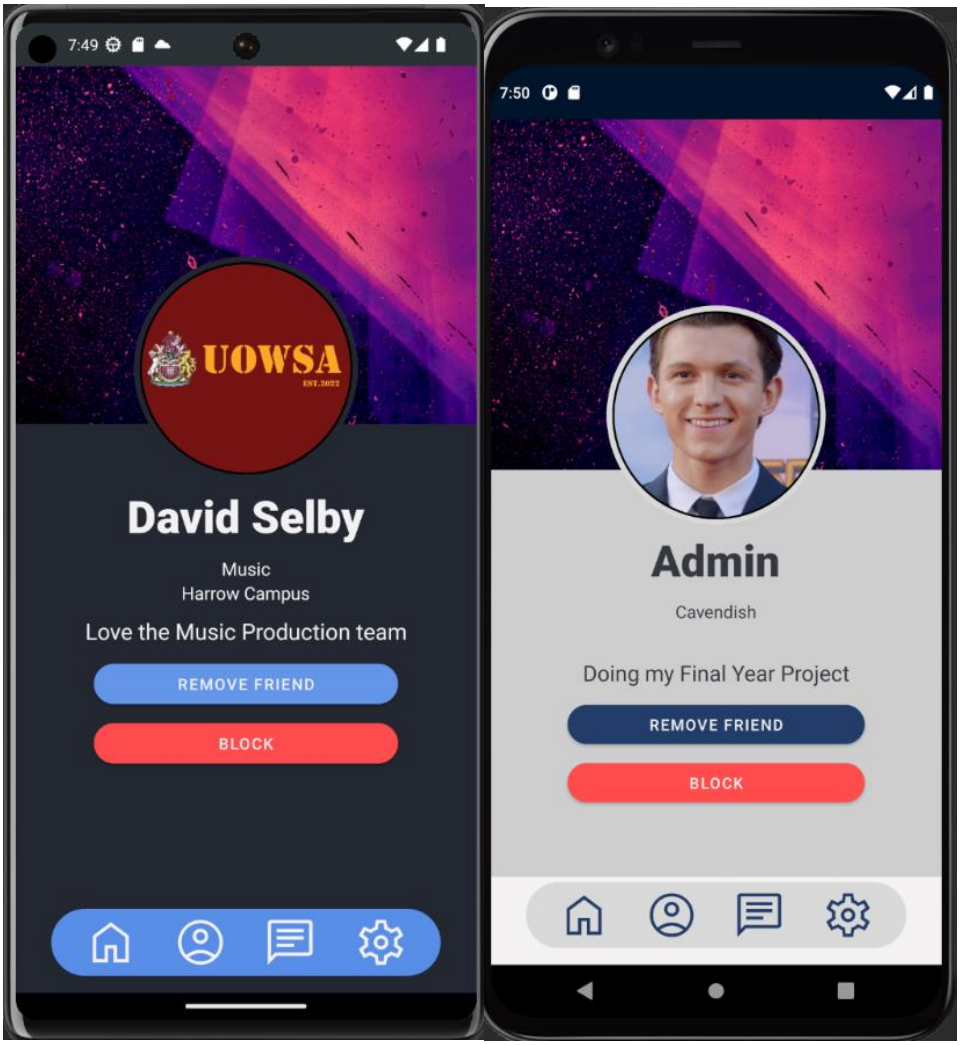


Figure 114 Remove or Block Friend

```
removeFriendBtn.setOnClickListener { it: View!
    val builder: AlertDialog.Builder = AlertDialog.Builder( context: this, R.style.DialogTheme)
    databaseRefresh.child( pathString: "userName").get().addOnSuccessListener{ it: DataSnapshot!
        val usernameRefresh = it.value.toString()
        val dialog: AlertDialog = builder.setTitle("Remove Friend")
            .setMessage("Are you sure you want to remove $usernameRefresh from your friends?")
            .setPositiveButton( text: "Cancel") {
                    dialog, _ ->
                dialog.dismiss()
            }
            .setNegativeButton( text: "Remove") { dialog, _ ->
                removeFriend()
                dialog.dismiss()
                val friendsIntent = Intent( packageContext: this, FriendsActivity::class.java)
                friendsIntent.flags = Intent.FLAG_ACTIVITY_NEW_TASK or Intent.FLAG_ACTIVITY_CLEAR_TASK
                startActivity(friendsIntent)
                finish()
            }
            .create()
        dialog.show()

        dialog.getButton(AlertDialog.BUTTON_POSITIVE).setTextColor(Color.BLACK)
        dialog.getButton(AlertDialog.BUTTON_NEGATIVE).setTextColor(Color.RED)
    }
}

blockFriendBtn.setOnClickListener { it: View!
    val builder: AlertDialog.Builder = AlertDialog.Builder( context: this, R.style.DialogTheme)
    databaseRefresh.child( pathString: "userName").get().addOnSuccessListener{ it: DataSnapshot!
        val usernameRefresh = it.value.toString()
        val dialog: AlertDialog = builder.setTitle("Block Friend")
            .setMessage("Are you sure you want to block $usernameRefresh?")
            .setPositiveButton( text: "Cancel") {
                    dialog, _ ->
                dialog.dismiss()
            }
            .setNegativeButton( text: "Block") { dialog, _ ->
                blockFriend()
                dialog.dismiss()
                val friendsIntent = Intent( packageContext: this, FriendsActivity::class.java)
                friendsIntent.flags = Intent.FLAG_ACTIVITY_NEW_TASK or Intent.FLAG_ACTIVITY_CLEAR_TASK
                startActivity(friendsIntent)
                finish()
            }
            .create()
        dialog.show()

        dialog.getButton(AlertDialog.BUTTON_POSITIVE).setTextColor(Color.BLACK)
        dialog.getButton(AlertDialog.BUTTON_NEGATIVE).setTextColor(Color.RED)
    }
}
```

Figure 115 Remove or Block Friend Code

In Figure 114 Remove or Block Friend, when the user selects a specific user from the friends list, they will be directed to the user profile of the user they've selected with two buttons including remove friend and block. The code for both of these buttons is illustrated in Figure 115 Remove or Block Friend Code, both of the buttons will show a dialog box indicating if they would like to remove or block. Removing or blocking will remove the other user from the friends list in the database, however the blocking feature creates a new blocking list which is illustrated in Figure 90 Blocked Users.

6.3.6 Implementation – Direct Messaging

The direct messaging section of the application has two options – “Contact Administrator” and “UoW Message”. Users can open a message box by selecting either of these options, which will expand the keyboard for them to type their message. After composing their message, users can click the send button on the right-hand side to send the message. Users can also view and delete specific messages by selecting them. Additionally, a delete all button is location at the top-right corner of the page. If the user selects this button, a dialog box will appear, explaining that all sent messages will be deleted. The dialog box will contain two buttons – “Delete” and “Cancel”.

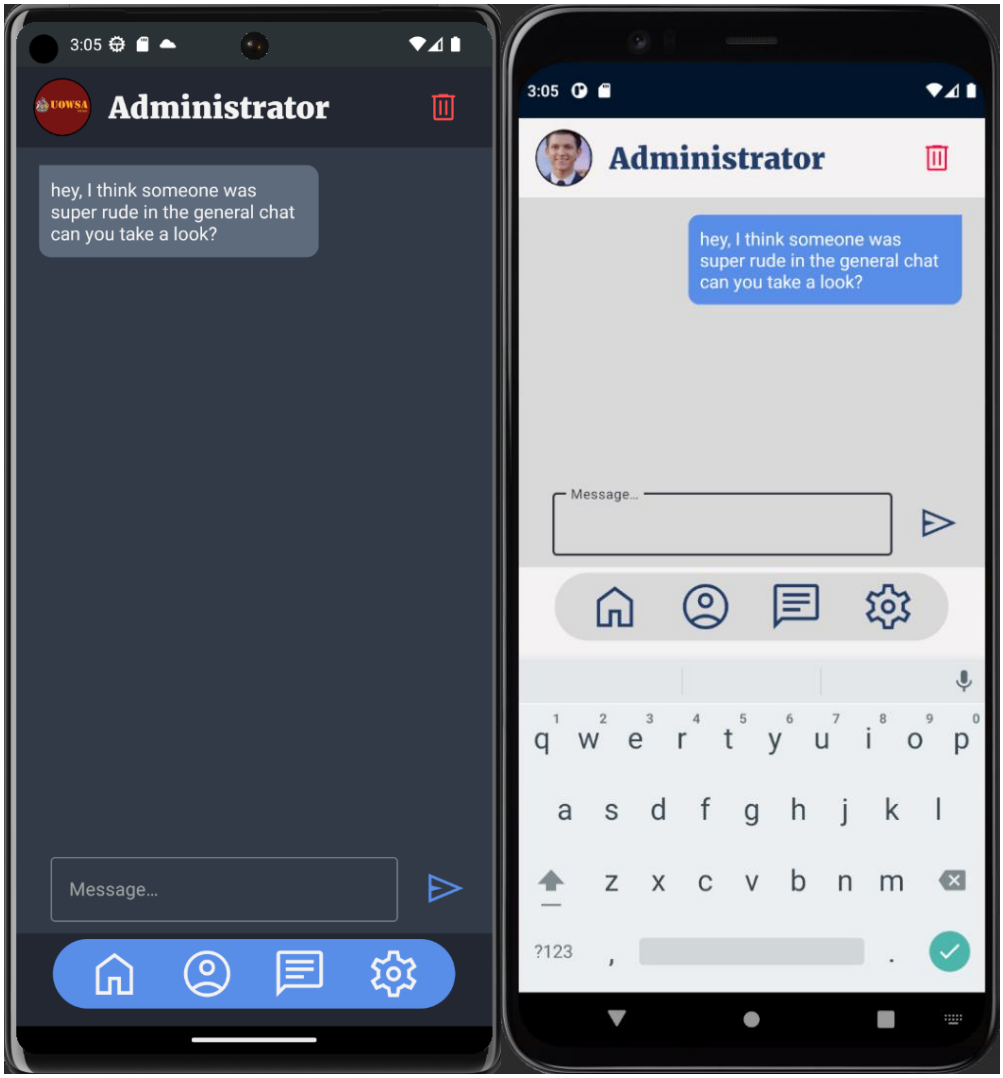


Figure 116 Administrator Messaging

```
val layoutManager = LinearLayoutManager( context: this)
layoutManager.reverseLayout = true
adminMessageRecyclerView?.layoutManager = layoutManager
adminMessageRecyclerView?.layoutManager = LinearLayoutManager( context: this, RecyclerView.VERTICAL,  reverseLayout: false)

adminUserId = intent.getStringExtra( name: "userId").toString()

  intent.putExtra("userId",user.userId)
  intent.putExtra("userName",user.userName)
  intent.putExtra("profileImage", user.profileImage)
  intent.putExtra("status", user.status)
  intent.putExtra("studyCourse", user.studyCourse)
  intent.putExtra("campusLocation", user.campusLocation)

readMessage()

deleteAllMessagesBtn.setOnClickListener { it: View!
    val builder: AlertDialog.Builder = AlertDialog.Builder( context: this, R.style.DialogTheme)
    val dialog: AlertDialog = builder.setTitle("Delete All Your Messages")
        .setMessage("Are you sure you want to delete all of your messages?")
        .setPositiveButton( text: "Cancel") { dialog, _ ->
            dialog.dismiss()
        }
        .setNegativeButton( text: "Delete") { dialog, _ ->
            deleteAllMessages()
            dialog.dismiss()
        }
        .create()
    dialog.show()

    dialog.getButton(AlertDialog.BUTTON_POSITIVE).setTextColor(Color.BLACK)
    dialog.getButton(AlertDialog.BUTTON_NEGATIVE).setTextColor(Color.RED)

}

val databaseUserInfo = FirebaseDatabase.getInstance().getReference( path: "Users").child(adminUserId!!)
databaseUserInfo.child( pathString: "profileImage").get().addOnSuccessListener { it: DataSnapshot!
    val imageGet = it.value.toString()
    Glide.with( activity: this).load(imageGet).into(friendImage)
}

adminMessageRecyclerView?.scrollToPosition( position: adminMessageList.size-1)

sendBtn.setOnClickListener { it: View!
    Log.d( tag: "tagtext", messageTxt.toString())
    val message = messageTxt.toString()
    Log.d( tag: "tagtext1", message)
    if(messageTxt.isEmpty()){
        Toast.makeText(applicationContext,  text: "Message is empty", Toast.LENGTH_SHORT).show()
    }else{
        sendMessage(message)
        clearMessage.text.clear()
    }
}
```

Figure 117 Administrator Messaging Code

```
private fun deleteAllMessages() {
    val user: FirebaseUser? = FirebaseAuth.getInstance().currentUser
    val userId: String = user!!.uid

    val userRef = FirebaseDatabase.getInstance().getReference( path: "contactAdmin").child(userId).child(adminUserId!!)
    val friendRef = FirebaseDatabase.getInstance().getReference( path: "contactAdmin").child(adminUserId!!).child(userId)

    val userMessageDelete = userRef.orderByChild( path: "senderAdminMessageId").equalTo(userId)
    userMessageDelete.addListenerForSingleValueEvent(object: ValueEventListener {
        override fun onDataChange(snapshot: DataSnapshot) {
            for (dataSnapshot01: DataSnapshot in snapshot.children){
                dataSnapshot01.ref.removeValue()
            }
        }

        override fun onCancelled(error: DatabaseError) {
        }

    })

    val userFriendNodeMessageDelete = friendRef.orderByChild( path: "senderAdminMessageId").equalTo(userId)
    userFriendNodeMessageDelete.addListenerForSingleValueEvent(object : ValueEventListener {
        override fun onDataChange(snapshot: DataSnapshot) {
            for (dataSnapshot02: DataSnapshot in snapshot.children){
                dataSnapshot02.ref.removeValue()
            }
        }

        override fun onCancelled(error: DatabaseError) {
        }

    })
}
```

Figure 118 Administrator Messaging Delete All Messages Code

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:paddingTop="10dp"
    android:id="@+id/layoutUserMessage"
    android:orientation="vertical">

    <LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:orientation="horizontal"
        android:layout_marginEnd="150dp"
        tools:ignore="RtlSymmetry">

        <TextView
            android:id="@+id/message"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:fontFamily="sans-serif"
            android:padding="10dp"
            android:layout_marginStart="20dp"
            android:layout_marginTop="5dp"
            android:textColor="?colorPrimary"
            android:background="@drawable/item_left_background"
            android:textSize="16sp"
            tools:text="testing this information can i see where the length will take me" />


    </LinearLayout>


</LinearLayout>
```

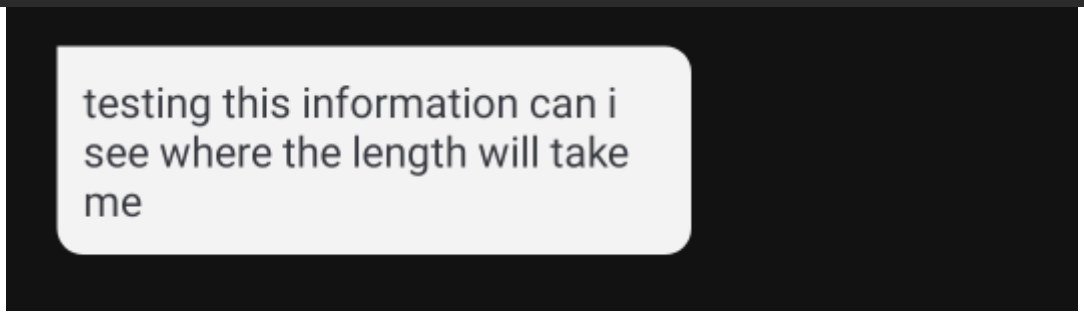


Figure 119 Message Left

```xml
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:paddingTop="10dp"
    android:gravity="end"
    android:layout_marginStart="150dp"
    android:orientation="vertical">

    <LinearLayout
        android:id="@id/layoutUserMessage"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:gravity="end"
        android:orientation="horizontal"
        tools:ignore="RtlSymmetry">

        <TextView
            android:id="@+id/message"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:fontFamily="sans-serif"
            android:padding="10dp"
            android:layout_marginTop="5dp"
            android:layout_marginEnd="20dp"
            android:textColor="@color/white"
            android:background="@drawable/item_right_background"
            android:textSize="16sp"
            tools:text="Testing length of message" />


    </LinearLayout>


</LinearLayout>
```

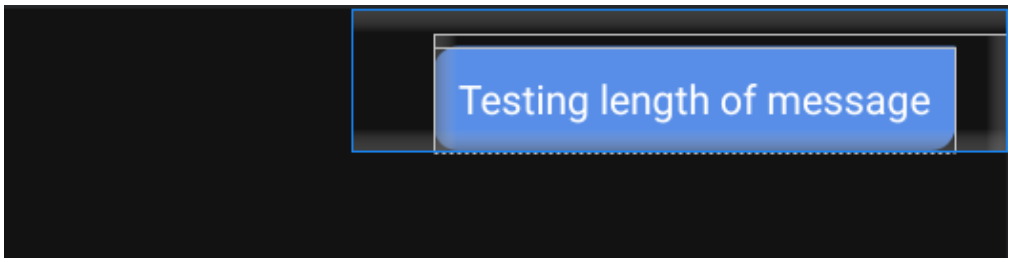


Figure 120 Message Right

```
private val messageLeft = 0
private val messageRight = 1
var firebaseUser: FirebaseUser? = null

override fun onCreateViewHolder(parent: ViewGroup, viewType: Int): ViewHolder {
    return if (viewType == messageRight) {
        val view = LayoutInflater.from(parent.context).inflate(R.layout.item_right, parent, attachToRoot: false)
        ViewHolder(view)
    } else {
        val view = LayoutInflater.from(parent.context).inflate(R.layout.item_left, parent, attachToRoot: false)
        ViewHolder(view)
    }
}

override fun onBindViewHolder(holder: ViewHolder, position: Int) {
    val chat = adminMessageList[position]
    holder.txtMessage.text = chat.message

    if (chat.senderAdminMessageId == firebaseUser!!.uid){
        holder.layoutUserMessage.setOnClickListener { it: View!

            val builder: AlertDialog.Builder = AlertDialog.Builder(context, R.style.DialogTheme)
            val dialog: AlertDialog = builder.setTitle("Delete Message")
                .setMessage("Are you sure you want to delete this message?")
                .setPositiveButton( text: "Cancel") {
                        dialog, _ ->
                    Toast.makeText(context, "you have pressed cancel", Toast.LENGTH_SHORT).show()
                    dialog.dismiss()
                }
                .setNegativeButton( text: "Delete") { dialog, _ ->
                    Toast.makeText(context, "you have pressed delete", Toast.LENGTH_SHORT).show()
                    deleteMessage(position)
                    dialog.dismiss()
                }
                .create()
            dialog.show()

            dialog.getButton(AlertDialog.BUTTON_POSITIVE).setTextColor(Color.BLACK)
            dialog.getButton(AlertDialog.BUTTON_NEGATIVE).setTextColor(Color.RED)

        }
    }
}
```

Figure 121 Administrator Recycler View Adapter Code

In Figure 116 Administrator Messaging, the Direct Messaging area of the application is demonstrated for both Administrators and UoW Message areas, The implementation of this feature is further elaborated in Figure 117 Administrator Messaging Code, where the “readMessage()” function is introduced. This function works the Recycler View adapter to display all of the messages between two users, retrieved from the database. If the message box is empty and the user tries to send the message, a prompt will display the message “Message is empty”.

Figure 118 Administrator Messaging Delete All Messages Code demonstrates the “Delete All Messagees” function, which removes all messages from the database with the sender ID being the original user. The messages are presented in a model as shown in Figure 119 Message Left and Figure 120 Message Right using the XML code provided. The messages are separated using “item_left” and “item_right” positioning in the “onCreateViewHolder” function, as shown in Figure 121.

Users can delete specific messages by selecting the with the setOnClickListener function, which prompts a dialog to delete the selected message using the position of the message from the user model and incorporating this position to the database messaging position.

6.5.7 Implementation – Notifications

The notifications are set up in the database to be read and unread, when the user selects a specific user and automatically scrolled to the bottom of the message, the user sends a message and the message sent to the database is set as unread, this will set a counter for the other user to notify them they have received a new message. However, when the other participant opens the message, all of the messages will be set as read, the counter on the UoW Message page will then reset to invisible.

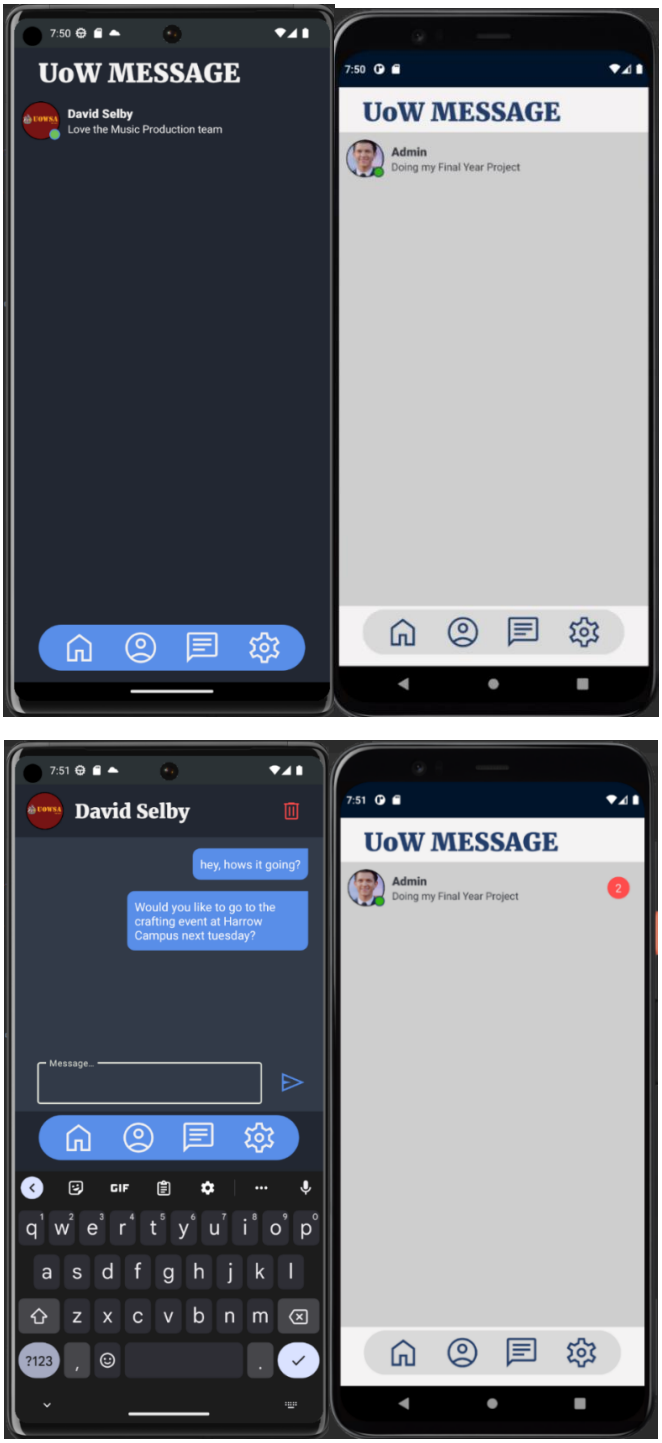


Figure 122 Notifications

```
private fun sendMessage(message: String) {
    val user: FirebaseUser? = auth.currentUser
    val userId: String = user!!.uid

    val sendMessageDatabase = FirebaseDatabase.getInstance().getReference( path: "UoWMessage").child(userId).child(friendUserId!!)

    val formatter = DateTimeFormatter.ofPattern( pattern: "dd-MM-yyyy HH:mm:ss")
    val dateTime = LocalDateTime.now().format(formatter)
    Log.d( tag: "tagdatetime", dateTime.toString())

    val messagingContent: HashMap <String, String> = HashMap()
    messagingContent["senderUoWMessageId"] = userId
    messagingContent["receiverUoWMessageId"] = friendUserId!!
    messagingContent["message"] = message
    messagingContent["messageDateTime"] = dateTime
    messagingContent["senderMessageStatus"] = "read"
    messagingContent["receiverMessageStatus"] = "unread"


    sendMessageDatabase.push().setValue(messagingContent)

    val sendMessageDatabaseFriendUser = FirebaseDatabase.getInstance().getReference( path: "UoWMessage").child(friendUserId!!).child(userId)

    val messagingContentFriendUser: HashMap <String, String> = HashMap()
    messagingContentFriendUser["senderUoWMessageId"] = userId
    messagingContentFriendUser["receiverUoWMessageId"] = friendUserId!!
    messagingContentFriendUser["message"] = message
    messagingContentFriendUser["messageDateTime"] = dateTime
    messagingContentFriendUser["senderMessageStatus"] = "read"
    messagingContentFriendUser["receiverMessageStatus"] = "unread"

    sendMessageDatabaseFriendUser.push().setValue(messagingContentFriendUser)

}
```

Figure 123 Notifications Message Sent Code

```
val sendMessageDatabase = FirebaseDatabase.getInstance().getReference( path: "UoWMessage").child(userId).child(friendMessageList.friendUserId)
var unreadCount :Long
val unreadCounter = sendMessageDatabase.orderByChild( path: "receiverMessageStatus").equalTo( value: "unread")
unreadCounter.addValueEventListener(object: ValueEventListener {
    override fun onDataChange(snapshot: DataSnapshot) {
        unreadCount = snapshot.childrenCount
        if (unreadCount > 0){
            val unreadCountToString = unreadCount.toString()
            holder.unreadMessage.text = unreadCountToString
            holder.unreadMessage.visibility = View.VISIBLE
        }

    }

    override fun onCancelled(error: DatabaseError) {
    }

})
```

Figure 124 Notifications Receive Notifications Code

Figure 122 Notifications, feature the notification counter is demonstrated. If one user sends a message, the other will be notified of the number of messages sent to them. In Figure 123 Notifications Message Sent Code illustrates how notifications are sent in the form of a hashmap with its corresponding key and value. Figure 124 Notifications Receive Notifications Code demonstrates how messages are added to the counter, it is then converted to a string and displayed on the user's device.

When messages are sent, they are marked as "read" for the sender and "unread" for the receiver. When the user opens the message, it notifies the database to indicate that the sent message is now marked as "read".

6.5.8 Implementation – Discussion Board (All Chat / Forums)

The discussion board feature of the application includes the Hangout All Chat, where users can message all user of the platform. This feature is further categorised into three sections: General, Make New Friends, and Student Wellbeing.

The Forums section allows users to create a new forum with a title and description. Once a forum is created, its details will be listed at the top of the Forums messaging section, including information such as the title, description, posted date, and the user who posted it.

All users of the application can enter the created forum and message in that particular page.

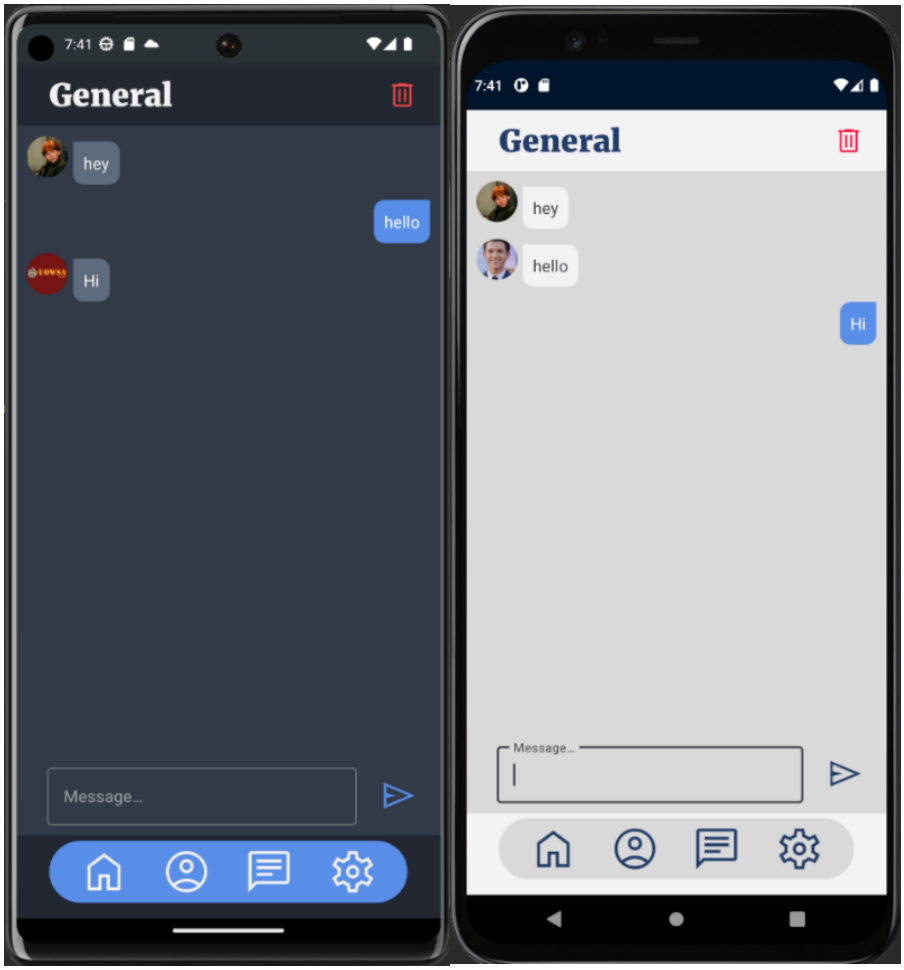
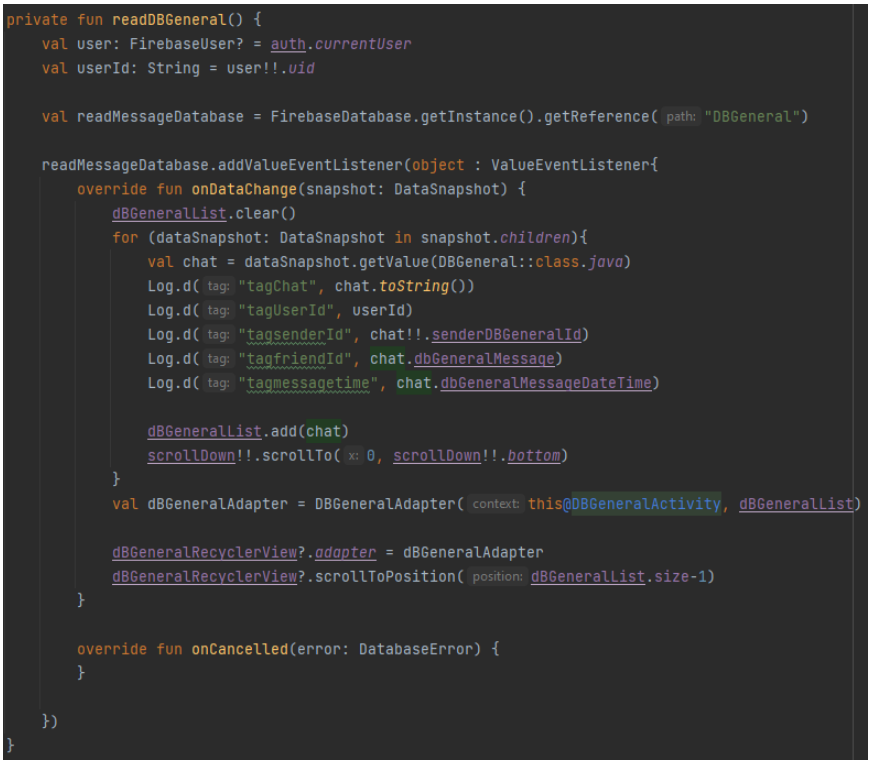


Figure 125 Hangout All Chat



```
private fun readDBGeneral() {
    val user: FirebaseUser? = auth.currentUser
    val userId: String = user!!.uid

    val readMessageDatabase = FirebaseDatabase.getInstance().getReference( path: "DBGeneral")

    readMessageDatabase.addValueEventListener(object : ValueEventListener{
        override fun onDataChange(snapshot: DataSnapshot) {
            dBGeneralList.clear()
            for (dataSnapshot: DataSnapshot in snapshot.children){
                val chat = dataSnapshot.getValue(DBGeneral::class.java)
                Log.d( tag: "tagChat", chat.toString())
                Log.d( tag: "tagUserId", userId)
                Log.d( tag: "tagsenderId", chat!!.senderDBGeneralId)
                Log.d( tag: "tagfriendId", chat.dbGeneralMessage)
                Log.d( tag: "tagmessagetime", chat.dbGeneralMessageDateTime)

                dBGeneralList.add(chat)
                scrollDown!!.scrollTo( x: 0, scrollDown!!.bottom)
            }
            val dBGeneralAdapter = DBGeneralAdapter( context: this@DBGeneralActivity, dBGeneralList)

            dBGeneralRecyclerView?.adapter = dBGeneralAdapter
            dBGeneralRecyclerView?.scrollToPosition( position: dBGeneralList.size-1)
        }

        override fun onCancelled(error: DatabaseError) {
        }

    })
}
```

Figure 126 Hangout All Chat Code

Figure 125 All Chat incorporates with Figure 126 Hangout All Chat Code, shows a list of messages in the “DBGeneral” node in the database. Every time the readDBGeneral() function executes, it clears the local list and replace the list with a new list from the database, this ensures the local list is updated for the user to view new messages.

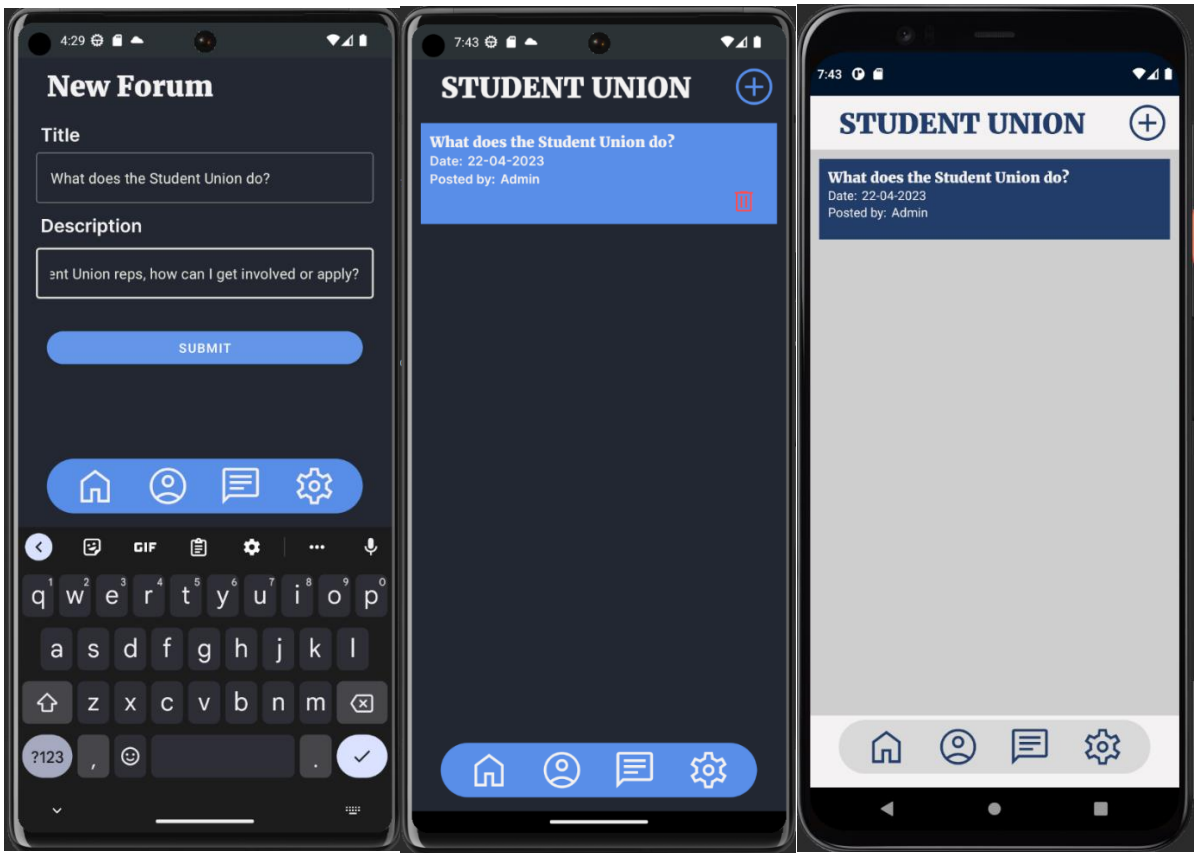


Figure 127 New Forum

```
submitBtn.setOnClickListener { it: View!
    if (titleTxt.text.toString().isEmpty()){
        Toast.makeText( context: this, text: "Title field is empty", Toast.LENGTH_SHORT).show()
    }else if (descriptionTxt.text.toString().isEmpty()){
        Toast.makeText( context: this, text: "The description field is empty", Toast.LENGTH_SHORT).show()
    }else{
        val title = titleTxt.text.toString()
        val description = descriptionTxt.text.toString()
        Log.d( tag: "tag1", title)
        Log.d( tag: "tag1", description)

        createStudentUnionForum(title, description)
        val intentStudy = Intent( packageContext: this@StudentUnionNewForumActivity, ForumStudentUnionActivity::class.java)
        intentStudy.flags = Intent.FLAG_ACTIVITY_NEW_TASK or Intent.FLAG_ACTIVITY_CLEAR_TASK
        startActivity(intentStudy)
        finish()
    }
}
```

Figure 128 New Forum Validation Code

```
private fun createStudentUnionForum(title: String, description: String) {


    Log.d( tag: "tag",  msg: "testing")
    val user: FirebaseUser? = FirebaseAuth.getInstance().currentUser
    val userId: String = user!!.uid
    Log.d( tag: "tag", userId)

    val formatter = DateTimeFormatter.ofPattern( pattern: "dd-MM-yyyy HH:mm:ss")
    val dateTime = LocalDateTime.now().format(formatter)
    val format = DateTimeFormatter.ofPattern( pattern: "dd-MM-yyyy")
    val date = LocalDateTime.now().format(format)
    Log.d( tag: "tag1", userId)

    val databaseRef = FirebaseDatabase.getInstance().getReference( path: "StudentUnionForum")
    Log.d( tag: "tagdatabaseref", databaseRef.toString())

    val newPost : HashMap <String, String> = HashMap()
    newPost["studentUnionCreatorId"] = userId
    newPost["studentUnionTitle"] = title
    newPost["studentUnionDescription"] = description
    newPost["studentUnionDate"] = date
    newPost["studentUnionCreatedDateTime"] = dateTime

    Log.d( tag: "tag1", title)
    Log.d( tag: "tag1", description)
    Log.d( tag: "tag1", date)
    Log.d( tag: "tag1", dateTime)

    Log.d( tag: "tag", newPost.toString())

    databaseRef.push().setValue(newPost)
}
```

Figure 129 Create New Forum Code

Figure 127 New Forum allows the user to enter a title and description along with the submit button to confirm the creation of a new forum. Figure 128 New Forum Validation Code ensures the user enters text for both title and description or it will prompt the specific field is empty. Figure 129 Create New Forum Validation Code shows how the information is inserted into the database. To create a new forum, the application requires the following information: the user ID of the authentication user, the title and description of the forum, as well as the date and time of the forum was created.

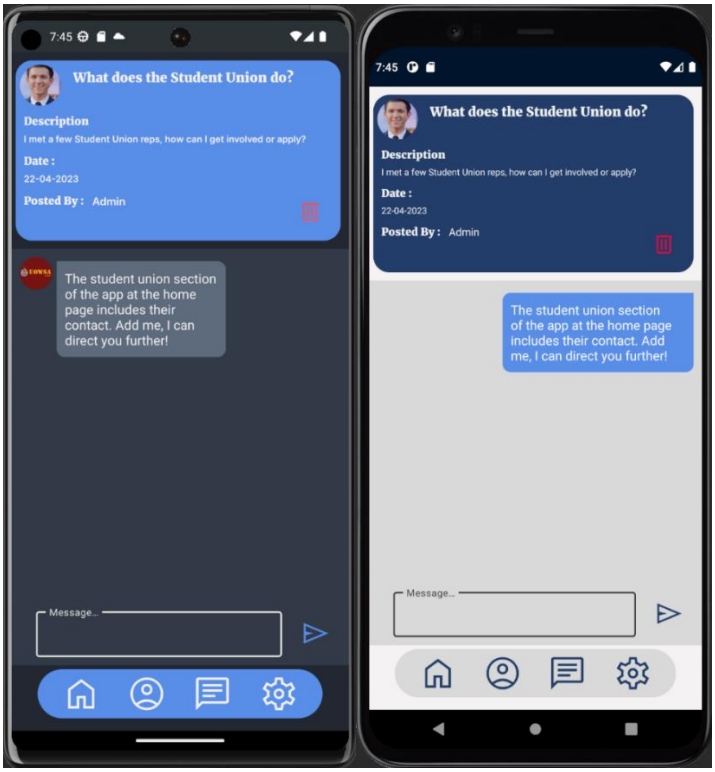


Figure 130 Forum Messaging

```
studentUnionCreatorId = intent.getStringExtra( name: "creatorId").toString()
val studyTitle = intent.getStringExtra( name: "studyTitle").toString()
val studyDescription = intent.getStringExtra( name: "studyDescription").toString()
val studyDate = intent.getStringExtra( name: "studyDate").toString()
studentUnionCreatedDateTime = intent.getStringExtra( name: "createdDateTime").toString()

val postedUserImage = findViewById<CircleImageView>(R.id.circleImageView)
val title = findViewById<TextView>(R.id.title)
val description = findViewById<TextView>(R.id.description)
val date = findViewById<TextView>(R.id.date)
val postedBy = findViewById<TextView>(R.id.postedBy)

title.setText(studyTitle)
description.setText(studyDescription)
date.setText(studyDate)

val databaseRef = FirebaseDatabase.getInstance().getReference( path: "Users").child(studentUnionCreatorId!!)
databaseRef.child( pathString: "userName").get().addOnSuccessListener { it: DataSnapshot!
    val name = it.value.toString()
    postedBy.setText(name)
}
databaseRef.child( pathString: "profileImage").get().addOnSuccessListener { it: DataSnapshot!
    val image = it.value.toString()
    Glide.with( activity: this).load(image).into(postedUserImage)
}
```

Figure 131 Forum Messaging Code

In Figure 130 Forum Messaging, the implementation of the forum feature includes who created the forum including how the specific forum information is retrieved. Figure 131 Forum Messaging Code shows the information for title description and date is transferred from locally stored data from Figure 127 New Forum, however username and profile image are retrieved from the database.

6.5.9 Implementation – Themes & Colours XML

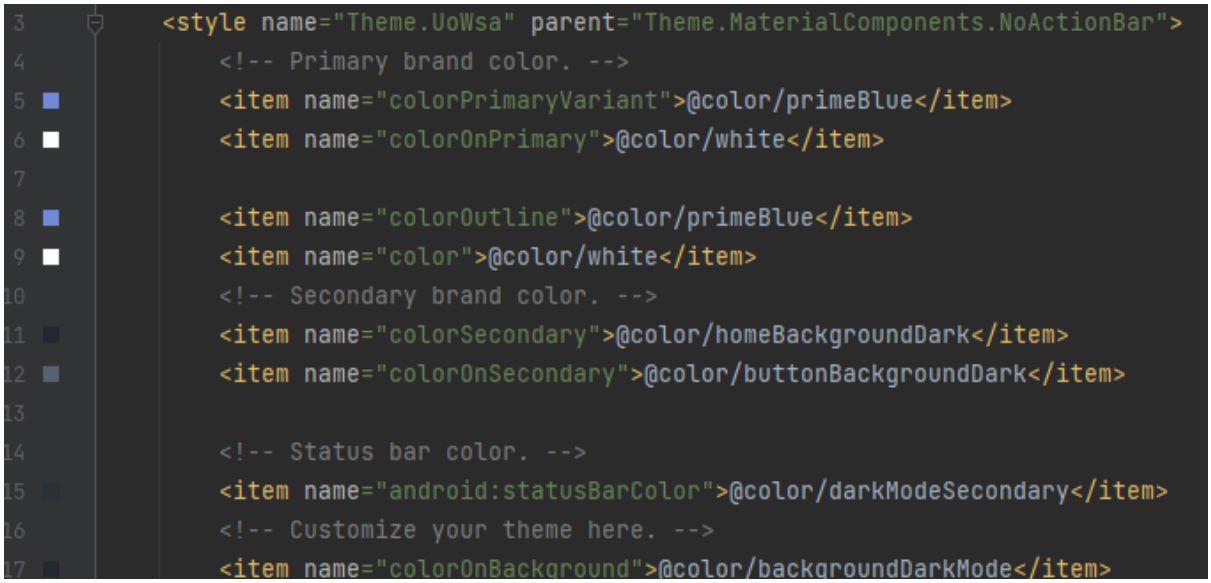
The themes and colours used in the application can be separated depending on the device settings from the user.

```
45  <color name="backgroundLightMode">#D9D9D9</color>
46      <color name="backgroundDarkMode">#222831</color>
47      <color name="navBarLightMode">#f4f3f3</color>
48      <color name="navBarDarkMode">#588EE8</color>
49      <color name="buttonColorLightMode">#233E6A</color>
50      <color name="buttonColorDarkMode">#f4f3f3</color>
51      <color name="colorErrorLightMode">#FC0D38</color>
52      <color name="colorErrorDarkMode">#FF4D4D</color>
53      <color name="colorRecMessage">#5E6B7D</color>
54      <color name="senderMessageColor">#588EE8</color>
55      <color name="receiverColorLightMode">#f4f3f3</color>
56      <color name="receiverColorDarkMode">#5E6B7D</color>
57      <color name="messageBackgroundDarkMode">#323B48</color>
58      <color name="pendingLightMode">#323B48</color>
59      <color name="pendingDarkMode">#323B48</color>
60      <color name="onlineLightMode">#34B511</color>
61      <color name="onlineDarkMode">#6EBF58</color>
62  </resources>
```

Figure 132 Colors XML Code

```
3   <style name="Theme.UoWsa" parent="Theme.MaterialComponents.NoActionBar">
4       <!-- Primary brand color. -->
5       <item name="colorPrimaryVariant">@color/navy</item>
6       <item name="colorOnPrimary">@color/white</item>
7
8       <item name="colorOutline">@color/navy</item>
9       <item name="color">@color/white</item>
0       <!-- Secondary brand color. -->
1       <item name="colorSecondary">@color/backgroundColorLight</item>
2       <item name="colorOnSecondary">@color/navLight</item>
3
4
5       <!-- Status bar color. -->
6       <item name="android:statusBarColor">@color/systemMenuColor</item>
7       <item name="textInputStyle">@color/darkModePrimary</item>
8       <!-- Customize your theme here. -->
```

Figure 133 Light Mode Code



```
<style name="Theme.UoWsa" parent="Theme.MaterialComponents.NoActionBar">
    <!-- Primary brand color. -->
    <item name="colorPrimaryVariant">@color/primeBlue</item>
    <item name="colorOnPrimary">@color/white</item>

    <item name="colorOutline">@color/primeBlue</item>
    <item name="color">@color/white</item>
    <!-- Secondary brand color. -->
    <item name="colorSecondary">@color/homeBackgroundDark</item>
    <item name="colorOnSecondary">@color/buttonBackgroundDark</item>

    <!-- Status bar color. -->
    <item name="android:statusBarColor">@color/darkModeSecondary</item>
    <!-- Customize your theme here. -->
    <item name="colorOnBackground">@color/backgroundDarkMode</item>
```

Figure 134 Dark Mode Code

Figure 132 Colours XML Code includes all of the created variable names along with the colour hex code to clearly identify the name and value. The variables implemented will then be called in Figure 133 Light Mode Code and 134 Dark mode Code implement the colours for the application. The item name for both light and dark mode is identical, however the values used from the Figure 132 will separate for light and dark mode users. To change between light and dark mode, it can be executed from the device settings.

7. Testing

This purpose of this chapter is to provide a comprehensive amount of testing process for the developed application. All features, both implemented and not implemented, will be examined during testing. The chapter will also address the various testing methods that were employed, as well as any challenges that arose during the process. These challenges may have resulted in the development of desired and luxury features or caused features to be disregarded.

7.1 Obstacles Encountered

This subchapter will provide an analysis and detailed account of all the challenges encountered during the development of an Android Studio Kotlin project. Specifically, it will focus on the issues identified through the use of throwaway prototypes. This section will explain how bugs and related problems were detected and how they impacted the development process. The solution to these problems will also be unfolded with each problem encountered.

7.1.1 Android Studio Update

Issue details: Android Studio provides frequent updates throughout the 7months of this project, however some of these updates has caused a catastrophic Gradle build issue. All codes written previously on every single file cannot be interpreted and appears red, it will not run after the update. It becomes a bigger problem attempting to return to the previous version and the build icon button does not appear.

Solution: After doing countless researching into the matter information a user on Stack Overflow posted a similar issue I was encountering, during the first prototype this issue was resolved using the answers given.

1. Delete .gradle file
2. Rebuild the project
3. Clean the project
4. Install APK
5. If issue still persists, disable instant run

(*StackOverflow [1], 2019*)

The problem still persists over different updates; however, the first solution did not work for other updates afterwards. The file version of the project does not change, despite the frequent updates from Android Studio, therefore every time an update releases I will have to manually change version compiler.

7.1.2 Firebase Realtime Database Rules and Permissions

Issue details: Creating a firebase Realtime Database gives the developer an option to first create and test the database without implementing any rules. This is because the database comes with predefined rules to ensure security. However, after 30 days of implementation, Firebase sends a notification requiring immediate action to insert new rules for the database as the default test rules are set to expire. The implementation of new rules based on the resources provided by Firebase resulted in an error when attempting to write or retrieve data from the database. The issue was discovered when retrieved data from the database returned as null. After several tests, it was almost certain that the issue was caused by the rules governing the transit of data.

Solution: To address the issue with Firebase Realtime Database, the research involved integrating both Firebase Authentication and Firebase Database. When using Firebase Authentication, it is important to set the level of security behind a specific user's identification in the rules. This ensures that only authorised users can access their own account, view, read, and write content. However, this extra layer of security was not defined in the Firebase guide, which lead to further research. By learning a specific style and set of rules from Stack Overflow, the issue was resolved. These new rules were implemented successfully, ensuring data transit was secure and accessible only by the authorised user (*StackOverflow [2], 2017*).

7.2 Test Coverage

The test coverage will include both White Box Testing and Black Box Testing. White box testing involves testing the application where the tester has knowledge of the structure, design, and implementation (*checkpoint, n.d.*) of the application being tested. White Box testing will involve the client-side, server-side, storage, desirables, and luxury requirements which were implemented from subchapter 4.3.8 Refined List of Requirements.

7.2.1 Client-Side White Box Testing – Essential

Table 49 Client-Side White Box Testing - Essential

Test No.	Test Case	Expected Outcomes	Actual Outcomes	Pass / Fail
1	The app icon from the Android device main screen is accessible	The app icon is visible and selectable to land on the starting app page	The app icon is visible and selectable to land on the starting app page	Pass
2	Starting App page select Terms of Service button	Opens a new page with Terms of Service contents	Opens a new page with Terms of Service contents	Pass
3	Starting App page	Opens the Sign up	Opens the Sign up	Pass

	select Sign Up button	page	page	
4	Starting App page select Login button	Opens the Login page	Opens the Login page	Pass
5	Sign Up page enter email, username, password, and confirm password text fields	Can use the keyboard to type on all text fields on the Sign up page	Can use the keyboard to type on all text fields on the Sign up page	Pass
6	Sign Up page empty email, username, password and confirm password	Error indicating, please insert a “(corresponding text field missing)”	Error indicating, please insert a “(corresponding text field missing)”	Pass
7	Sign Up page Invalid email address	Error indicating, the email inserted is not a defined email address	Error indicating, the email inserted is not a defined email address	Pass
8	Sign Up page Invalid username “Admin” with different variants of upper and lower case and select submit button	Error indicating, the username “(inserted username)” is reserved, please enter a new username	Error indicating, the username “(inserted username)” is reserved, please enter a new username	Pass
9	Sign Up page Username less than 4 or more than 12 characters and select submit button	Error indicating, please enter a username between 4 – 12 characters	Error indicating, please enter a username between 4 – 12 characters	Pass
10	Sign Up page Confirm password is not identical to password and select submit button	Error indicating, the confirm password inserted is incorrect	Error indicating, the confirm password inserted is incorrect	Pass
11	Sign Up page with valid details and selects the sign up button	Message indicating, You’re registered successfully. User will then be directed to the login page	Message indicating, You’re registered successfully. User will then be directed to the login page	Pass
12	Sign Up page selecting the Already have an	Directed to the login page	Directed to the login page	Pass

	account? Sign in button			
13	Login Page selecting the Don't have an account? Sign Up button	Directed to the Sign up page	Directed to the Sign up page	Pass
14	Login Page selecting the forgot my password button	Directed to the Forgot my password page	Directed to the Forgot my password page	Pass
15	Forgot my password page entering non existing email address on the platform and selects the submit button	Error message indicating, There is no user corresponding to this identifier, The user may have been deleted.	Error message indicating, There is no user corresponding to this identifier, The user may have been deleted.	Pass
16	Forgot my password page invalid email address and selects the submit button	Error indicating email address is badly formatted	Error indicating email address is badly formatted	Pass
17	Forgot my password page inserts email address correctly and selects the submit button	Email sent successfully to reset your password	Email sent successfully to reset your password	Pass
18	Login Page empty email or password and selects the login button	Error message indicating, please insert "(corresponding field missing)"	Error message indicating, please insert "(corresponding field missing)"	Pass
19	Login page Incorrect Details and selects sign in button	Error message indicating, failed to login	Error message indicating, failed to login	Pass
20	Login page Correct Details and selects sign in button	Message indicating you've signed in successfully and directed to the home page	Message indicating you've signed in successfully and directed to the home page	Pass
21	Home page select logout icon	Opens the Logout Dialog box with logout and cancel buttons	Opens the Logout Dialog box with logout and cancel buttons	Pass
22	Logout Dialog	Dialog box closes	Dialog box closes	Pass

	box selects cancel			
23	Logout Dialog box selects logout	Directed to the Starting app screen	Directed to the Starting app screen	Pass
24	Home Page selects Discussion Board button	Directed to the discussion board page	Directed to the discussion board page	Pass
25	Home Page selects Friends button	Directed to the friends page	Directed to the friends page	Pass
26	Home Page selects UoW Message button	Directed to the UoW Message page	Directed to the UoW Message page	Pass
27	Home Page selects Student Union button	Directed to the Student Union page	Directed to the Student Union page	Pass
28	Home Page selects Clubs and Societies button	Directed to the Clubs and Societies page	Directed to the Clubs and Societies page	Pass
29	Home Page selects Student Wellbeing button	Directed to the Student Wellbeing page	Directed to the Student Wellbeing page	Pass
30	Home Page selects Profile button	Directed to the profile page	Directed to the profile page	Pass
31	Home Page selects Settings button	Directed to the settings page	Directed to the settings page	Pass
32	Discussion Board Page (Hangout) selects General button	Directed to the General All Chat message page	Directed to the General All Chat message page	Pass
33	Discussion Board Page (Hangout) selects Make New Friends button	Directed to the Make New Friends All Chat message page	Directed to the Make New Friends All Chat message page	Pass
34	Discussion Board Page (Hangout) selects Student Wellbeing button	Directed to the Student Wellbeing All Chat message page	Directed to the Student Wellbeing All Chat message page	Pass
35	Discussion Board Page (Forums) select Study button	Directed to the Study forums page to view all study forums created and create a new study forum	Directed to the Study forums page to view all study forums created and create a new study forum	Pass
36	Discussion Board	Directed to the	Directed to the	Pass

	Page (Forums) select Events button	Events forums page to view all events forums created and create a new events forum	Events forums page to view all events forums created and create a new events forum	
37	Discussion Board Page (Forums) select Accommodation button	Directed to the Accommodation forums page to view all Accommodation forums created and create a new Accommodation forum	Directed to the Accommodation forums page to view all Accommodation forums created and create a new Accommodation forum	Pass
38	Discussion Board Page (Forums) select Student Union button	Directed to the Student Union forums page to view all Student Union forums created and create a new Student Union forum	Directed to the Student Union forums page to view all Student Union forums created and create a new Student Union forum	Pass
39	Discussion Board Page (Forums) select Clubs and Societies button	Directed to the Clubs and Societies forums page to view all Clubs and Societies forums created and create a new Clubs and Societies forum	Directed to the Clubs and Societies forums page to view all Clubs and Societies forums created and create a new Clubs and Societies forum	Pass
40	Discussion Board Page (Forums) select Cavendish Campus button	Directed to the Cavendish Campus forums page to view all Cavendish Campus forums created and create a new Cavendish Campus forum	Directed to the Cavendish Campus forums page to view all Cavendish Campus forums created and create a new Cavendish Campus forum	Pass
41	Discussion Board Page (Forums) select Harrow Campus button	Directed to the Harrow Campus forums page to view all Harrow Campus forums created and create a new Harrow Campus forum	Directed to the Harrow Campus forums page to view all Harrow Campus forums created and create a new Harrow Campus forum	Pass
42	Discussion Board Page (Forums) select Regents Campus button	Directed to the Regents Campus forums page to view all Regents Campus forums created and create a new Regents Campus forum	Directed to the Regents Campus forums page to view all Regents Campus forums created and create a new Regents Campus forum	Pass
43	Discussion Board	Directed to the	Directed to the	Pass

	Page (Forums) select Marylebone Campus button	Marylebone Campus forums page to view all Marylebone Campus forums created and create a new Marylebone Campus forum	Marylebone Campus forums page to view all Marylebone Campus forums created and create a new Marylebone Campus forum	
44	Discussion Board Page (Forums) select Other button	Directed to the Other forums page to view all Other forums created and create a new Other forum	Directed to the Other forums page to view all Other forums created and create a new Other forum	Pass
45	Hangout General Messaging all chat page selecting a specific user profile image	Directed to the specific target selected user profile page	Directed to the specific target selected user profile page	Pass
46	Hangout General Messaging all chat page selects send button with empty message	Error message indicating, message is empty	Error message indicating, message is empty	Pass
47	Hangout General Messaging all chat page selects messaging box and type on keyboard and selects the send button	The inserted message will be displayed on the right	The inserted message will be displayed on the right	Pass
48	Hangout General Messaging all chat page other user sends a message	The inserted message will be displayed on the left	The inserted message will be displayed on the left	Pass
49	Hangout General Messaging all chat page selects a specific message on the right	Opens a dialog indicating if you would like to delete this message along with two buttons, delete and cancel	Opens a dialog indicating if you would like to delete this message along with two buttons, delete and cancel	Pass
50	Delete Dialog box select cancel	Closes the dialog box	Closes the dialog box	Pass
51	Delete dialog box	Message deleted	Message deleted	Pass
52	Hangout General Messaging all chat page selects delete all messages button	All messages on the right are deleted	All messages on the right are deleted	Pass

53	Hangout Make New Friends Messaging all chat page selecting a specific user profile image	Directed to the specific target selected user profile page	Directed to the specific target selected user profile page	Pass
54	Hangout Make New Friends Messaging all chat page selects send button with empty message	Error message indicating, message is empty	Error message indicating, message is empty	Pass
55	Hangout Make New Friends Messaging all chat page selects messaging box and type on keyboard and selects the send button	The inserted message will be displayed on the right	The inserted message will be displayed on the right	Pass
56	Hangout Make New Friends Messaging all chat page other user sends a message	The inserted message will be displayed on the left	The inserted message will be displayed on the left	Pass
57	Hangout Make New Friends Messaging all chat page selects a specific message on the right	Opens a dialog indicating if you would like to delete this message along with two buttons, delete and cancel	Opens a dialog indicating if you would like to delete this message along with two buttons, delete and cancel	Pass
58	Hangout Make New Friends Messaging all chat page selects delete all messages button	All messages on the right are deleted	All messages on the right are deleted	Pass
59	Hangout Student Wellbeing Messaging all chat page selecting a specific user profile image	Directed to the specific target selected user profile page	Directed to the specific target selected user profile page	Pass
60	Hangout Student Wellbeing	Error message indicating, message	Error message indicating, message	Pass

	Messaging all chat page selects send button with empty message	is empty	is empty	
61	Hangout Student Wellbeing Messaging all chat page selects messaging box and type on keyboard and selects the send button	The inserted message will be displayed on the right	The inserted message will be displayed on the right	Pass
62	Hangout Student Wellbeing Messaging all chat page other user sends a message	The inserted message will be displayed on the left	The inserted message will be displayed on the left	Pass
63	Hangout Student Wellbeing Messaging all chat page selects a specific message on the right	Opens a dialog indicating if you would like to delete this message along with two buttons, delete and cancel	Opens a dialog indicating if you would like to delete this message along with two buttons, delete and cancel	Pass
64	Hangout Student Wellbeing Messaging all chat page selects delete all messages button	All messages on the right are deleted	All messages on the right are deleted	Pass
65	Study Forum page select create new forum button	Directed to create a new study forum	Directed to create a new study forum	Pass
66	Events Forum page select create new forum button	Directed to create a new events forum	Directed to create a new events forum	Pass
67	Accommodation Forum page select create new forum button	Directed to create a new accommodation forum	Directed to create a new accommodation forum	Pass
68	Student Union Forum page select create new forum button	Directed to create a new student union forum	Directed to create a new student union forum	Pass
69	Clubs and Societies Forum	Directed to create a new clubs and	Directed to create a new clubs and	Pass

	page select create new forum button	societies forum	societies forum	
70	Cavendish Campus Forum page select create new forum button	Directed to create a new cavendish campus forum	Directed to create a new cavendish campus forum	Pass
71	Harrow Campus Forum page select create new forum button	Directed to create a new harrow campus forum	Directed to create a new harrow campus forum	Pass
72	Regents Campus Forum page select create new forum button	Directed to create a new regents campus forum	Directed to create a new regents campus forum	Pass
73	Marylebone Campus Forum page select create new forum button	Directed to create a new marylebone campus forum	Directed to create a new marylebone campus forum	Pass
74	Other Forum page select create new forum button	Directed to create a new other forum	Directed to create a new other forum	Pass
75	Study Create new forum page empty title or description and selects submit button	Error indicating (title / description) is empty	Error indicating (title / description) is empty	Pass
76	Events Create new forum page empty title or description and selects submit button	Error indicating (title / description) is empty	Error indicating (title / description) is empty	Pass
77	Accommodation Create new forum page empty title or description and selects submit button	Error indicating (title / description) is empty	Error indicating (title / description) is empty	Pass
78	Student Union Create new forum page empty title or description and selects submit button	Error indicating (title / description) is empty	Error indicating (title / description) is empty	Pass
79	Clubs and	Error indicating (title /	Error indicating (title /	Pass

	Societies Create new forum page empty title or description and selects submit button	description) is empty	description) is empty	
80	Cavendish Campus Create new forum page empty title or description and selects submit button	Error indicating (title / description) is empty	Error indicating (title / description) is empty	Pass
81	Harrow Campus Create new forum page empty title or description and selects submit button	Error indicating (title / description) is empty	Error indicating (title / description) is empty	Pass
82	Regents Campus Create new forum page empty title or description and selects submit button	Error indicating (title / description) is empty	Error indicating (title / description) is empty	Pass
83	Marylebone Campus Create new forum page empty title or description and selects submit button	Error indicating (title / description) is empty	Error indicating (title / description) is empty	Pass
84	Other Create new forum page empty title or description and selects submit button	Error indicating (title / description) is empty	Error indicating (title / description) is empty	Pass
85	Study Create new forum page inserted title and description and selects submit button	Directed to the study forums page with the title, date and who it was posted by	Directed to the study forums page with the title, date and who it was posted by	Pass
86	Events Create new forum page inserted title and description and selects submit button	Directed to the events forums page with the title, date and who it was posted by	Directed to the events forums page with the title, date and who it was posted by	Pass

87	Accommodation Create new forum page inserted title and description and selects submit button	Directed to the accommodation forums page with the title, date and who it was posted by	Directed to the accommodation forums page with the title, date and who it was posted by	Pass
88	Student Union Create new forum page inserted title and description and selects submit button	Directed to the student union forums page with the title, date and who it was posted by	Directed to the student union forums page with the title, date and who it was posted by	Pass
89	Clubs and Societies Create new forum page inserted title and description and selects submit button	Directed to the clubs and societies forums page with the title, date and who it was posted by	Directed to the clubs and societies forums page with the title, date and who it was posted by	Pass
90	Cavendish Campus Create new forum page inserted title and description and selects submit button	Directed to the cavendish campus forums page with the title, date and who it was posted by	Directed to the cavendish campus forums page with the title, date and who it was posted by	Pass
91	Harrow Campus Create new forum page inserted title and description and selects submit button	Directed to the harrow campus forums page with the title, date and who it was posted by	Directed to the harrow campus forums page with the title, date and who it was posted by	Pass
92	Regents Campus Create new forum page inserted title and description and selects submit button	Directed to the regents campus forums page with the title, date and who it was posted by	Directed to the regents campus forums page with the title, date and who it was posted by	Pass
93	Marylebone Campus Create new forum page inserted title and description and selects submit button	Directed to the Marylebone campus forums page with the title, date and who it was posted by	Directed to the Marylebone campus forums page with the title, date and who it was posted by	Pass
94	Other Create new forum page inserted title and description and selects submit	Directed to the other forums page with the title, date and who it was posted by	Directed to the other forums page with the title, date and who it was posted by	Pass

	button			
95	Study Forum page only the creator of the forum can view and select delete forum button	Opens an alert dialog with title delete forum and description, Are you sure you want to delete this forum? All data inside this forum will be erased. Two optional buttons include delete and cancel. Cancel closes the alert dialog and delete removes the forum from the study forum page.	Opens an alert dialog with title delete forum and description, Are you sure you want to delete this forum? All data inside this forum will be erased. Two optional buttons include delete and cancel. Cancel closes the alert dialog and delete removes the forum from the study forum page.	Pass
96	Events Forum page only the creator of the forum can view and select delete forum button	Opens an alert dialog with title delete forum and description, Are you sure you want to delete this forum? All data inside this forum will be erased. Two optional buttons include delete and cancel. Cancel closes the alert dialog and delete removes the forum from the events forum page.	Opens an alert dialog with title delete forum and description, Are you sure you want to delete this forum? All data inside this forum will be erased. Two optional buttons include delete and cancel. Cancel closes the alert dialog and delete removes the forum from the events forum page.	Pass
97	Accommodation Forum page only the creator of the forum can view and select delete forum button	Opens an alert dialog with title delete forum and description, Are you sure you want to delete this forum? All data inside this forum will be erased. Two optional buttons include delete and cancel. Cancel closes the alert dialog and delete removes the forum from the accommodation forum page.	Opens an alert dialog with title delete forum and description, Are you sure you want to delete this forum? All data inside this forum will be erased. Two optional buttons include delete and cancel. Cancel closes the alert dialog and delete removes the forum from the accommodation forum page.	Pass
98	Student Union Forum page only the creator of the forum can view and select delete forum button	Opens an alert dialog with title delete forum and description, Are you sure you want to delete this forum? All data inside this forum	Opens an alert dialog with title delete forum and description, Are you sure you want to delete this forum? All data inside this forum	Pass

		will be erased. Two optional buttons include delete and cancel. Cancel closes the alert dialog and delete removes the forum from the student union forum page.	will be erased. Two optional buttons include delete and cancel. Cancel closes the alert dialog and delete removes the forum from the student union forum page.	
99	Clubs and societies Forum page only the creator of the forum can view and select delete forum button	Opens an alert dialog with title delete forum and description, Are you sure you want to delete this forum? All data inside this forum will be erased. Two optional buttons include delete and cancel. Cancel closes the alert dialog and delete removes the forum from the clubs and societies forum page.	Opens an alert dialog with title delete forum and description, Are you sure you want to delete this forum? All data inside this forum will be erased. Two optional buttons include delete and cancel. Cancel closes the alert dialog and delete removes the forum from the clubs and societies forum page.	Pass
100	Cavendish Campus Forum page only the creator of the forum can view and select delete forum button	Opens an alert dialog with title delete forum and description, Are you sure you want to delete this forum? All data inside this forum will be erased. Two optional buttons include delete and cancel. Cancel closes the alert dialog and delete removes the forum from the cavendish campus forum page.	Opens an alert dialog with title delete forum and description, Are you sure you want to delete this forum? All data inside this forum will be erased. Two optional buttons include delete and cancel. Cancel closes the alert dialog and delete removes the forum from the cavendish campus forum page.	Pass
101	Harrow Campus Forum page only the creator of the forum can view and select delete forum button	Opens an alert dialog with title delete forum and description, Are you sure you want to delete this forum? All data inside this forum will be erased. Two optional buttons include delete and cancel. Cancel closes the alert dialog and	Opens an alert dialog with title delete forum and description, Are you sure you want to delete this forum? All data inside this forum will be erased. Two optional buttons include delete and cancel. Cancel closes the alert dialog and	Pass

		delete removes the forum from the harrow campus forum page.	delete removes the forum from the harrow campus forum page.	
102	Regents Campus Forum page only the creator of the forum can view and select delete forum button	Opens an alert dialog with title delete forum and description, Are you sure you want to delete this forum? All data inside this forum will be erased. Two optional buttons include delete and cancel. Cancel closes the alert dialog and delete removes the forum from the regents campus forum page.	Opens an alert dialog with title delete forum and description, Are you sure you want to delete this forum? All data inside this forum will be erased. Two optional buttons include delete and cancel. Cancel closes the alert dialog and delete removes the forum from the regents campus forum page.	Pass
103	Marylebone Campus Forum page only the creator of the forum can view and select delete forum button	Opens an alert dialog with title delete forum and description, Are you sure you want to delete this forum? All data inside this forum will be erased. Two optional buttons include delete and cancel. Cancel closes the alert dialog and delete removes the forum from the Marylebone campus forum page.	Opens an alert dialog with title delete forum and description, Are you sure you want to delete this forum? All data inside this forum will be erased. Two optional buttons include delete and cancel. Cancel closes the alert dialog and delete removes the forum from the Marylebone campus forum page.	Pass
104	Other Forum page only the creator of the forum can view and select delete forum button	Opens an alert dialog with title delete forum and description, Are you sure you want to delete this forum? All data inside this forum will be erased. Two optional buttons include delete and cancel. Cancel closes the alert dialog and delete removes the forum from the other forum page.	Opens an alert dialog with title delete forum and description, Are you sure you want to delete this forum? All data inside this forum will be erased. Two optional buttons include delete and cancel. Cancel closes the alert dialog and delete removes the forum from the other forum page.	Pass
105	Friends page select search for	Directed to the search for friends	Directed to the search for friends	Pass

	friends button	page	page	
106	Friends page select Friend Requested button	Directed to the friend requested page	Directed to the friend requested page	Pass
107	Search for friends page select a specific user button	Specific user profile with the add button, if specific user and current user are already friends a non-clickable button indicating we're friends	Specific user profile with the add button, if specific user and current user are already friends a non-clickable button indicating we're friends	Pass
108	Specific user profile page select add friend button	Add friend button will appear with two buttons, pending friend request button which is non-clickable and cancel friend request which will cancel the friend request sent.	Add friend button will appear with two buttons, pending friend request button which is non-clickable and cancel friend request which will cancel the friend request sent.	Pass
109	Friends page	Can view the incoming friend request counter	Can view the incoming friend request counter	Pass
110	Friend requested page select specific user button	Directed to the specific user profile friend request page who requested to be friends	Directed to the specific user profile friend request page who requested to be friends	Pass
111	Specific user profile friend request page accept friend request or decline friend request button	Declining the friend request will direct the user to the friends page without showing the friend added, however accepting friend request will direct user to the friends page and show a new friend added to the friends list	Declining the friend request will direct the user to the friends page without showing the friend added, however accepting friend request will direct user to the friends page and show a new friend added to the friends list	Pass
112	Friends page selecting a specific user from friends list button	Directs the user to the specific user profile page	Directs the user to the specific user profile page	Pass
113	Specific user profile page remove friend and block button	Selecting the remove friend button will remove the user friend list. Blocking the user from friends	Selecting the remove friend button will remove the user friend list. Blocking the user from friends	Pass

		list will remove friend from the friends list and add the user to the blocked list	list will remove friend from the friends list and add the user to the blocked list	
114	UoW Message page view list of friends	Friends list appears in UoW Message	Friends list appears in UoW Message	Pass
115	UoW Message page select specific user button	Directed to the direct messaging page with the specific user	Directed to the direct messaging page with the specific user	Pass
116	Direct Messaging page view specific user details	Can view user profile image and user name	Can view user profile image and user name	Pass
117	Direct Messaging page no message and selects send button	Error message indicating, message is empty	Error message indicating, message is empty	Pass
118	Direct Messaging page selects send button with message	Can view sent message on the right side of the messaging view	Can view sent message on the right side of the messaging view	Pass
119	Direct Messaging page another user sends message	Can view sent message on the left side of the messaging view	Can view sent message on the left side of the messaging view	Pass
120	Direct Messaging selecting a specific message on the right side	Opens a dialog box with title and description. The title illustrates delete message, and the message indicates if you would like this message to be deleted? Selecting delete will delete the message and selecting cancel will close the dialog box	Opens a dialog box with title and description. The title illustrates delete message, and the message indicates if you would like this message to be deleted? Selecting delete will delete the message and selecting cancel will close the dialog box	Pass
121	Direct Messaging selecting the delete all bin icon button	Opens a dialog box with title and description. The title illustrates delete message, and the message indicates if you would like to delete all of your messages? Selecting delete will delete all	Opens a dialog box with title and description. The title illustrates delete message, and the message indicates if you would like to delete all of your messages? Selecting delete will delete all	Pass

		messages and selecting cancel will close the dialog box	messages and selecting cancel will close the dialog box	
122	Student Union page view student union content	Can view all of the information in regard to what student union offers	Can view all of the information in regard to what student union offers	Pass
123	Student Union page Select Facebook icon button	Opens the student union Facebook link on an external default browser set by the user	Opens the student union Facebook link on an external default browser set by the user	Pass
124	Student Union page Select Twitter icon button	Opens the student union Twitter link on an external default browser set by the user	Opens the student union Twitter link on an external default browser set by the user	Pass
125	Student Union page Select Instagram icon button	Opens the student union Instagram link on an external default browser set by the user	Opens the student union Instagram link on an external default browser set by the user	Pass
126	Student Union page Select LinkedIn icon button	Opens the student union LinkedIn link on an external default browser set by the user	Opens the student union LinkedIn link on an external default browser set by the user	Pass
127	Student Union page Select YouTube icon button	Opens the student union YouTube link on an external default browser set by the user	Opens the student union YouTube link on an external default browser set by the user	Pass
128	Clubs and Societies page view Clubs and Societies content	Can view all of the information in regard to what clubs and societies offers	Can view all of the information in regard to what clubs and societies offers	Pass
129	Clubs and Societies page selecting Sports teams find out more button	Directed to Sports teams page	Directed to Sports teams page	Pass
130	Clubs and Societies page selecting Society Groups find out more button	Directed to Society Groups page	Directed to Society Groups page	Pass
131	Sports teams page select uwsu sports teams button	Opens the sports team page on the student union website	Opens the sports team page on the student union website	Pass

132	Society groups page select uwsu society groups button	Opens the society groups page on the student union website	Opens the society groups page on the student union website	Pass
133	Student Wellbeing page view student wellbeing content	Can view all of the information in regard to what student wellbeing offers	Can view all of the information in regard to what student wellbeing offers	Pass
134	Student wellbeing page select wellbeing resource page button	Opens the student wellbeing university of Westminster website link on an external default browser set by the user	Opens the student wellbeing university of Westminster website link on an external default browser set by the user	Pass
135	Student wellbeing page select feeling good app button	Opens the student wellbeing university of Westminster feeling good app website link on an external default browser set by the user	Opens the student wellbeing university of Westminster feeling good app website link on an external default browser set by the user	Pass
136	Student wellbeing page select report and support incident button	Opens the student wellbeing university of Westminster report and report incident website link on an external default browser set by the user	Opens the student wellbeing university of Westminster report and report incident website link on an external default browser set by the user	Pass
137	Student wellbeing page select five ways to wellbeing button	Opens the student wellbeing university of Westminster YouTube link on an external default browser set by the user	Opens the student wellbeing university of Westminster YouTube link on an external default browser set by the user	Pass
138	Student wellbeing page select disability learning support button	Opens the student wellbeing university of Westminster disability learning support website link on an external default browser set by the user	Opens the student wellbeing university of Westminster disability learning support website link on an external default browser set by the user	Pass
139	User Profile page view user profile content	Can view all the information on what is set by the user	Can view all the information on what is set by the user	Pass
140	User Profile page	Opens user photo	Opens user photo	Pass

	select upload profile image button	gallery	gallery	
141	User photo gallery and select an image	Returns to the application with the image selected and replaced by the previous / default image set on the profile page	Returns to the application with the image selected and replaced by the previous / default image set on the profile page	Pass
142	User Profile page select confirm upload button	Go to a separate page and return to the profile page to confirm the profile image is replaced correctly.	Go to a separate page and return to the profile page to confirm the profile image is replaced correctly.	Pass
143	Settings page select Update profile button	Directed to the Update profile page	Directed to the Update profile page	Pass
144	Settings page select Blocked users button	Directed to the Blocked users page	Directed to the Blocked users page	Pass
145	Settings page select Administrator button	Directed to the Administrator page	Directed to the Administrator page	Pass
146	Settings page select Contact us button	Opens the user's mailing application with the recipient UoWsa support email address automatically entered	Opens the user's mailing application with the recipient UoWsa support email address automatically entered	Pass
147	Settings page select Help center button	Directed to the Help center page	Directed to the Help center page	Pass
148	Settings page select Terms of Service button	Directed to the Terms of Service page	Directed to the Terms of Service page	Pass
149	Settings page select About us button	Directed to the About us page	Directed to the About us page	Pass
150	Settings page select Delete account button	Directed to the Delete account page	Directed to the Delete account page	Pass
151	Settings page select Logout button	Logs out of the user account and directed to the start up page	Logs out of the user account and directed to the start up page	Pass
152	Update profile page view all current user	Can view and update user profile information including,	Can view and update user profile information including,	Pass

	profile information	username, status, campus location, and study course	username, status, campus location, and study course	
153	Update profile page username is empty and select confirm update button	Error message indicating, please enter a username	Error message indicating, please enter a username	Pass
154	Update profile page username is less than or more than 12 characters and select confirm update button	Error message indicating, please enter a username between 4 – 12 characters	Error message indicating, please enter a username between 4 – 12 characters	Pass
155	Update profile page Invalid username “Admin” with different variants of upper and lower case and select confirm update button	Error indicating, the username “(inserted username)” is reserved, please enter a new username	Error indicating, the username “(inserted username)” is reserved, please enter a new username	Pass
156	Update profile page status is empty and select confirm update button	Directs to the User profile page with the default value “Hey, UoWsa is the future!”	Directs to the User profile page with the default value “Hey, UoWsa is the future!”	Pass
157	Update profile page optional campus location and study course empty and select confirm update button	Directs to the User profile page with no campus location and study course information included	Directs to the User profile page with no campus location and study course information included	Pass
158	Update profile page with all valid information for username, status, campus location and study course inserted into the text field and select confirm update button	Directs to the user profile page with all inserted information including, username, status, campus location, and study course on display	Directs to the user profile page with all inserted information including, username, status, campus location, and study course on display	Pass
159	Blocked users page on all users who was blocked	Can view a list of blocked users	Can view a list of blocked users	Pass
160	Blocked users	A dialog box with the	A dialog box with the	Pass

	page selecting a specific user	title Unblock user and description are you sure you want to unblock (“specific username”)? This follows with two buttons, unblock and cancel. Cancel closes the dialog box and unblock will remove the user from the blocked list and added back to the friends list	title Unblock user and description are you sure you want to unblock (“specific username”)? This follows with two buttons, unblock and cancel. Cancel closes the dialog box and unblock will remove the user from the blocked list and added back to the friends list	
161	Help center page view all frequently asked questions	Can read all of the frequently asked questions in regard to the questions asked by other users	Can read all of the frequently asked questions in regard to the questions asked by other users	Pass
162	Delete account page view content	Can view title please be aware once you delete your account your account cannot be recovered, are you sure you want to delete your account? This follows with two buttons. Cancel will direct the user back to the settings menu and confirm will direct the user to the app start up page	Can view title please be aware once you delete your account your account cannot be recovered, are you sure you want to delete your account? This follows with two buttons. Cancel will direct the user back to the settings menu and confirm will direct the user to the app start up page	Pass

7.2.2 Server-Side (Database) White Box Testing - Essential

Table 50 Server-Side (Database) White Box Testing - Essential

Test No.	Test Case	Expected Outcomes	Actual Outcomes	Pass / Fail
1	User Sign Up	User sign up details stored on authentication database and user name in Firebase Realtime Database	User sign up details stored on authentication database and user name in Firebase Realtime Database	Pass
2	User login with valid details	User authenticated and logs in	User authenticated and logs in	Pass
3	User login invalid details	User not authenticated with error message	User not authenticated with error message	Pass
4	User Password reset with valid email address	User identity confirmed and password reset email sent to user emailing service	User identity confirmed and password reset email sent to user emailing service	Pass
5	User Password reset with invalid email address	Error message indicating the email address provided is not recognised	Error message indicating the email address provided is not recognised	Pass
6	Hangout General creates a new message with user ID, message and datetime	Shows a new entry in DB General of the database with new user message including the user ID, message and datetime	Shows a new entry in DB General of the database with new user message including the user ID, message and datetime	Pass
7	Hangout Make New Friends creates a new message with user ID, message and datetime	Shows a new entry in DB Make New Friends of the database with new user message including the user ID, message and datetime	Shows a new entry in DB Make New Friends of the database with new user message including the user ID, message and datetime	Pass
8	Hangout Student Wellbeing creates a new message with user ID, message and datetime	Shows a new entry in DB Student Wellbeing of the database with new user message including the user ID, message and datetime	Shows a new entry in DB Student Wellbeing of the database with new user message including the user ID, message and datetime	Pass

9	Forum Study create new forum	Shows a new entry in the Study Forum section of the database with the forum created using a title, description, userId and datetime	Shows a new entry in the Study Forum section of the database with the forum created using a title, description, userId and datetime	Pass
10	Forum Events create new forum	Shows a new entry in the Events Forum section of the database with the forum created using a title, description, userId and datetime	Shows a new entry in the Events Forum section of the database with the forum created using a title, description, userId and datetime	Pass
11	Forum Accommodation create new forum	Shows a new entry in the Accommodation Forum section of the database with the forum created using a title, description, userId and datetime	Shows a new entry in the Accommodation Forum section of the database with the forum created using a title, description, userId and datetime	Pass
13	Forum Student Union create new forum	Shows a new entry in the Student Union Forum section of the database with the forum created using a title, description, userId and datetime	Shows a new entry in the Student Union Forum section of the database with the forum created using a title, description, userId and datetime	Pass
14	Forum Clubs and Societies create new forum	Shows a new entry in the Clubs and Societies Forum section of the database with the forum created using a title, description, userId and datetime	Shows a new entry in the Clubs and Societies Forum section of the database with the forum created using a title, description, userId and datetime	Pass
15	Forum Cavendish Campus create new forum	Shows a new entry in the Cavendish Campus Forum section of the database with the forum created using a title, description, userId and datetime	Shows a new entry in the Cavendish Campus Forum section of the database with the forum created using a title, description, userId and datetime	Pass
16	Forum Harrow Campus create new forum	Shows a new entry in the Harrow Campus Forum section of the database with the forum created using a title, description,	Shows a new entry in the Harrow Campus Forum section of the database with the forum created using a title, description,	Pass

		userId and datetime	userId and datetime	
17	Forum Regents Campus create new forum	Shows a new entry in the Regents Campus Forum section of the database with the forum created using a title, description, userId and datetime	Shows a new entry in the Regents Campus Forum section of the database with the forum created using a title, description, userId and datetime	Pass
18	Forum Marylebone Campus create new forum	Shows a new entry in the Marylebone Campus Forum section of the database with the forum created using a title, description, userId and datetime	Shows a new entry in the Marylebone Campus Forum section of the database with the forum created using a title, description, userId and datetime	Pass
19	Forum Other create new forum	Shows a new entry in the Other Forum section of the database with the forum created using a title, description, userId and datetime	Shows a new entry in the Other Forum section of the database with the forum created using a title, description, userId and datetime	Pass
20	Forum Study Message enter a new message	Shows a new entry in the Forum Study and specific user unique identifier User ID and datetime, this will then show all the conversations under this node of the database including sender user id, message and datetime	Shows a new entry in the Forum Study and specific user unique identifier User ID and datetime, this will then show all the conversations under this node of the database including sender user id, message and datetime	Pass
21	Forum Events Message enter a new message	Shows a new entry in the Forum Events and specific user unique identifier User ID and datetime, this will then show all the conversations under this node of the database including sender user id, message and datetime	Shows a new entry in the Forum Events and specific user unique identifier User ID and datetime, this will then show all the conversations under this node of the database including sender user id, message and datetime	Pass
22	Forum Accommodation Message enter a	Shows a new entry in the Forum Accommodation and	Shows a new entry in the Forum Accommodation and	Pass

	new message	specific user unique identifier User ID and datetime, this will then show all the conversations under this node of the database including sender user id, message and datetime	specific user unique identifier User ID and datetime, this will then show all the conversations under this node of the database including sender user id, message and datetime	
23	Forum Student Union Message enter a new message	Shows a new entry in the Forum Student Union and specific user unique identifier User ID and datetime, this will then show all the conversations under this node of the database including sender user id, message and datetime	Shows a new entry in the Forum Student Union and specific user unique identifier User ID and datetime, this will then show all the conversations under this node of the database including sender user id, message and datetime	Pass
24	Forum Clubs and Societies Message enter a new message	Shows a new entry in the Forum Clubs and Societies and specific user unique identifier User ID and datetime, this will then show all the conversations under this node of the database including sender user id, message and datetime	Shows a new entry in the Forum Clubs and Societies and specific user unique identifier User ID and datetime, this will then show all the conversations under this node of the database including sender user id, message and datetime	Pass
25	Forum Cavendish Campus Message enter a new message	Shows a new entry in the Forum Cavendish Campus and specific user unique identifier User ID and datetime, this will then show all the conversations under this node of the database including sender user id, message and datetime	Shows a new entry in the Forum Cavendish Campus and specific user unique identifier User ID and datetime, this will then show all the conversations under this node of the database including sender user id, message and datetime	Pass
26	Forum Harrow Campus Message enter a new message	Shows a new entry in the Forum Harrow Campus and specific user unique identifier	Shows a new entry in the Forum Harrow Campus and specific user unique identifier	Pass

		User ID and datetime, this will then show all the conversations under this node of the database including sender user id, message and datetime	User ID and datetime, this will then show all the conversations under this node of the database including sender user id, message and datetime	
27	Forum Regents Campus Message enter a new message	Shows a new entry in the Forum Regents Campus and specific user unique identifier User ID and datetime, this will then show all the conversations under this node of the database including sender user id, message and datetime	Shows a new entry in the Forum Regents Campus and specific user unique identifier User ID and datetime, this will then show all the conversations under this node of the database including sender user id, message and datetime	Pass
28	Forum Marylebone Campus Message enter a new message	Shows a new entry in the Forum Marylebone Campus and specific user unique identifier User ID and datetime, this will then show all the conversations under this node of the database including sender user id, message and datetime	Shows a new entry in the Forum Marylebone Campus and specific user unique identifier User ID and datetime, this will then show all the conversations under this node of the database including sender user id, message and datetime	Pass
29	Forum Other Message enter a new message	Shows a new entry in the Forum Other and specific user unique identifier User ID and datetime, this will then show all the conversations under this node of the database including sender user id, message and datetime	Shows a new entry in the Forum Other and specific user unique identifier User ID and datetime, this will then show all the conversations under this node of the database including sender user id, message and datetime	Pass
30	Add Friend	Shows a friend request node built up with sender of the friend request and receiver of friend	Shows a friend request node built up with sender of the friend request and receiver of friend	Pass

		request	request	
31	Cancel Friend Request	Deletes the existing friend request node of the sender and receiver	Deletes the existing friend request node of the sender and receiver	Pass
32	Friend request received decline friend request	Deletes the existing friend request node of the sender and receiver	Deletes the existing friend request node of the sender and receiver	Pass
33	Friend request received accept friend request	Deletes the existing friend request node of the sender and receiver, add user to friends list under the user ID for both users	Deletes the existing friend request node of the sender and receiver, add user to friends list under the user ID for both users	Pass
34	Remove Friend	Removes the user from the friends list on the database	Removes the user from the friends list on the database	Pass
35	Block Friend	Removes the user from the friends list on the database and add user to the blocked list	Removes the user from the friends list on the database and add user to the blocked list	Pass
36	Unblock friend	Remove user from the blocked list and added to the friends list	Remove user from the blocked list and added to the friends list	Pass
37	Sending a message using UoW Message direct messaging	Creates new UoW Message node between 2 users with message contents	Creates new UoW Message node between 2 users with message contents	Pass
38	Remove selected Message	Deletes specific message	Deletes specific message	Pass
39	Remove all messages	Deletes all messages under the same sender	Deletes all messages under the same sender	Pass
40	Administrator Messageing	Creates new Administrator node between a standard user and admin	Creates new Administrator node between a standard user and admin	Pass
41	Delete Account	Removes all database details for the user and deletes the identifier User ID from authentication services	Removes all database details for the user and deletes the identifier User ID from authentication services	Pass
42	Logout	User logout from authentication services	User logout from authentication services	Pass

7.2.3 Storage White Box Testing - Essential

Table 51 Firebase Storage White Box Testing - Essential

Test No.	Test Case	Expected Outcomes	Actual Outcomes	Pass / Fail
1	User creates an account without image, when they login for the first time to the home page, it should automatically use the app image stored on Firebase storage an make a copy	Copies image from the storage and replace the image name to their authenticated unique identifier ID	Copies image from the storage and replace the image name to their authenticated unique identifier ID	Pass
2	User confirms profile image upload	Image stored in Firebase Storage	Image stored in Firebase Storage	Pass

7.2.4 White Box Testing - Desirables

Table 52 White Box Testing - Desirables

Test No.	Test Case	Expected Outcomes	Actual Outcomes	Pass / Fail
1	Navigation Menu select Home Page button from any other page	Directed to the home page	Directed to the home page	Pass
2	Navigation Menu select Profile Page button from any other page	Directed to the profile page	Directed to the profile page	Pass
3	Navigation Menu select UoW Message Page button from any other page	Directed to the UoW Message page	Directed to the UoW Message page	Pass
4	Navigation Menu select Settings Page button from any other page	Directed to the settings page	Directed to the settings page	Pass
5	About Us page view content	Can view all a brief description of the project	Can view all a brief description of the project	Pass

7.2.5 White Box Testing – Luxuries

Table 53 White Box Testing - Luxuries

Test No.	Test Case	Expected Outcomes	Actual Outcomes	Pass / Fail
1	User's device settings default to light mode	Appears with light theme	Appears with light theme	Pass
2	User's device settings default to dark mode	Appears with dark theme	Appears with dark theme	Pass
3	Notification counter UoW Message	Shows notifications counter for each unread message in UoW Message	Shows Notifications for each unread message in UoW Message	Pass

7.2.6 Black Box Testing – Compatibility

Testing on a physical mobile device an provide a more accurate representation of real-world usage and can help identify issues that may not be presented on an emulator such as sensors, feedback, and network connectivity.

Table 54 Black Box Testing – Compatibility

Test No.	Test Case	Expected Outcomes	Actual Outcomes	Pass / Fail
1	Test application using a physical device Samsung Galaxy Note 10+ 5G	All contents of the application are constraint and positioned correctly	All contents of the application are constraint and positioned correctly	Pass
2	Internet features including authentication, database, storage, and external web link on physical device	All contents are available whilst using online and offline mode	All contents are available whilst using online and offline mode	Pass
3	Test application using Pixel 4 XL API 30	All contents of the application are constraint and positioned correctly	All contents of the application are constraint and positioned correctly	Pass
3	Test application using Pixel 6 Pro API 33	All contents of the application are constraint and positioned correctly	All contents of the application are constraint and positioned correctly	Pass
3	Test application	All contents of the	All contents of the	Pass

	using Pixel XL API 33	application are constraint and positioned correctly	application are constraint and positioned correctly	
3	Internet features including authentication, database, storage, and external web link on Android emulator	All contents are available whilst using online and offline mode	All contents are available whilst using online and offline mode	Pass

7.2.7 Black Box Testing – User's Testing Questionnaire

To obtain feedback on the design and functionality of the project, a questionnaire was disseminated to two distinct groups of participants:

- University of Westminster students
- Academic adults with prior exposure to university life or social platform catering to students

This approach was chosen to obtain more precise outcomes compared to data collected from the general public. The questionnaire largely comprised mandatory multiple-choice questions, supplemented by two optional open-ended queries that aimed to identify any issues or areas for improvement. The total of responses amounted to 27, thereby ensuring unbiased feedback.

The questionnaire was designed to ensure that all participants we well-informed about the testing process. In Figure 135 Testing Questionnaire Consent Statement illustrates all data collected remained private and anonymous, and no participant identification was sought or recorded in the questionnaire. Participation was entirely voluntary, and participants were provided with an email address to withdraw their submission if they so desired.

Note: Please refer to Appendix XV for testing consent form responses details.

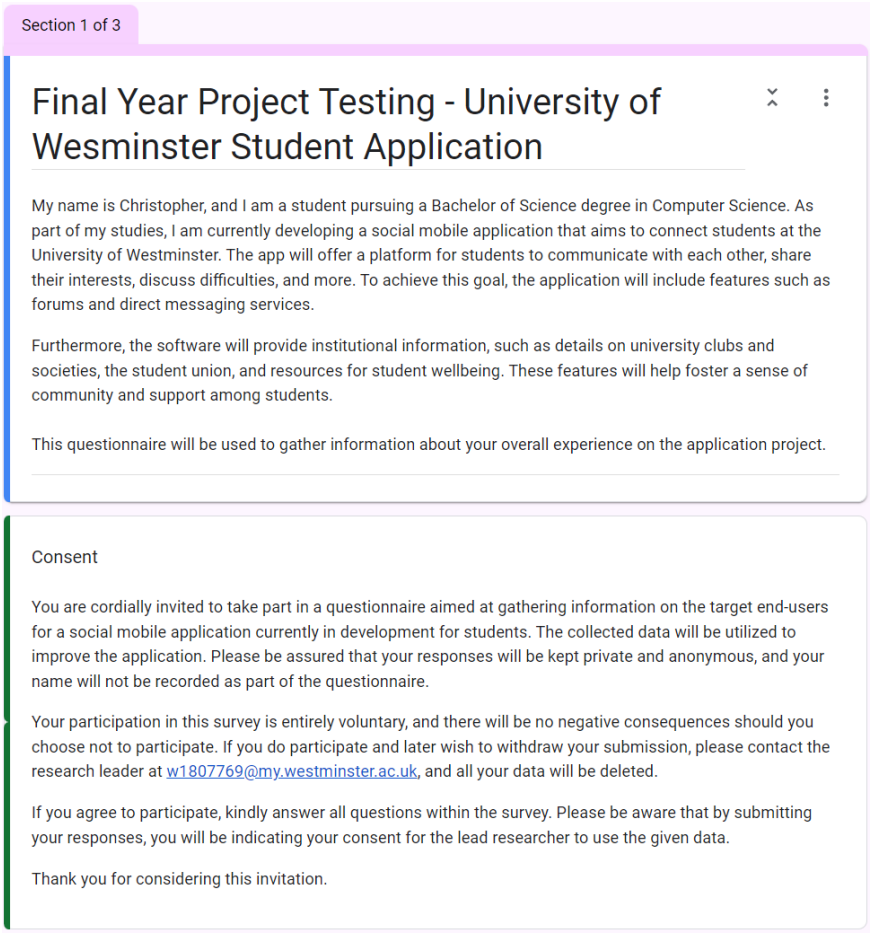


Figure 135 Testing Questionnaire Consent Statement

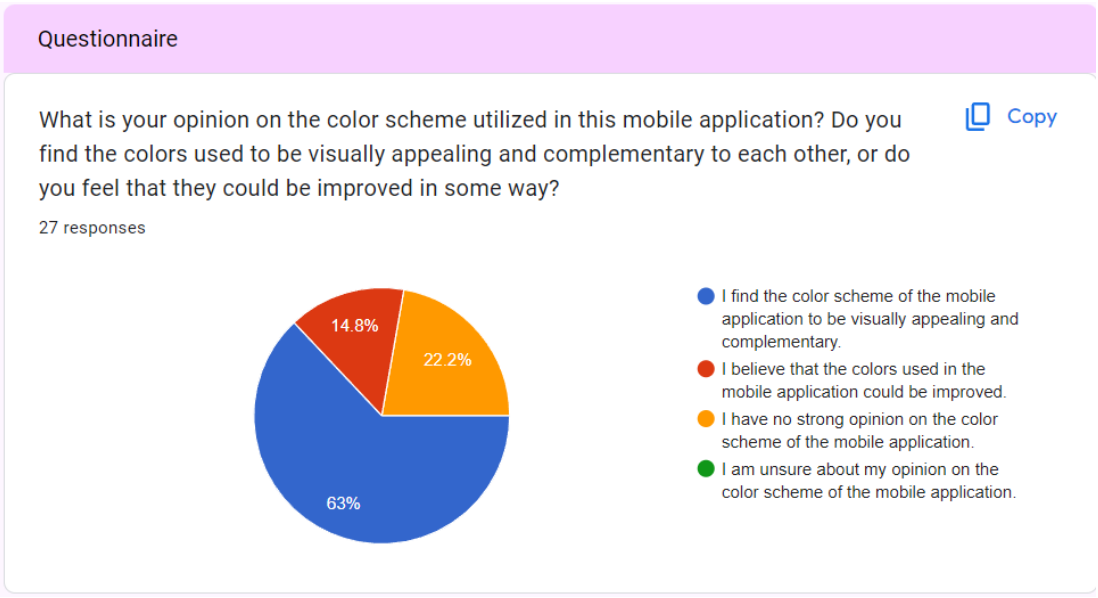


Figure 136 Testing Questionnaire Q1

From Figure 136 Testing Questionnaire Q1, 63% of the total participants believe the colour scheme used for both light and dark mode is visually appealing and complementary, however approximately a 1/3 of participants suggest the colours can be improved or do not have a strong opinion on the matter.

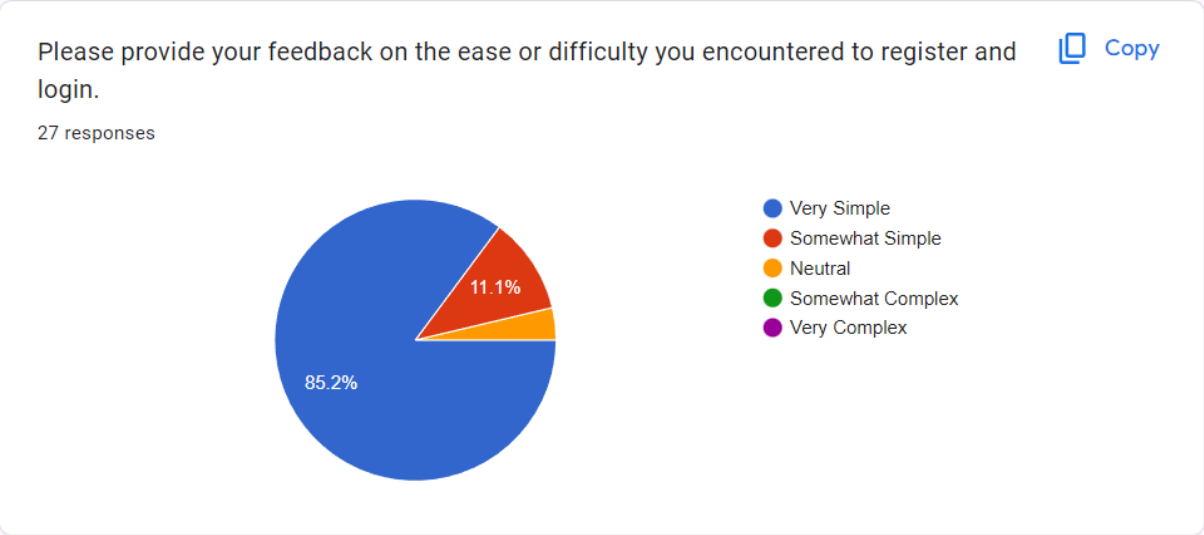


Figure 137 Testing Questionnaire Q2

From Figure 137 Testing Questionnaire Q2, over 90% of participants believe it is very simple or somewhat simple to create an account and login to the application. There are no participants who thought the registration and login process is complex.

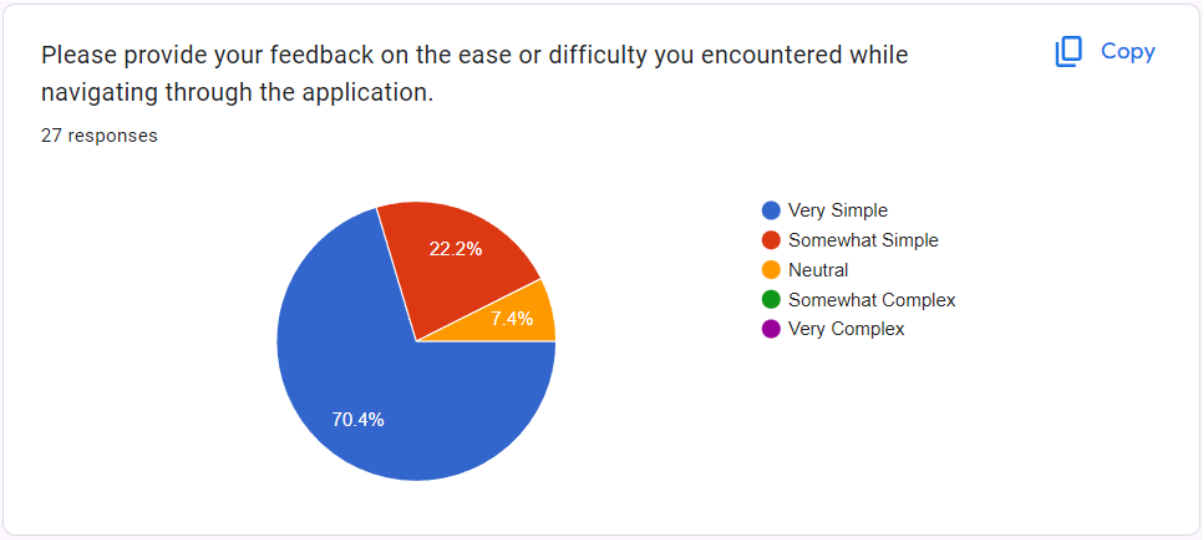


Figure 138 Testing Questionnaire Q3

From Figure 138 Testing Questionnaire Q3, over 90% of participants believe it was very simple or somewhat simple to navigate around the navigation. The rest of the participants believe the navigation experience is neutral.

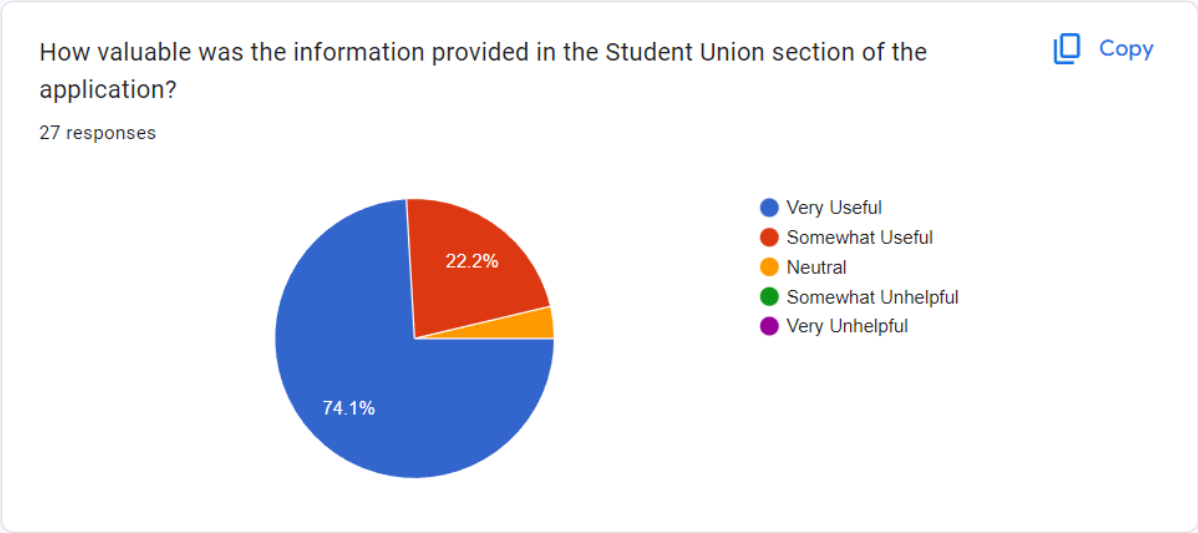


Figure 139 Testing Questionnaire Q4

From Figure 139 Testing Questionnaire Q4 illustrates how useful was the information provided in the Student Union, 95% of participants believe the information was useful, there were no unhelpful information used in the student union section of the application.

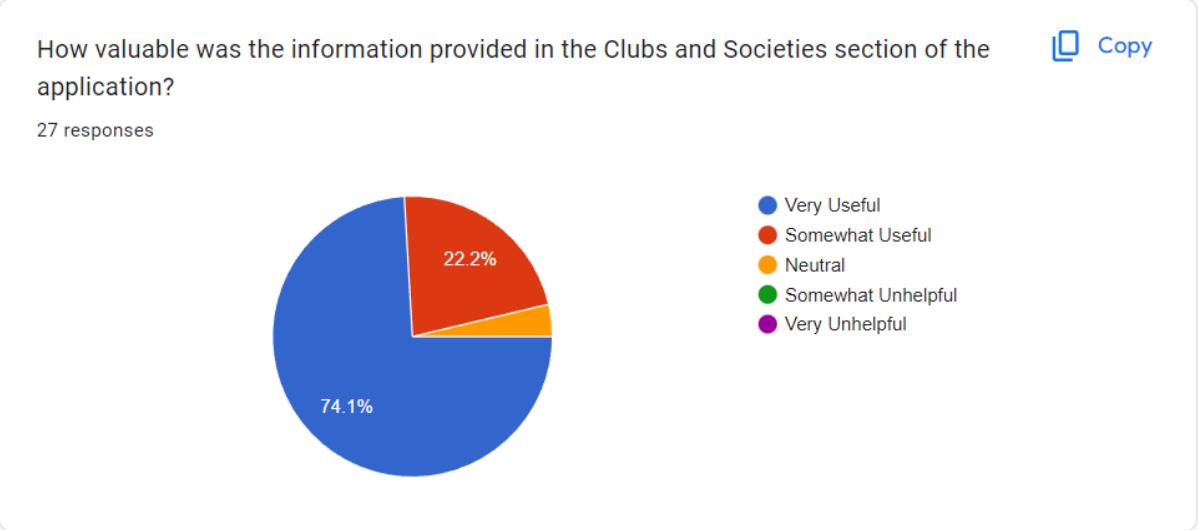


Figure 140 Testing Questionnaire Q5

From Figure 140 Testing Questionnaire Q5 illustrates how useful was the information provided in the Clubs and societies, 95% of participants believe the information was useful, there were no unhelpful information used in the clubs and societies section of the application.

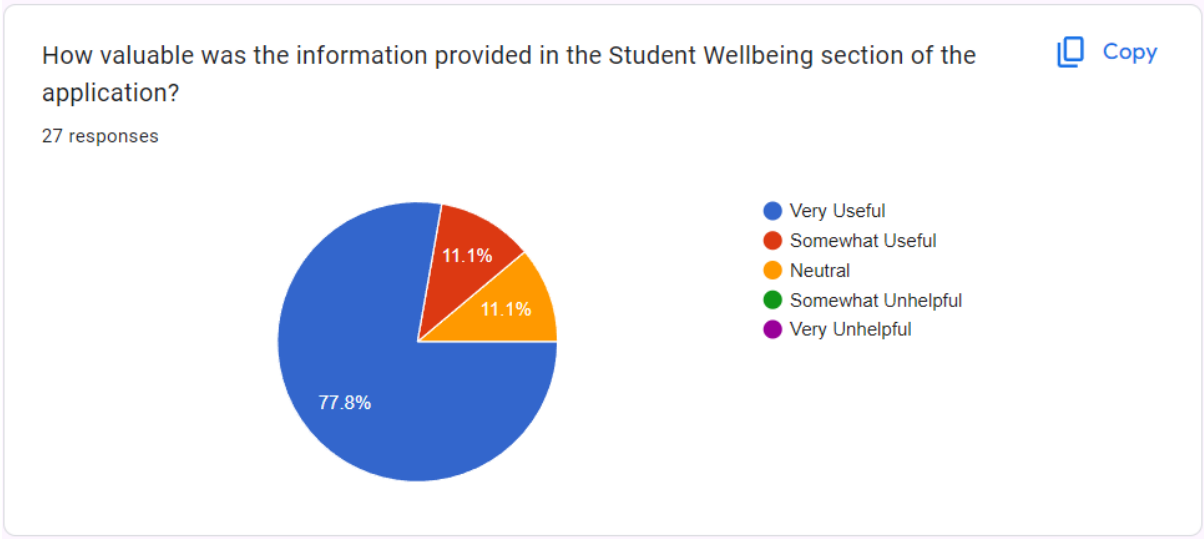


Figure 141 Testing Questionnaire Q6

From Figure 141 Testing Questionnaire Q6 illustrates how useful the information provided in Student wellbeing, 77.8% of participants believe it was very useful, 11.1% thought it was either somewhat useful or neutral.

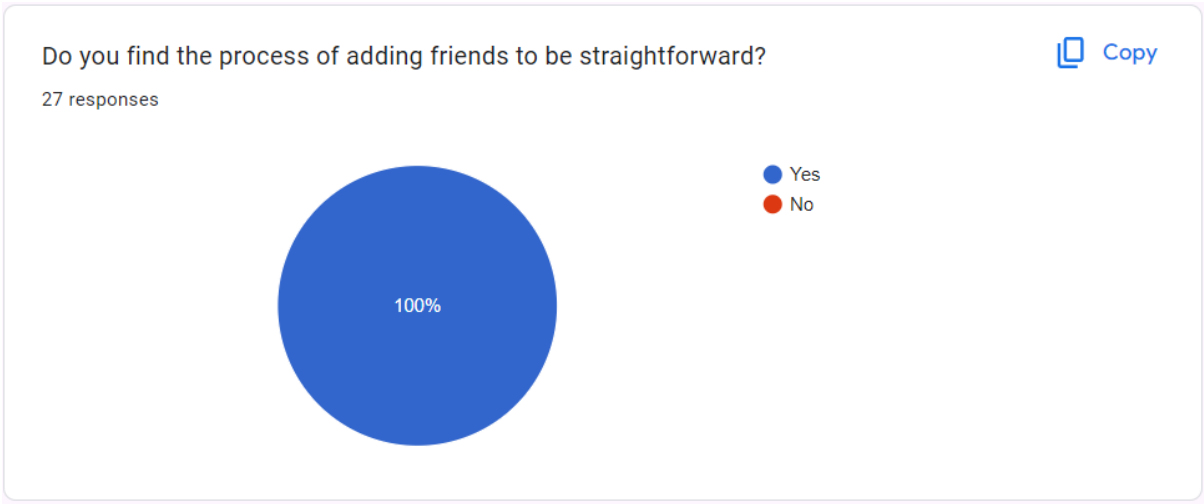


Figure 142 Testing Questionnaire Q7

From Figure 142 Testing Questionnaire Q7, all participants believe adding friends was straight forward.

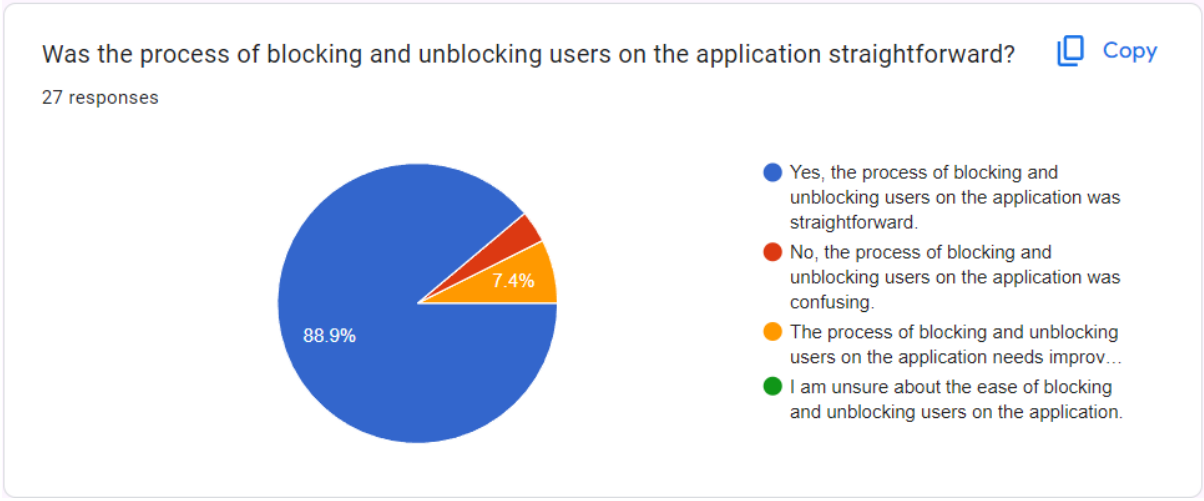


Figure 143 Testing Questionnaire Q8

From Figure 143 Testing Questionnaire Q8, 88.9% of participants believe the process of blocking and unblocking users on the application was straightforward, 7.9% believes improvements can be made and 1 participant believe the blocking and unblocking users was confusing.

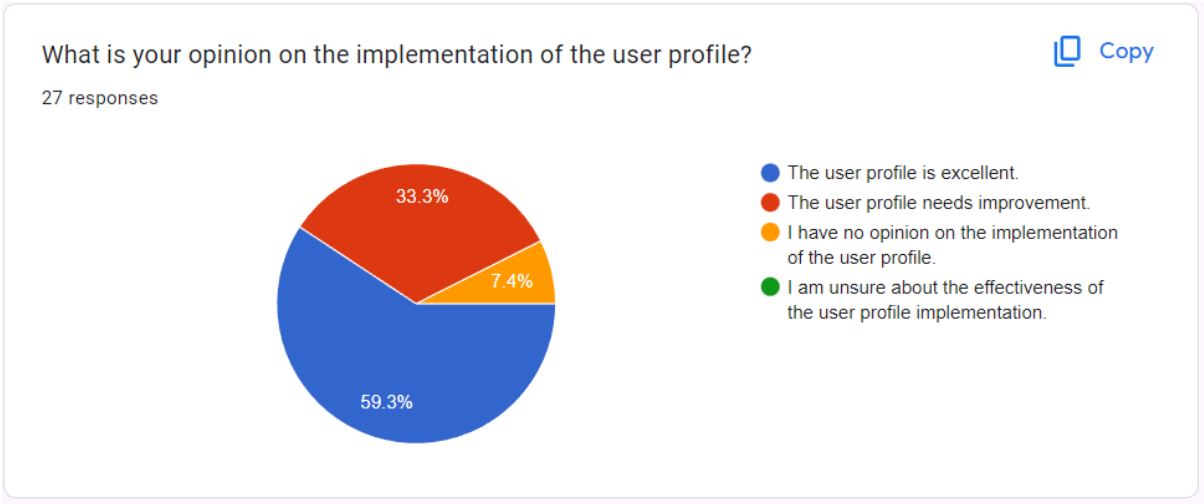


Figure 144 Testing Questionnaire Q9

From Figure 144 Testing Questionnaire Q9, 1/3 of users believe the user profile requires improvement. 59.3% of participants believes the user profile is excellent and 7.4% participants have no opinion on the user profile implementation.

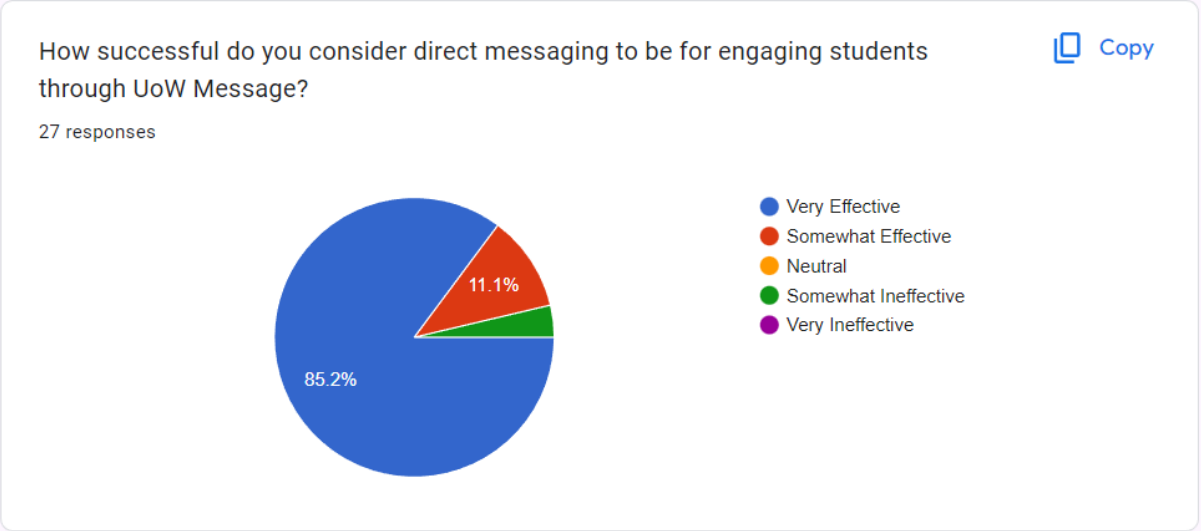


Figure 145 Testing Questionnaire Q10

From Figure 145 Testing Questionnaire Q10 illustrates the directing messaging UoW Message feature of the application. 85.2% of participants believe UoW Message is engaging for students, 11.1% believe its somewhat effective and 1 participant believe it was somewhat ineffective.

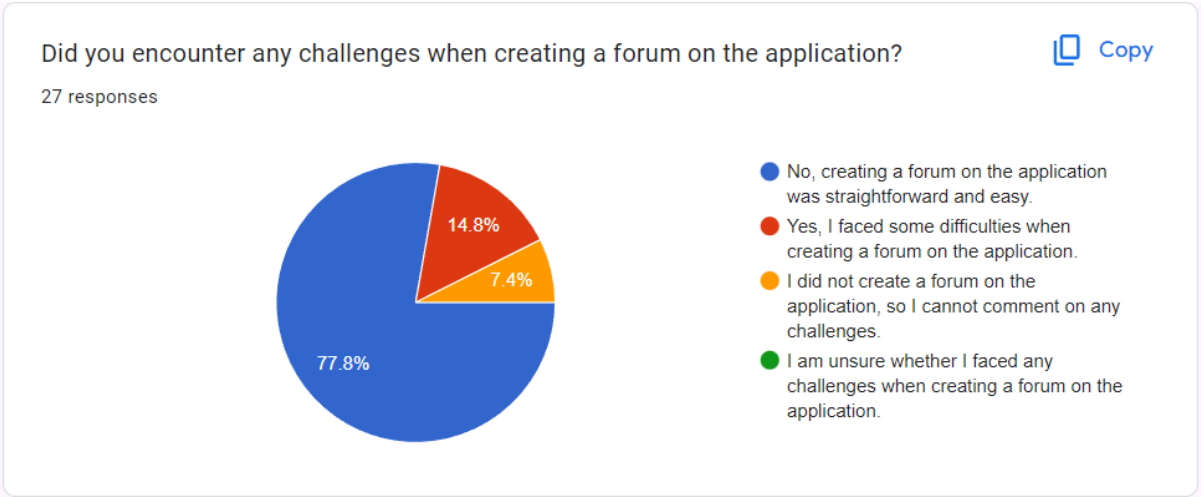


Figure 146 Testing Questionnaire Q11

From Figure 146 Testing Questionnaire Q11, 77.8% of participants believe it was straightforward and easy to create a new forum. 14.8% of participants found difficulties in creating a forum and 2 participants did not use this feature.

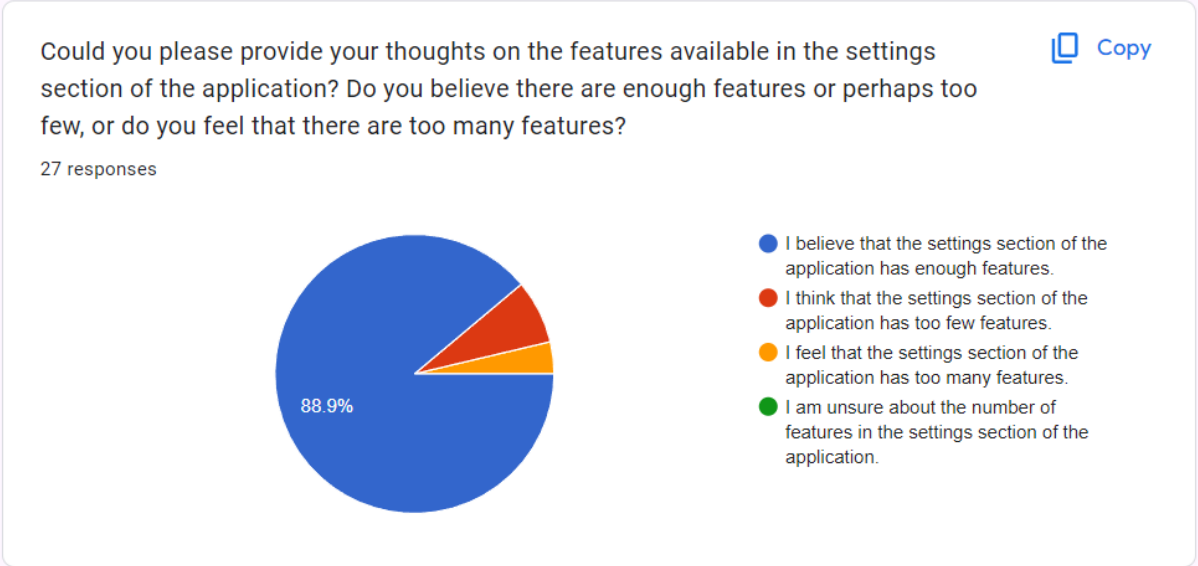


Figure 147 Testing Questionnaire Q12

From Figure 147 Testing Questionnaire Q12 24 participants believe there are enough and necessary features on the settings area of the application, however 4 participants thought there are either too few or too many features.

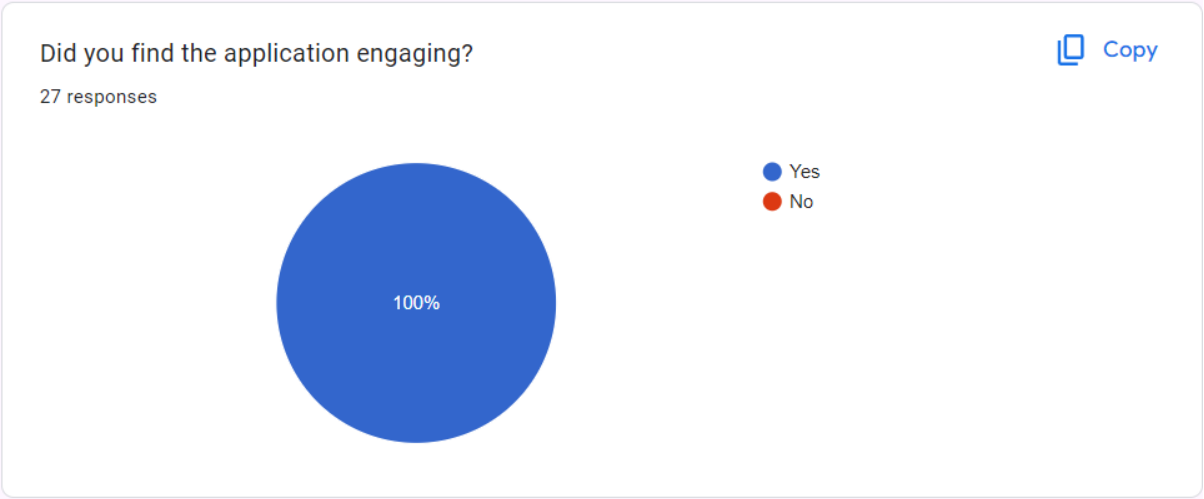


Figure 148 Testing Questionnaire Q13

From Figure 148 Testing Questionnaire Q13 all participants believe the application is engaging.

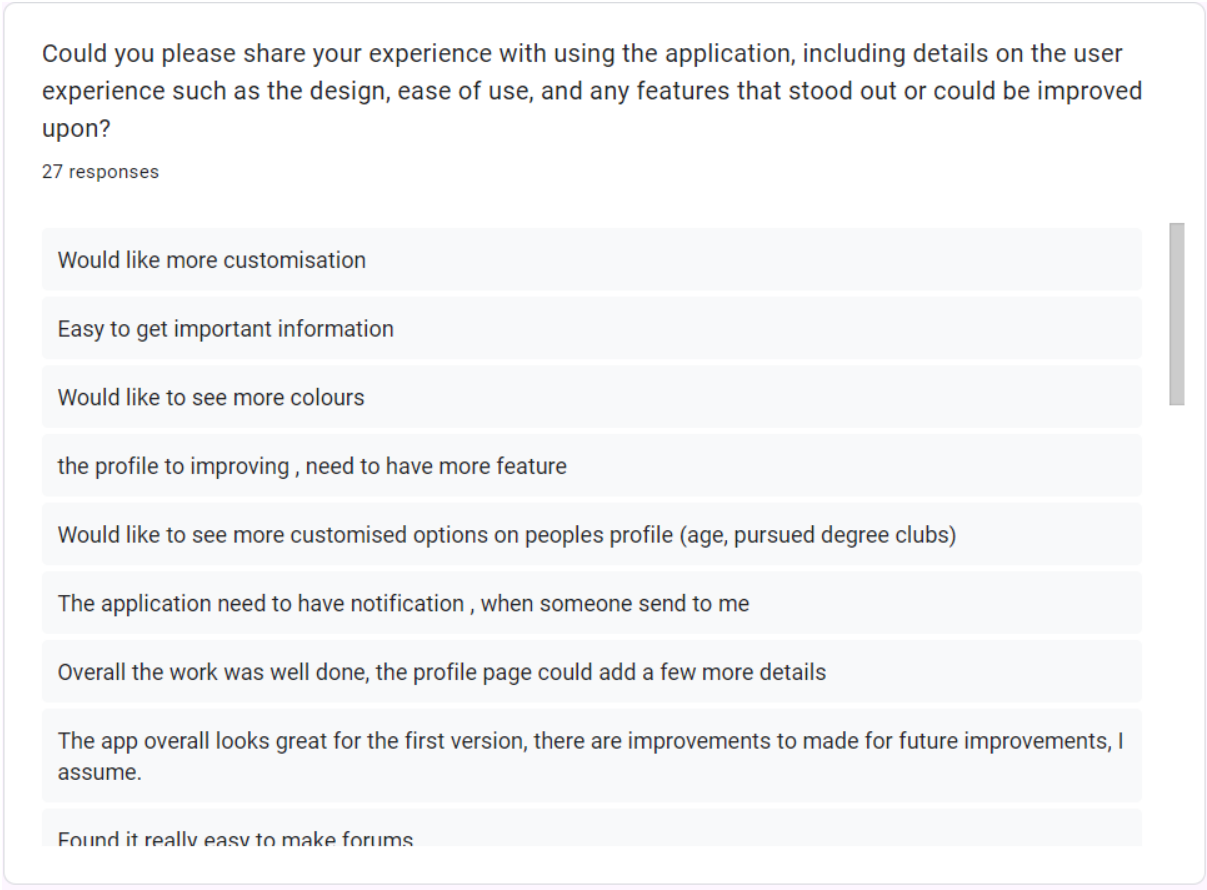


Figure 149 Testing Questionnaire Q14

Table 55 Testing Questionnaire Q14 Responses

Participants	Answer
1	Would like more customisation
2	Easy to get important information
3	Would like to see more colours
4	the profile to improving , need to have more feature
5	Would like to see more customised options on peoples profile (age, clubs)
6	The application need to have notification , when someone send to me
7	Overall the work was well done, the profile page could add a few more details
8	The app overall looks great for the first version, there are improvements to made for future improvements, I assume.
9	Found it really easy to make forums
10	The application looks great and very responsive.
11	Overall the design is positive for user experience. Perhaps the notifications can be expanded a little more throughout the application.
12	Easy way to communicate and find information on events
13	profile need to improve , and adding more feature

14	Would love a way to pin conversations
15	The application was very easy to use and understand. I especially liked the forum section. I think the application can benefit from having group direct messages.
16	I believe the direct messaging system as it is currently would need to have a way to display when a message is deleted as it feels like a bad actor would be able to abuse messages just being gone, and also adding a sort of preview of what the contents of a direct message would be as seeing just a number however big it could be could feel daunting or intimidating for some
17	Easy to use
18	Didn't encounter any problems using the app
19	The format is really easy to understand.
20	Made it easy to get in contact with students without the need of their phone numbers
21	when the user send a message, at least have notification because it important to know someone sending to us
22	Would like a way to filter forums
23	Love how easy the forum section is
24	Love the dark mode
25	I think the application can be improved to make the design more visually appealing and stylish, with high-quality graphics and animations
26	I think the block and unblock need to have more detail
27	Feels very professional. Would love to have an app like it!

Positives from participants

- Easy to gain knowledge on important information from the university
- Overall good design and responsive
- Easy Communication and find information on the Events Forum
- Application is simple
- Can contact other students without using phone numbers
- Dark mode looks great
- Feels very professional

Improvements from participants

- Colours used
- Require more information and details for each user profile
- Notifications can be expanded to push notifications
- Notifications can be used for other areas, not just UoW Message
- Possibility to pin messages
- Create a custom group direct messaging
- Show messages are deleted could cause ethical issues
- Notify a new message is received on the messaging page

- The use of a filter in the forums
- Possibly add high quality graphics and animations
- Block and unblock feature require more detail

Note: Please refer to Appendix XVI for full testing questionnaire report details.

7.3 Test Methodology

White box and Black box testing are two widely used software testing approaches that are essential to ensure the intended functionality of the system output (*Astari, 2022*).

In subchapter 7.2.1 Client-Side White Box Testing – Essential, 7.2.2 Server-Side (Database) White Box Testing – Essential, 7.2.3 Storage White Box Testing – Essential, 7.2.4 White Box Testing – Desirables, 7.2.5 White Box Testing – Luxuries, and 7.2.6 Black Box Testing – Compatibility, are all tested methods made by the developer of the project, are white box and black box testing to ensure the functionality of the application output are intended. This is followed by the method of pass and fail of the specific test cases involved.

Prior to administering the questionnaire, the UoWsa mobile application was disseminated to some participants to evaluate its full functionality using wither Android Studio AVD Manager or Genymotion. Some participants were also provided with the device to conduct in-person testing. Based on the results from Figures 7.2 to 7.15 of the testing questionnaire, it can be inferred that most participants found the application intuitive and captivating. More than 90% of participants indicated that creating an account, navigating the application, and accessing information in the Student Union and Clubs and Societies sections were straightforward. The majority also found the colour scheme, whether in light or dark mode, visually appealing and well-coordinated. Furthermore, all participants agreed that adding friends was uncomplicated.

However, certain participants suggested areas of improvement. Roughly a third of participants express that the colour scheme could be enhanced, and one participant found it challenging to block or unblock users. In addition, some participants proposed having more customisation options and additional features in the user profile section.

Overall, the results of the testing questionnaire indicate the UoWsa mobile application is generally well-designed and user-friendly. However, there are further opportunities for improvement to elevate the user experience.

8. Conclusion and Reflections

This chapter provides a comprehensive summary of the project by evaluating the accomplishments, areas of improvement, and skills acquired during the development of the mobile application. It also highlights the limitations of the project and proposed recommendations for its future development based on the overall project outcomes.

8.1 Results

It has been one academic year long process; from the proposal project requirements, design, implementation and delivering the project. Table 58 Results of Refined Functional Requirements illustrates the completed, partially completed, and incomplete functional requirements of the of the Refined Project Requirements. In Table 57 Functional requirement completion Table illustrates 66.66% of all functional requirements listed was completed during this project. The desirable requirement for events has been marked as partially complete because the implementation of certain features for events have only been implemented in the forums section. However, notifications have only been implemented in the direct messaging section of UoW Message due to time constraints during the project. Nonetheless, this implementation can be extended to other messaging areas of the application.

Table 56 Functional Requirement Completion Table

Requirement Priority	Percentage %
Essentials Completed	100
Essentials Partially Completed	0
Essentials Incomplete	0
Desirables Completed	28.57
Desirables Partially Competed	14.28
Desirables Incomplete	57.14
Luxury Completed	12.5
Luxury Partially Competed	12.5
Luxury Incomplete	75
Total Completed	**66.66**

Table 57 Results of Refined Functional Requirements

Priority	Requirements	Description	Completed / Partially Completed / Incomplete
(E)	Starting Menu	The application should open a new window and lead to the starting menu on an Android device.	Complete
(E)	Log in	The user is able to login to the home menu.	Complete
(E)	Security	The use of security behind the login	Complete

	Authentication Login	process.	
(E)	Log out	The user is able to log out.	Complete
(E)	Sign-Up	The user is able to sign-up and creates a new entity in the log in authentication.	Complete
(E)	Home Menu	The home menu should lead to different areas of the application.	Complete
(E)	Discussion Board / Forums	The user can see all the discussions and forums posted on the Discussion board / Forums section of the application.	Complete
(E)	Create new posts in the discussion board / forum	The user can create a new post with a Title and description.	Complete
(E)	Reply to posts in discussion board / forum	The user can visually see previous posts and reply to those posts.	Complete
(E)	Delete post and reply in discussion board / forum	The user can delete any post they have created or replied previously.	Complete
(E)	Add, Remove and Block	The user can add, remove, and block from their friends list.	Complete
(E)	Previous Messages	The user can visually see all the direct messages from the sender and recipient from previous conversations.	Complete
(E)	Add new Messages	The user can create a new message with their targeted user.	Complete
(E)	Delete Messages	The user is able to delete any messages they have sent previously.	Complete
(E)	Student Wellbeing	The user can explore the options of mental welfare and support from the university.	Complete
(E)	Student Union	The user can view all other students who represents them and what role does the student union take for all students.	Complete
(E)	Clubs and Societies	The user can view all the clubs and societies offered by the University of Westminster including a description.	Complete
(E)	Settings	The user can customise their application to their preference.	Complete
(E)	Friends	The user is able to see all the friends they have added including their connectivity status.	Complete
(E)	Profile	The user can see their profile name, profile image, connectivity state.	Complete
(E)	Admin	The admin can take control the forum section of the application including delete as well as messaging users	Complete

		without a friend's request.	
(D)	Navigation bar	The user can easily navigate to selected activities on all activities of the in the application.	Complete
(D)	Separate Admin Login area	The admin can login using alternative login area in comparison with the standard user.	Incomplete
(D)	Multiple users in private messaging	The ability for the user to add more users to a private messaging service.	Incomplete
(D)	Maps	The user can view the maps of all four campuses and directed to the targeted location.	Incomplete
(D)	About Us	The user can view the history of the application project including the developer of this app.	Complete
(D)	Courses	The degrees and modules with description available at the University of Westminster.	Incomplete
(D)	Events	The users are able to see the upcoming events offered by the university.	Partially Completed
(L)	Explicit content filter	The ability for students to toggle a function where the application is able to determine if there are explicit content and automatically removes this for the user.	Incomplete
(L)	Dark Mode	The user has the ability to toggle from light to dark mode and vice versa.	Complete
(L)	Posting images and emojis	The user has the ability to send images and emojis in the discussion board and the private messaging area.	Incomplete
(L)	Forum counter	The user can visually see how many posts they have made in the forum section.	Incomplete
(L)	Accessibility options	The user can adjust the size and font of text.	Incomplete
(L)	Accommodation	The user can view the recommended accommodations for students at the University of Westminster.	Incomplete
(L)	Student Life	The user can read upon what can be expected in a student including learning and structuring their time and not exclusive to find help where it's needed.	Incomplete
(L)	Notifications	The user is notified of any updates.	Partially completed

8.2 Future Development

This subchapter references all the future implementation plans for this project, these implementations will most likely start with the partially completed and incomplete desired and luxury requirements referenced in 8.2 Results of Refined Functional Requirements.

8.2.1 Separate Login Area

An advantage feature that could enhance the capabilities of application administrators is a distinct login function within the starting area of the application that grants them access to supplementary features. In the course of future development, it would be advantageous to provide administrators with the ability to modify the application further than its current state. With appropriate authorisation an administrator could be empowered to undertake actions such as removing accounts or communicating with users in the event of an issue.

8.2.2 Group Direct Messaging

The current messaging system in the application consists of direct messaging between two users, all chat messaging that is visible to all users, and forum discussions. However, there is a potential improvement that could be made by introducing a feature that enables group direct messaging. This functionality would provide users with the opportunity to create and join independent study groups and other groups. Overall, this implementation would enhance application's functionality by facilitating collaboration, improving communication, and promoting productive engagement among users.

8.2.3 Maps

The University of Westminster comprises four different campuses, namely Cavendish, Harrow, Regents, and Marylebone. Each of these campuses is located in a different are of the city, making it essential for students to be able to locate the campus they need to attend. To facilitate this, it would be beneficial to add a map feature to the application that would enable students to travel to their required campus with ease.

The map feature would allow students to access a visual representation of each campus's location, including details such as transportation options, and other pertinent information. This feature would be particularly useful for students who are new to the area or unfamiliar with the location of their campus.

In addition to the map feature, it would also be advantageous to include the floor plans for each campus within the application. These floor plans would provide a detail overview of the layout of each building, including the locations of classrooms, offices, and other important areas. By providing access to this information, students would be able to navigate their way around the university with ease, ensuring that they can find the locations they need quickly and efficiently.

8.2.4 Courses

The University of Westminster offers a vast array of courses, including hundreds of modules each year, which can make it challenging for students to navigate and access the content their require. To address this issue, a new Courses area of the application can be introduced to address this issue. This feature will include a comprehensive list of all courses offered by the university, including their descriptions, prerequisites, and availability. It would also include information on the different modules available for each course, including their titles, descriptions and any necessary prerequisites. This information would be presented in a clear and concise format, making it easy for students to browse and identify courses and modules that interest them.

8.2.5 Events

The University of Westminster hosts a range of events throughout the year, including seminars, workshops, exhibitions, student engagement, and other activities. To improve the accessibility and visibility of these events, a new feature could be added to the mobile application that displays all upcoming events in detail.

This feature would allow students to view all the events hosted by the university in one place, with the ability of filter these events based on their interests, preferences, and availability. Each event would be presented with a detailed description, including its date, time, location, and a view overview of its purpose and content.

8.2.6 Filtering

Upon reviewing the 7.2.7 Black Box Testing – User’s Testing questionnaire subchapter, it has come to our attention that the UoWsa application lacks important filtering features. Specifically, the application does not allow for efficient filtering of the friends list, existing forums, and messaging, resulting in a less than optimal user experience.

To address this issue, improvements can be made in the future to the search functionality within the application, enabling users to filter their friends list, existing forums, and messages using a specific keyword search method. This feature would enable users to search for their desired contact quickly and efficiently. Additionally, this function can be used throughout to the existing forums to search for specific forums or messages sent previously.

8.2.7 Explicit content filter

Adding an explicit content filter could highly improve upon social and ethical concerns related to explicit content. The filter would help prevent inappropriate content from being posted on the platform and minimise the workload for administrators.

The explicit content filter would operate by analysing user-generated content, such as, direct messaging, hangout all chat and forums, using artificial intelligence and machine learning algorithms. The algorithm would identify any content that contains explicit language and flag it for review by an administrator. This feature would enable administrators to act quickly to identify and remove any inappropriate content, thereby minimising the risk of exposure to explicit content and its associated social and ethical issues. The explicit content filter would also help to create a safer and more welcoming environment for all academic users of the application, promoting positive social interactions and encouraging responsible behaviour online.

8.2.8 Posting Images

Adding an image upload feature in the messaging area would enable users to share images with another user depending on which messaging service the user is operating on. For instance, users would share pictures of relevant documents or visual aids related to their discussions, or simply share pictures for leisure or personal reasons. To implement this feature, the current implementation must be modified where they would be directed to the device camera or select an image from the user's device gallery.

8.2.9 Add more information in User Profile and Forum Counter

It has been noted in 7.2.7 Black Box Testing – User's Testing questionnaire subchapter, the University of Westminster student mobile application lacks sufficient details in the user profile section. To address this issue, it would be beneficial to expand on the details that can be included in a user's profile to provide a more comprehensive view of their academic and personal background.

The current profile section displays the user's profile picture, username, status, field of study and study location. However, to provide a more detailed view of a user's background, the profile section can be expanded to additional information such as academic achievements, professional experience, personal interests, and hobbies.

By including these additional details, other users can have a better understand of their peers, leading to more productive and meaningful communication and collaboration. For example, if a student is looking for a study partner, they may be more inclined to reach out to someone who shares their interest and hobbies or has similar academic achievements.

8.2.10 Accessibility options

The application lacks accessibility, in particular, there is no provision for users to adjust the font size or type, which can be problematic for users with visual impairments. TO address this issue, it would be beneficial to include accessibility options in the app's settings, allowing users to customise the font size and type, among other features.

By incorporating accessibility options into the app, users with visual impairment or other accessibility needs can adjust the settings to better suit their needs. For instance, users with low vision may require a larger font size to read text comfortably, whilst others may prefer a different font type that is more legible to them. Providing these options in the app's setting swill help to ensure the app is accessible to a wider range of users, regardless of the disabilities or accessibility needs.

8.2.11 Student Life

As an educational institution, the University of Westminster stives to provide its students with not only a high-quality education but also the necessary resources to support succeed in their studies. One key aspect of this is helping student to learn how to manage their time effectively and structure their learning, which can be challenging for many students.

To address this need, it would be beneficial to expand on the current project to include features that can help students time management and structuring their learning. For example, the app could provide tools such as calendars, to-do list, and study planners that can help students to keep track of their assignments, deadlines, and study schedules.

In addition to these time-management tools, the app could also offer resources to help students with specific academic tasks, such as writing different coding languages or preparing for exams. This could include access to online tutorials or study guides, as well as links to external resources such as academic databases. Furthermore, it is important to note that the app could also include features that recommend social aspects of university life further such as extracurricular activities or events.

8.2.12 Notifications

Currently, the notification system on the app is limited to the UoW Message area only. However, according to the feedback from the 7.2.7 Black Box Testing – User's Testing questionnaire subchapter, users require notifications across the app. Therefore, it is necessary to update the app in the future to include notification features for all areas that require them. To achieve this, the app can be updated to use push notifications, which is a feature that enables notifications to be sent to the user's device even when the app is not open. This can be especially useful for urgent updates or time-sensitive information, such as changes to class schedules or upcoming deadlines.

Overall, it was a time constraint issue to implement the notifications in all areas of the application which requires notification, however in app notifications can be demonstrated in the UoW Message area.

8.3 Reflection of the Project

In this subchapter, a detailed reflection of the development project with evaluations on the final outcomes of the project.

8.3.1 Research

In the research stage played a crucial role in guiding the direction of the project. Documenting the problem statement, highlighting the challenges faced by university students in finding peers with common interest and engaging with them. The research also focused on the background review into existing social network platforms for students and their limitations, which included the methodologies to gather user requirements. The initial draft of user requirements and ensuring legal, social, and ethical concerns were addressed, 4.3.1 provided a brief user requirement list which was later refined in the functional and non-functional list of requirements in subchapter 4,3.8.for a clear direction on what design methods must be concluded prior to beginning the implementation phase.

8.3.2 Strengths and Weaknesses of implementation tools

The decision to deliver the project in Kotlin programming language and Firebase Realtime Database for the Android application had several strengths. The Firebase Realtime Database was easy to use and integrate with the application, making it simple to store and retrieve data. The documentation provided from Firebase was detailed, which helped with the integration process. Additionally, the Realtime database provided real-time synchonisation, ensuring that the data was always ip to date across multiple user devices. This was particularly beneficial for the project that require real-time updates, such as messaging.

Kotlin, being a modern and concise language, provided several features that made the development process faster and more efficient. The language is interoperable with Java, making it easy to integrate with Android Studio and Android SDK. Additionally, Kotlin has a wide range of built-in features, such as null safety and extension functions, which helps to reduce errors and improve code readability. This lead to a far stable application with lower probability of crashes or enexpected behaviour.

Android Sudio, being a powerful IDE specifically designed for Android development is another strength of the chosen platform. It integrates well with Android SDK and additional libraries, making it easy to develop this complex application. Furthermore, Android Studio has an intuitive user interface that is easy to navigate and provided relevant listing of features and implementations in explicit areas of the software, thereby making it a organised process to apply specific methods and a range of resources, including plugins and libraries, which makes it easier to build and maintain the quality of the application.

On the other hand, there are some weaknesses associated with the chosen technologies. One of the challenges of using Android Studio and Kotlin was the learning curve associated with the Android architecture components, such as

Recycler view, view model, adapters, and themes. The implementation of these components provides a robust and scalable architecture for the project; however they also require a certain level of expertise to be used effectively and efficiently. In addition, Kotlin is a relatively new language, which means that there are fewer resources or support available compared to more established languages such as Java. Finally, Firebase Realtime database has its own limitations when it comes to complex querying or data structuring, which may require the use of additional tools or databases which can carry out complex or advanced use cases.

8.3.3 Acquisition of new knowledge and skills

During the development of the mobile application, the developer acquired several skills and knowledge. One of the skills was the use of Clip Studio Paint software to create logos, images, and other design aspects of the application. Initially, the developer found it challenging to place images and logos inside the activity due to size and level concerns. However, after learning how to resize and export images tailored for the project, the developer became proficient in using the software to translate into strong UX design.

Another skill acquired was the use of Android Studio Drawable, which allows for the inflation of an image or shape created with XML. Although the process of creating shapes using XML could be challenging at first, it was a rewarding process after the time spent on practicing more.

The developer also learned about physical device testing after encountering issues testing the functionality of the application with an emulator implemented within Android Studio, this provided an alternative to frequent testing.

Firebase Authentication was another skill acquired, which provides an easy-to-use SDK to authenticate users for the application. This includes traditional email and password signups and initially tested another sign-up process in Google login which was swiftly abandoned due to the high number of limitations for free of use for developers. Realtime Database, a NoSQL structure, which provided great features such as real-time synchronisation, scalability, and security. Although this type of database was not taught during the time at university, it was a great opportunity to test and develop using an unfamiliar database structure. On the other hand, one of the greatest challenges faced was connecting the storage and database to work with each other harmoniously, once that hurdle overcame, it was a fantastic integrational across the tools used from Firebase services.

Colours and Themes XML, a way to store and call for a specific colour code, was a newly acquired knowledge which will be extremely useful for future projects. At first, implementing colours to specific objects was only known to the developer. However, during the course of this project, it was self-taught to apply all colours necessary into one file and expand this based on the user’s device theme settings. The light and dark mode implementation can be separated using this technique and overall, it was a crucial skill to obtain for all projects in different programming languages which require front-end designs.

Third party integration of libraries is a technique which is usually not appreciated in all modules the developer has completed previously at the university, this project is also a great indication on why this has been the case. External libraries can often be helpful in a project, however some libraries also come with limitations and disadvantages, this is because a list of ideas and available third-party libraries could be available and it requires strong knowledge and interpretation on what controls the developer could potentially give away using these techniques, therefore the project had very limited amount of third-party libraries used and only libraries which demonstrated its trust-worthy with low to no controls in all aspects of the project was implemented.

8.4 Limitations of the project

This subchapter aims to elucidate the various constraints that may hinder the development of the project. The limitations that could potentially affect the project's progress include, time constraints, testing limitations, and authentication control restriction.

8.4.1 Time Constraint

The project was assigned a deadline of seven months from its inception to draft a proposal, identify the project objectives, conduct research, establish project requirements, design, implementation, test, and deploy the project. This time constraint restricted further refinement of the project.

A clear boundary on the duration of the project, and it became crucial to manage time efficiently to ensure that each task and set milestones were completed on time. The priorities of the project requirements to deliver the essential features was set with a timeframe, this prioritisation process enables concentration on the most critical aspects of the project, avoiding unnecessary features that could have delayed the project's completion. Despite the time constraint, the project had to undergo several iterations to refine the design and ensure that the project met the project requirements. However, the time limitation meant that the number of iterations was restricted, and further refinements could not be made without compromising the project's delivery date.

8.4.2 Testing Limitations

If the mobile application gains popularity on release, it may be necessary to assess its performance under high loads from a large number of users. Testing the application with a simulated user count of 1000 per day would provide valuable insights into how the application responds to increased traffic, particularly in terms of database performance.

However, currently, there is no reliable method to perform such a test. Emulators are typically used to simulate user activity, but running more than three simulators at a time can be challenging due to the computing resources and processing power

cannot handle the task, which will also limit the accuracy of the testing. This means that the application's behaviour in a scenario where there are hundreds of simultaneous users cannot be fully replicated in the testing phase.

In light of these limitations, it is difficult to determine how the application will perform in a real-world environment where hundreds or thousands of users are active at once. The application may experience slow-downs, crashes or other unexpected issues that cannot be fully predicted based on the testing procedures so far. As such, it is important to continue monitoring the application's performance as user numbers increase to detect and address any potential issues as they arise.

8.4.3 Authentication Control Restriction

Firebase authentication is a valuable tool for developers who want to incorporate user authentication swiftly and easily in their application. However, it has certain limitations, especially when it comes to regulating access to the database. Although Firebase Authentication offers robust security measures to guarantee the credibility of user authentication, it lacks flexibility and customisation options that could be used in this project.

One of the principal constraints of Firebase Authentication is the absence of granular control over the database. As a result, the project cannot personalise authentication security to satisfy the precise requirements. For instance, there may be unable to access particular areas of the database based on user rules or permissions discussed in subchapter 7.1.2 Firebase Realtime Database Rules and Permissions.

Another limitation of Firebase authentication is that it relinquishes control to Firebase itself. While this is convenient to other developers who want to concentrate on other aspects of their project, further customisation on the security can provide clear cut demands between security and the database for greater depth in data privacy.

8.5 Conclusion

The development of a social mobile application aimed at improving the social aspects of university life for student has been a challenging yet rewarding experience. The project's primary aim was to address the difficulties faced by students in finding peers with common interest and engaging with them, especially the aftermath of the COVID-19 pandemic. The project's objectives included researching appropriate Kotlin methods and structures to create a mobile application, establishing a successful connection to the database, and researching appropriate software development kits (SDKs) for the project.

One of the significant challenges in developing the mobile application was gathering requirements from students and adults who has been through the university journey in previous years in developing a user-friendly mobile application. To achieve this, an agile methodology was used, which allowed for continuous testing, feedback, and improvement. This helped to ensure that the application met the needs of the users and user-friendly. Additionally, the developer prioritised the privacy and ethical

concerns of the application by complying with General Data Protection Regulation (GDPR) and addressing ethical concerns with the admin feature and other design and implementation methods in place.

The background and literature survey revealed the need for a mobile application tailored for university students. While high profile social media platforms such as Facebook are popular among students, they lack features based on fields of study and interest, making them far less suitable for academic environments. Furthermore, the current availability of social networking for students is incredibly limited, with many developed primarily for educational purposes. This highlighted the space on the market for a mobile application tailored to the needs of university students.

The legal, ethical and security issues associated with the development project were also addressed. The developer ensured that data collect, and analysis were conducted in compliance with ethical standards, and data security was given the utmost priority. Firebase Authentication, Realtime Database, and Data storage were employed to ensure storage and access of user data.

Testing and deployment of the mobile application were also carried out systematically. The testing process included all features implemented, partially implemented, and not implemented, and various white box and black box testing were employed. This ensured that the application met the required standards, and any challenges arose were addressed.

The project’s outcome was a social mobile application that enables students to connect and engage with each other. The mobile application provides a social space where students can communicate through direct messaging and forums with other students and provides information relating to assist students in finding new groups in Clubs and Societies, which is pre-existing from the Student Union. The application also tackles the problems of Student Wellbeing in seeking for help in aiding any problems a student may encounter in or outside of their university life.

The project’s contribution to social aspect of university for students for students cannot be overstated. The COVID-19 pandemic has made it challenging for students to engage in social activities, and the mobile application provides a platform for students to connect with each other and participate in social activities that interest them. The mobile application also addresses the mental heal and wellbeing of students, which is crucial during the pandemic.

In conclusion, the development of the social mobile application aimed at improving the social aspects of university life for students was extremely challenging yet rewarding experience with various skills learnt throughout the academic year, which will likely become useful in the future. Ensuring the prioritisation in privacy and ethical concerns and complied with the General Data Protection Regulation was one of the most difficult aspects to overcome and ensuring the application is engaging with social activities, thereby addressing the mental health and wellbeing of students. In future development after this valuable experience, numerous features used can be widely used for other future development projects.

9. References

Borup, J, Walters. S and Call-Cummings. M [2020] Student Perceptions of Their Interactions with Peers at a Cyber Charter High School

[online]

Available from:

https://files.eric.ed.gov/fulltext/EJ1260358.pdf

[Accessed 25th October 2022]

Berry, S. UniAcco [6th April 2021] The Best Social Networking Sites for Students

[online]

Available from:

https://uniacco.com/blog/best-social-networking-sites-for-students-2021

[Accessed 23rd October 2022]

Ukwishaka, M.C. and Aghaee, N. [2020] Using Social Media for Peer Interaction in Higher Education: Students' Perception of Using Facebook to Support Peer Learning [online]

Available from:

https://www.elearning-conf.org/wp-content/uploads/2020/07/02_202007L014_F073.pdf

[Accessed 25th Oct 2022]

Gikas, J. and Grant, M. M. [2013] Mobile computing devices in higher education: Student perspectives on learning with cellphones, smartphones & social media.

[online]

Available from:

https://www.sciencedirect.com/science/article/abs/pii/S1096751613000262?via%3Dihub

[Accessed 19th April 2023]

Lazarela, L. and Jakimoski, K. [2017] Analysis of the Advantages and Disadvantages of Android and iOS platform and Vice Versa

[online]

Available from:

https://www.researchgate.net/publication/337436321_Analysis_of_the_Advantages_and_Disadvantages_of_Android_and_iOS_Systems_and_Converting_Applications_from_Android_to_iOS_Platform_and_Vice_Versa

[Accessed 18th April 2023]

Han, M. Bootcamp [16th January 2021] Case Study: WhatsApp Redesign
[online]
Available from:
https://bootcamp.uxdesign.cc/ui-ux-case-study-whatsapp-redesign-8101e8fcbdcf
[Accessed 24th October 2022]

Whatsapp [n.d.] About end-to-end encryption
[online]
Available from:

https://faq.whatsapp.com/820124435853543

[Accessed 24th October 2022]

Firebase [n.d.] Firebase Your firebase projects

[online]

Available from:

https://console.firebase.google.com/

[Accessed 14th January 2023]

Bass, R. Terms Feed [1st July 2022] Data Protection Officers and Firebase Under the GDPR

[online]

Available from:

https://www.termsfeed.com/blog/gdpr-firebase-dpo/

[Accessed 14th January 2023]

Clip Studio Tips [n.d.] Clip Studio Paint Official Tips & Tutorials

[online]

Available from:

https://tips.clip-studio.com/en-us/official

[Accessed 14th January 2023]

Grec, Veaceslav. Wordpress [24th March 2012] Drawing Shapes in Android

[online]

Available from:

https://androidresearch.wordpress.com/2012/03/24/drawing-shapes-in-android/

[Accessed 14th January 2023]

Android Developers [n.d.] Run apps on hardware device

[online]

Available from:

https://developer.android.com/studio/run/device

[Accessed 14th January 2023]

Anurina, Olha. MLSDev [12th November 2021] Agile SDLC: Skyrocketing Your Project with Agile Principles

[online]

Available from:

https://developer.android.com/studio/run/device

[Accessed 20th April 2023]

Interaction Design Foundation. [n.d.] What is User Experience (UX) Design?

[online]

https://www.interaction-design.org/literature/topics/ux-design

[Accessed 21st April 2023]

Wesolko, D. Medium. [14th June 2016] Peter Morville's User Experience Honeycomb

[online]

https://danewesolko.medium.com/peter-morvilles-user-experience-honeycomb-904c383b6886

[Accessed 21st April 2023]

Design Rush [n.d.] Best Designs of 2023

[online]

Available from:

https://www.designrush.com/best-designs/apps

[Accessed 14th January 2023]

Al-Rahmi, W. M. and Zeki, A. Research Gate [September 2016] A model of using social media for collaborative learning to enhance learners? Performance on learning

[online]

Available from:

https://www.researchgate.net/publication/308578172_A_Model_of_Using_Social_Media_for_Collaborative_Learning_to_enhance_learners_Performance_on_learning

[Accessed 23rd April 2023]

Barnes, C. and Tynan, B. [September 2007] The adventures of Miranda in the brave new world: learning in a web 2.0 millennium

[online]

Available from:

https://files.eric.ed.gov/fulltext/EJ815338.pdf

[Accessed 23rd April 2023]

Manca, S. Ranieri, M. Wiley Online Library [9th August 2016] Is Facebook still a suitable technology-enhanced learning environment? An updated critical review of the literature from 2012 - 2015

[online]

Available from:

https://onlinelibrary.wiley.com/doi/10.1111/jcal.12154

[Accessed 23rd April 2023]

Stallman, H. M. Scientific Research [9th August 2016] Examining characteristics of resilience among university students: an international study

[online]

Available from:

https://www.scirp.org/(S(vtj3fa45qm1ean45vvffcz55))/reference/ReferencesPapers.aspx?ReferenceID=1341323

[Accessed 23rd April 2023]

Arean, P. A., Hallgren, K. A., Jordan, J. T., Gazzaley, A., Atkins, D. C., Heagerty, P. J. and Anguera, J. A. JMIR Publications [20th December 2016] The Use and Effectiveness of Mobile Apps for Depression: Results from a fully remote clinical trial

[online]

Available from:

https://www.scirp.org/(S(vtj3fa45qm1ean45vvffcz55))/reference/ReferencesPapers.aspx?ReferenceID=1341323

[Accessed 23rd April 2023]

Grieve, R., Indian, M., Witteveen, K., Anne Tolan, G and Marrington, J. Science Direct. [2013] Face-to-face or Facebook: Can social connectedness be derived online?

[online]

Available from:

https://www.sciencedirect.com/science/article/abs/pii/S0747563212003226?via%3Dihub

[Accessed 23rd April 2023]

Google Blogs. [n.d.] The need for mobile speed.

[online]

Available from:

https://blog.google/products/admanager/the-need-for-mobile-speed/
[Accessed 23rd April 2023]

Muchmore, M. PcMag [2nd October 2020] Android vs. iOS: Which mobile OS Is Best?

[online]

Available from:

https://www.pcmag.com/comparisons/android-vs-ios-which-mobile-os-is-best

[Accessed 23rd April 2023]

Firebase Documentation Realtime Database [n.d.] Firebase Realtime Database

[online]

Available from:

https://firebase.google.com/docs/database

[Accessed 23rd April 2023]

CampusGroups [n.d.] CampusGroups the all-in-one campus experience management platform

[online]

Available from:

https://www.campusgroups.com/product/home/

[Accessed 23rd April 2023]

U.S. Department of Education [n.d.] Family Educational Rights and Privacy Act (FERPA)

[online]

Available from:

https://www2.ed.gov/policy/gen/guid/fpco/ferpa/index.html

[Accessed 23rd April 2023]

Android Developers [15th May 2013] Android Studio: An IDE built for Android

[online]

Available from:

https://android-developers.googleblog.com/2013/05/android-studio-ide-built-for-android.html

[Accessed 23rd April 2023]

Karczewski, D. Ideamotive. [30th October 2021] Best Android Development Tools to Use in 2022

[online]

Available from:

https://www.ideamotive.co/blog/best-android-development-tools

[Accessed 24th April 2023]

Developers Android Kotlin. [n.d.] Develop Android Apps with Kotlin

[online]

Available from:

https://developer.android.com/kotlin

[Accessed 24th April 2023]

Wharton, J., Muntenescu, F, and Lau J. [5th February 2018] Introducing Android KTX: Even Sweeter Kotlin Development for Android

[online]

Available from:

https://android-developers.googleblog.com/2018/02/introducing-android-ktx-even-sweeter.html

[Accessed 24th April 2023]

JetBrains Kotlin Census 2018 [n.d.] Kotlin Census 2018

[online]

Available from:

https://www.jetbrains.com/research/kotlin-census-2018/

[Accessed 24th April 2023]

Stack Overflow Developer Survey Results 2019 [n.d.] Developer Survey Results 2019

[online]

Available from:

https://insights.stackoverflow.com/survey/2019#technology-_-most-loved-dreaded-and-wanted-languages

[Accessed 24th April 2023]

JetBrains Kotlin Census 2020 [n.d.] Kotlin Census 2020

[online]

Available from:

https://www.jetbrains.com/lp/kotlin-census-2020/

[Accessed 24th April 2023]

Raj, D. Medium [30th August 2021] Kotlin vs Android Development

[online]

Available from:

https://medium.com/codex/kotlin-vs-java-for-android-development-7cce3e90ed08

[Accessed 24th April 2023]

Statistic Times [27th June 2022] Top Computer Languages

[online]

Available from:

https://statisticstimes.com/tech/top-computer-languages.php

[Accessed 24th April 2023]

Sagara Technology Idea Lab [24th January 2020] What are the benefits of Kotlin

[online]

Available from:

https://sagaratechnology.medium.com/what-are-the-benefits-of-kotlin-d7fdcd1cfc0

[Accessed 24th April 2023]

Ajas. Tech Quintal. [27th September 2022] Advantages and Disadvantages of Kotlin

[online]

Available from:

https://www.techquintal.com/advantages-and-disadvantages-of-kotlin/

[Accessed 24th April 2023]

Firebase Authentication [n.d.] Firebase Authentication

[online]

Available from:

https://firebase.google.com/docs/auth#:~:text=Firebase%20Authentication%20provides%20backend%20services,Facebook%20and%20Twitter%2C%20and%20more.

[Accessed 24th April 2023]

Clark, J. back4app [n.d.] Firebase Advantages and Disadvantages

[online]

Available from:

https://blog.back4app.com/firebase-advantages-and-disadvantages/#:~:text=pricing%20structure%20unacceptable.-,Vendor%20Lock%2DIn,platforms%20when%20the%20need%20arises.

[Accessed 24th April 2023]

Firebase Sync. Firebase [n.d.] Store and sync data in real time

[online]

Available from:

https://firebase.google.com/products/realtime-database

[Accessed 24th April 2023]

Firebase Storage. Firebase [n.d.] Store and serve content with ease

[online]

Available from:

https://firebase.google.com/products/storage

[Accessed 24th April 2023]

Intersoft consulting [1] [2018] Lawfulness of processing

[online]

Available from:

https://gdpr-info.eu/art-6-gdpr/

[Accessed 24th April 2023]

Intersoft consulting [2] [2018] Information to be provided where personal data are collected from the data subject

[online]

Available from:

https://gdpr-info.eu/art-13-gdpr/

[Accessed 24th April 2023]

Intersoft consulting [3] [2018] Notification of a personal data breach to the supervisory authority

[online]

Available from:

https://gdpr-info.eu/art-33-gdpr/

[Accessed 24th April 2023]

Intersoft consulting [4] [2018] Definitions

[online]

Available from:

https://gdpr-info.eu/art-4-gdpr/

[Accessed 24th April 2023]

ico. [n.d.] what is valid consent?

[online]

Available from:

https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/consent/what-is-valid-consent/#:~:text=Consent%20means%20giving%20people%20genuine,consent%20easily%20at%20any%20time

[Accessed 24th April 2023]

Laerd Dissertation [n.d.] Principles of research ethics?

[online]

Available from:

https://dissertation.laerd.com/principles-of-research-ethics.php#:~:text=In%20practice%2C%20these%20ethical%20principles,give%20participants%20the%20right%20to

[Accessed 24th April 2023]

Intersoft consulting [5] [2018] Right to erasure

[online]

Available from:

https://gdpr-info.eu/art-17-gdpr/

[Accessed 24th April 2023]

Oturu, D. Business Day. [9th September 2022] Intellectual property considerations when building a mobile app

[online]

Available from:

https://businessday.ng/news/legal-business/article/intellectual-property-considerations-when-building-a-mobile-app/

[Accessed 24th April 2023]

Gov.uk [9th December 2022] New rules for apps to boost consumer security and privacy

[online]

Available from:

https://www.gov.uk/government/news/new-rules-for-apps-to-boost-consumer-security-and-privacy

[Accessed 24th April 2023]

Morris, C., Scott, R. E., and Mars, M. PubMed. [2018] Security and Other Ethical Concerns of Instant Messaging in Healthcare

[online]

Available from:

https://pubmed.ncbi.nlm.nih.gov/30306960/

[Accessed 24th April 2023]

Unicef. [n.d.] Cyberbullying: What is it and how to stop it?

[online]

Available from:

https://www.unicef.org/end-violence/how-to-stop-cyberbullying

[Accessed 24th April 2023]

Google Cloud [n.d.] Google Cloud & the general data protection regulation GDPR)

[online]

Available from:

https://cloud.google.com/privacy/gdpr

[Accessed 24th April 2023]

Cloudflare [n.d.] What is vendor lock-in? vendor lock-in and cloud computing

[online]

Available from:

https://www.cloudflare.com/en-gb/learning/cloud/what-is-vendor-lock-in/

[Accessed 24th April 2023]

Aditya's Blog [14th November 2022] Firebase: Insecure by Default (feat. That one time our classmates tried to sue us)

[online]

Available from:

https://saligrama.io/blog/post/firebase-insecure-by-default/

[Accessed 24th April 2023]

Miro [n.d.] A simple guide to using and creating a context diagram

[online]

Available from:

https://miro.com/blog/context-diagram/#:~:text=A%20context%20diagram%20outlines%20how,to%20improve%20an%20existing%20system.

[Accessed 15th January 2023]

Association for Computing Machinery [n.d.] ACM Code of Ethics and Professional Conduct

[online]

Available from:

https://www.acm.org/code-of-ethics

[Accessed 24th April 2023]

Hamilton, T. GURU99. [4th March 2023] STLC (Software Testing Life Cycle) Phases, Entry, Exit Criteria

[online]

Available from:

https://www.guru99.com/software-testing-life-cycle.html

[Accessed 24th April 2023]

ico [2] [n.d.] Privacy in mobile apps.

[online]

Available from:

https://ico.org.uk/media/for-organisations/documents/1596/privacy-in-mobile-apps-dp-guidance.pdf

[Accessed 24th April 2023]

Kharychkova, A. Orangesoft. [25th September 2022.] How to publish an android app on google play store

[online]

Available from:

https://orangesoft.co/blog/how-to-publish-an-android-app-on-google-play-store

[Accessed 24th April 2023]

Google Play Developer Policy Center [n.d.] Providing a safe and trusted experience for everyone

[online]

Available from:

https://play.google.com/about/developer-content-policy/

[Accessed 24th April 2023]

Kwon, H. E, Research Gate [November 2016] Excessive dependence on mobile social apps: A rational addiction persepective

[online]

Available from:

https://www.researchgate.net/publication/309689396_Excessive_Dependence_on_Mobile_Social_Apps_A_Rational_Addiction_Perspective

[Accessed 24th April 2023]

Redis Growth Team. Redis [5th January 2022] How to build an app that will tackle social issue by using redis

[online]

Available from:

https://redis.com/blog/how-to-build-social-issues-app/

[Accessed 24th April 2023]

Envato. Envato [10th November 2021] 8 Color Scheme Trends in Mobile App Design

[online]

Available from:

https://www.envato.com/blog/color-scheme-trends-in-mobile-app-design/

[Accessed 13th January 2023]

Visual Paradigm. [1] [n.d.] UML Class Diagram

[online]

https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-class-diagram-tutorial/#:~:text=Association,a%20solid%20line%20between%20classes.

[Accessed 23rd January 2023]

Visual Paradigm. [2] [n.d.] What is Class Diagram?

[online]

https://www.visual-paradigm.com/guide/uml-unified-modeling-language/what-is-class-diagram/

[Accessed 23rd January 2023]

Design Rush [n.d.] Best Designs of 2023

[online]

Available from:

https://www.designrush.com/best-designs/apps

[Accessed 14th January 2023]

StackOverflow [1] [2019] Android Studio not building gradle

[online]

Available from:

https://stackoverflow.com/questions/47441114/android-studio-not-building-gradle

[Accessed 25th March 2023]

StackOverflow [2] [2017] Firebase database security rules

[online]

Available from:

https://stackoverflow.com/questions/37321308/firebase-database-security-rules

[Accessed 25th March 2023]

Checkpoint [n.d.] What Does White Box Testing Focus On?

[online]

Available from:

https://www.checkpoint.com/cyber-hub/cyber-security/what-is-white-box-testing/#:~:text=White%20box%20testing%20is%20a,gray%20and%20black%20box%20testing.

[Accessed 25th March 2023]

Ashtari, H. [29th September 2022] Black Box vs. White Box Testing: Understanding 3 Key Differences

[online]

https://www.spiceworks.com/tech/devops/articles/black-box-vs-white-box-testing/#:~:text=Black%20box%20testing%20is%20a,and%20analyzes%20it%20during%20testing.

[Accessed 25th March 2023]

10. Bibliography

General Reading and Technical Material

Stack Overflow [n.d.] Stack Overflow

[online]

https://stackoverflow.com/

[Accessed 24th January 2023]

Android [n.d.] Android Developer

[online]

https://developer.android.com/docs

[Accessed 24th January 2023]

Firebase [n.d.] Firebase Documents

[online]

https://firebase.google.com/docs

[Accessed 24th January 2023]

Intersoft consulting [n.d.] General Data Protection Regulation GDPR

[online]

https://gdpr-info.eu/

[Accessed 24th January 2023]

Build Fire [n.d.] Mobile App Design: The Complete Designer's Guid | Build Fire

[online]

https://buildfire.com/tools-for-mobile-app-designers/

[Accessed 24th January 2023]

Terms Feed [n.d.] Data Protection Officers and Firebase Under the GDPR

[online]

https://www.termsfeed.com/blog/gdpr-firebase-dpo/

[Accessed 24th January 2023]

Firebase [n.d.] Cloud Messaging

[online]

https://firebase.google.com/docs/cloud-messaging/android/client

[Accessed 24th January 2023]

Synopsys [n.d.] Mobile Application Security

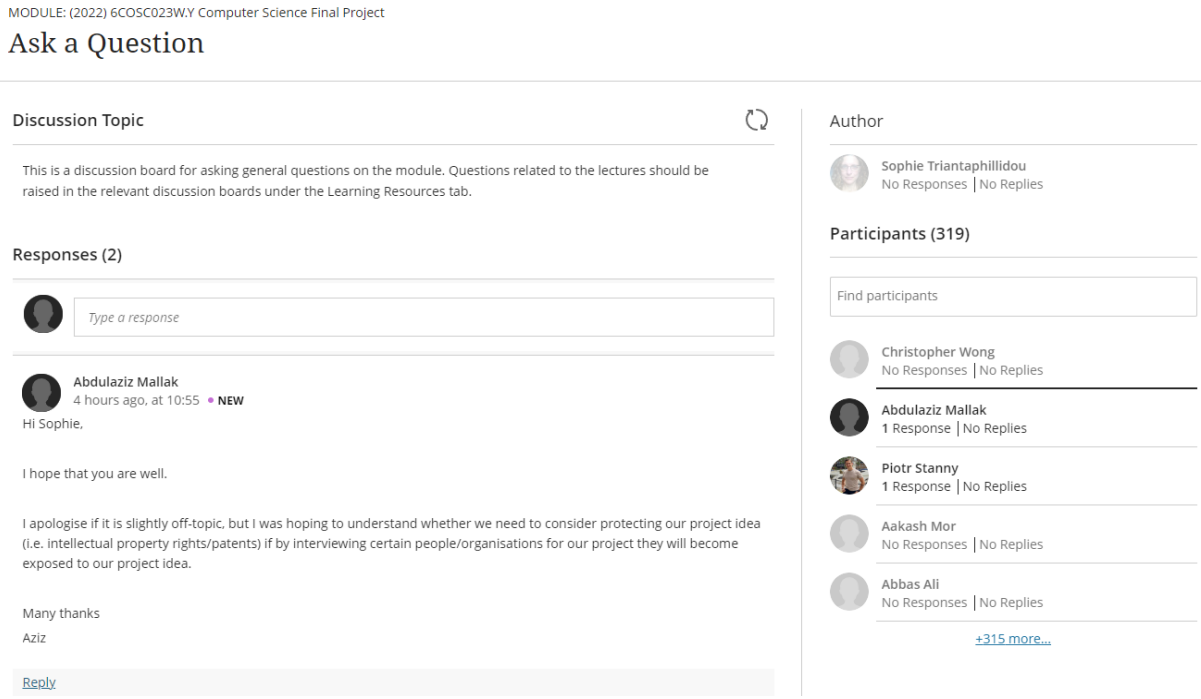
[online]

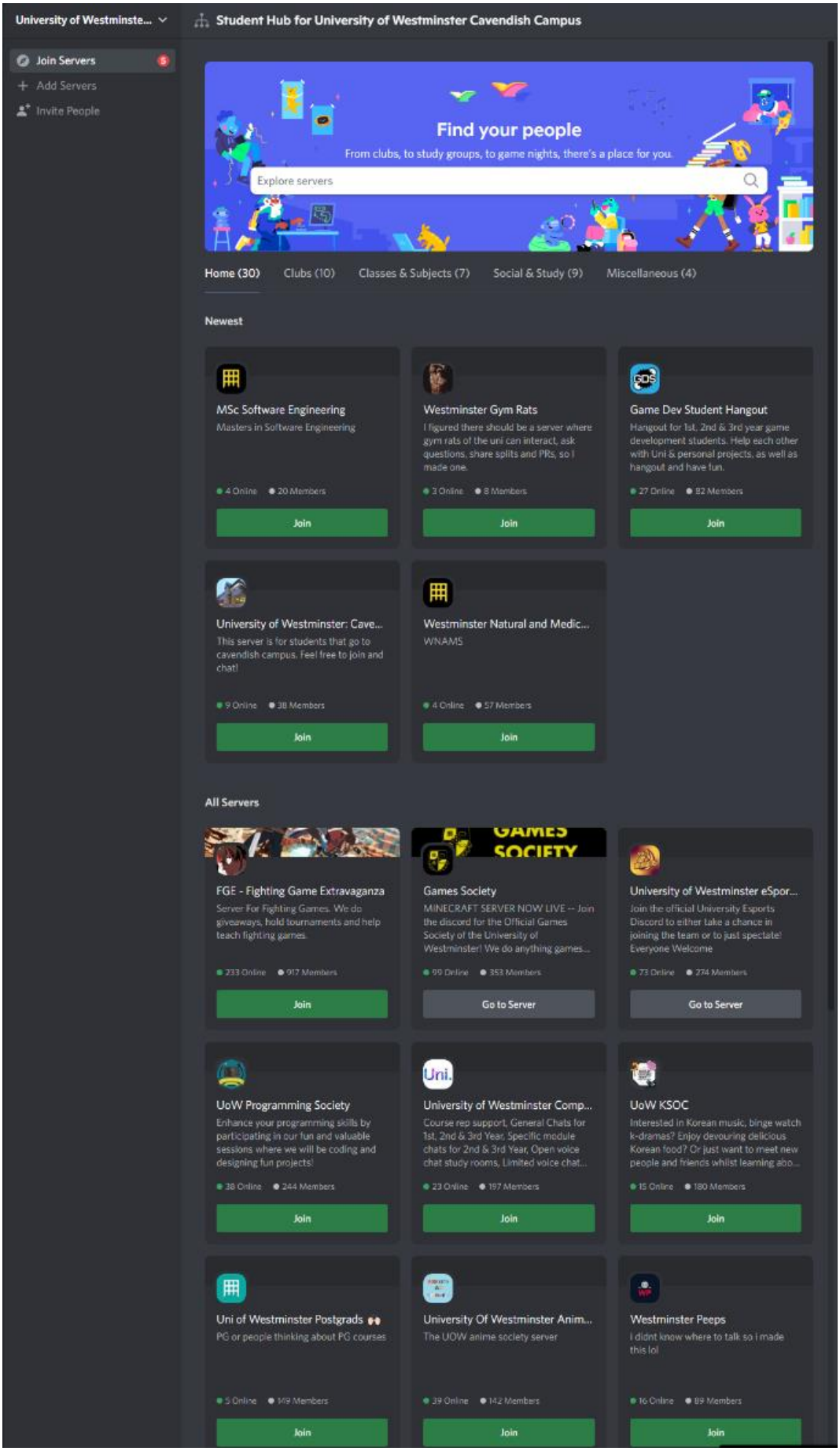
https://www.synopsys.com/glossary/what-is-mobile-application-security.html

[Accessed 24th January 2023]

11. Appendix

Appendix I




University of Westminste...
Student Hub for University of Westminster Cavendish Campus
Join Servers
Add Servers
Invite People
Find your people
From clubs, to study groups, to game nights, there's a place for you.
Explore servers
Home (30)
Clubs (10)
Classes & Subjects (7)
Social & Study (9)
Miscellaneous (4)
Newest
MSc Software Engineering
Masters in Software Engineering
4 Online
20 Members
Join
Westminster Gym Rats
I figured there should be a server where gym rats of the uni can interact, ask questions, share splits and PRs, so I made one.
3 Online
8 Members
Join
Game Dev Student Hangout
Hangout for 1st, 2nd & 3rd year game development students. Help each other with Uni & personal projects, as well as hangout and have fun.
27 Online
82 Members
Join
University of Westminster: Cave...
This server is for students that go to cavendish campus. Feel free to join and chat!
9 Online
38 Members
Join
Westminster Natural and Medic...
WNAMS
4 Online
57 Members
Join
All Servers
FGE - Fighting Game Extravaganza
Server For Fighting Games. We do giveaways, hold tournaments and help teach fighting games.
233 Online
917 Members
Join
Games Society
MINECRAFT SERVER NOW LIVE -- Join the discord for the Official Games Society of the University of Westminster! We do anything games...
99 Online
353 Members
Go to Server
University of Westminster eSpor...
Join the official University Esports Discord to either take a chance in joining the team or to just spectate! Everyone Welcome
73 Online
274 Members
Go to Server
UoW Programming Society
Enhance your programming skills by participating in our fun and valuable sessions where we will be coding and designing fun projects!
38 Online
244 Members
Join
University of Westminster Comp...
Course rep support, General Chats for 1st, 2nd & 3rd Year, Specific module chats for 2nd & 3rd Year, Open voice chat study rooms, Limited voice chat...
23 Online
197 Members
Join
UoW KSOC
Interested in Korean music, binge watch k-dramas? Enjoy devouring delicious Korean food? Or just want to meet new people and friends whilst learning abo...
15 Online
180 Members
Join
Uni of Westminster Postgrads
PG or people thinking about PG courses
5 Online
149 Members
Join
University Of Westminster Anim...
The UOW anime society server
39 Online
142 Members
Join
Westminster Peeps
I didnt know where to talk so i made this lol
16 Online
89 Members
Join

Appendix II

CONSENT FORM

Title of Study: University of Westminster Social Platform Mobile Application for Students

Lead researcher: Christopher Wong

I have been given the Participation Information Sheet and/or had its contents explained to me.
I have had an opportunity to ask any questions and I am satisfied with the answers given.
I understand I have a right to withdraw from the research at any time and I do not have to provide a reason.
I understand that if I withdraw from the research any data included in the results will be removed if that is practicable (I understand that once anonymised data has been collated into other datasets it may not be possible to remove that data).
I would like to receive information relating to the results from this study.
I wish to receive a copy of this Consent form.
I confirm I am willing to be a participant in the above research study.
I note the data collected may be retained in an archive and I am happy for my data to be reused as part of future research activities. I note my data will be fully anonymised (if applicable).

Participant's Name: Christopher Wong

Signature: Christopher Wong **Date:** 25/10/2022
This consent form will be stored separately from any data you provide so that your responses remain anonymous.

_____
_
I confirm I have provided a copy of the Participant Information Sheet approved by the Research Ethics Committee to the participant and fully explained its contents. I have given the participant an opportunity to ask questions, which have been answered.

Researcher's Name: _____

Signature: _____ **Date:** _____

PARENTAL/CARER ASSENT FORM

Appendix III

PARTICIPATION INFORMATION SHEET

Researcher(s): Christopher Wong

Supervisor: Markos Mentzelopoulos

You are being invited to take part in a research study on an android mobile application to investigate the likelihood and usefulness of students socialising among their peers using modern technology of a mobile application in comparison to emails. The aim of the research is to receive feedback from students who communicate with their peers using mobile phones and what are their preferred methods of communication. This research is being undertaken as part of the researcher's studies for Computer Games Development BSc programme at the university and is being carried out in collaboration with Markos Mentzelopoulos.

The study will involve you:

1) Testing the demo version of the project for about 15 minutes.
2) Completing a questionnaire about the mobile application.

Please note:

- Your participation in this research is entirely voluntary.
- You have the right to withdraw at any time without giving a reason.
- Wherever practicable, withdrawal from the research will not affect any treatment and/or services that you receive.
- You have the right to ask for your data to be withdrawn if this is practical, and for personal information to be destroyed.
- You do not have to answer questions either on questionnaires or in interviews if you do not wish to do so.
- Your responses will normally be made anonymous, unless indicated above to the contrary, and will be kept confidential unless you provide explicit consent to do otherwise, for example, the use of your image from photographs and/or video recordings. [NOTE: it may not be possible to maintain confidentiality in certain circumstances, e.g., where issues of child safety have been identified. You should seek clarification from the researcher and/or their supervisor if you are concerned about this].
- No individuals should be identifiable from any collated data, written report of the research, or any publications arising from it.
- All computer data files will be encrypted, and password protected. The researcher will keep files in a secure place and will comply with the requirements of the Data Protection Act.
- All hard copy documents, e.g., consent forms, completed questionnaires, etc. will be kept securely and in a locked cupboard, wherever possible on university premises. Documents may be scanned and stored electronically. This may be done to enable secure transmission of data to the university's secure computer systems.
- If you wish, you can receive information on the results of the research. Please indicate on the consent form if you would like to receive this information.
- The researcher can be contacted during and after participation by email (**w1807769**@my.westminster.ac.uk) or by telephone (07429481111).

If you have a complaint about this research project you can contact the project supervisor, Markos Mentzelopoulos by e-mail (mentzem@my.wesminster.ac.uk).

Appendix IV

University Research Ethics Committee
Front cover sheet for applications

Applicant's name: Christopher Wong

Project Title: University of Westminster Social Platform Mobile Application for Students

Application Reference ___ - ___ - ___ (for office use only)

Please complete the checklist below before submitting your ethics application

Enclosed:	**YES**	**NO**	**N/A**
Application Form Part A attached.	☒	☐	☐
Application Form Part B attached (if applicable).	☐	☐	☐
Any external ethical approval (copy of application and approval letter) attached.	☐	☐	☒
Draft Participant Information Sheet attached (see exemplars).	☒	☐	☐
Draft Informed Consent Form attached (see exemplars).	☒	☐	☐
Draft Indicative Questions, e.g. questionnaire(s), proposed interview questions or questioning areas, etc. attached.	☒	☐	☐
Appropriate risk assessments have been completed, e.g. Control of Substances Hazardous to Health (COSHH), Radiation, etc. (if applicable) – *Contact the University's Safety, Health and Wellbeing Team for advice on this and other aspects of health and safety.*	☐	☐	☒
Fieldwork Risk Assessment attached (if applicable). (*UCEA Guidance on Health and Safety in Fieldwork Including offsite visits and travel in the UK and overseas) – Contact the University's Safety, Health and Wellbeing Team for advice on this and other aspects of health and safety.*	☐	☐	☒
Travel Insurance Request clearance notification attached (if applicable). *Contact - Andrew Clarke (a.clarke03@westminster.ac.uk) or Alison Sylvestre (a.sylvestre@westminster.ac.uk) in Procurement if advice is required – This is essential if there is any Foreign and Commonwealth Office or RED24 advice against travel.*	☐	☐	☒
Confirmation of Insurance coverage for research undertaken off campus.	☐	☐	☒
Security-sensitive research assessment completed (if applicable) and uploaded (see UniversitiesUK Guidance and, if applicable, complete the Annex to Part B and upload).	☐	☐	☒
Other (please specify, e.g. letters from collaborators, etc.):	☐	☐	☒

_____

Applicant's signature: Christopher Wong **Date:** 25/10/2020

Supervisor's or Faculty Research Director's signature: Date:

PLEASE RETURN THIS COVER SHEET ALONG WITH THE REQUIRED DOCUMENTS BY EMAIL ATTACHMENT TO:
SECRETARY, FACULTY RESEARCH ETHICS COMMITTEE (or Contact your Supervisor).

Appendix V
Application for Research Ethics

PART A

Section 1 – PROJECT AND APPLICANT DETAILS

1. **Project Title:** University of Westminster Social Platform Mobile Application for Students

2. **Applicant Details**	
Name: Christopher Wong	**University Email Address:** w18077769@my.westminster.ac.uk
Contact Address: christopher.wong@hotmail.co.uk	**Telephone Number:** 07429481111
Faculty: Computer Science and Technology	
Please check the relevant box: **Undergraduate ☐ Postgraduate ☐ MPhil/PhD Student ☐ Staff ☐**	
I confirm I have read the *University's Code of Practice Governing the Ethical Conduct of Research*	**YES ☐ NO ☐**
1.3 Supervisor/Dean of Faculty/Faculty Research Director details Please note that all applicants with a supervisor(s) must ensure that the supervisor signs the declaration at the bottom of this page if completing Part A only or in **Section 10.3** if completing Part B All **staff** must ensure that their Dean of Faculty, or Faculty Research Director (or nominee), as appropriate, signs the declaration at the bottom of this page if completing Part A only or in **Section 10.3** if completing Part B	
Name: Markos Mentzelopoulos	**University Email Address:** mentzem@my.westminster.ac.uk
Faculty: Computer Science and Technology	**Telephone Number:** Int: 64522

PART A (Continued)

Section 2 – Project Details

2.1 Please provide a description of the background with references to relevant literature (250 words maximum):

The majority of students in higher education use Facebook to communicate with their peers, this is because Facebook has an influence on students to collaborate, share ideas, motivate and learning efficiency. The involvement of students using groups, likes, post, message are the key areas where a student is able to get involved with learning. Students are able to use a social media platform to share videos, books, and documents to their peers to accomplish their goals, learning could vary to hobbies and interests, therefore this incentivises students to get together and learn without a formal setting.

Despite the mass amount of involvement on Facebook, there are limitations as the social platform is developed with the intention of generalised social networking, therefore there is a space in the market specialising in students to develop their skills with their peers. Furthermore, there is a connection block in accessibility, ethical measures, and privacy, which researchers have discovered and should be eliminated (**Ukwishaka and Aghaee**).

Ukwishaka, M.C. and Aghaee, N. 2020 Using Social Media for Peer Interaction in Higher Education: Students' Perception of Using Facebook to Support Peer Learning [online] https://www.elearning-conf.org/wp-content/uploads/2020/07/02_202007L014_F073.pdf [Accessed 25th Oct 2022]

2.2. Please provide a brief description and the aims of your study (250 words maximum):

The aim of this study is to deliver an android mobile application which brings group and direct chat along with discussion board / forums to accommodate the welfare of University of Westminster students. This will also branch into existing clubs and societies from the student union. Furthermore, this can only be achieved with compliance with the General Data Protection Regulation (GDPR) and addressing ethical concerns such as, identity theft, privacy, data security and consent (**Legislation.gov.uk**).

The convenience for students to be able to communicate within common grounds of the university provides a sense of relevance, interests, and peer support. This can lead to a list of personal development, empowerment, acceptance, motivation, and social support for students to help each other (**Admin**).

There has been a limited amount of dedicated university student app which accommodate a social space for students where they can thrive into their interests. There are apps which has these aspects of socialising such as, WhatsApp, Telegram and Discord to name a few, however this mobile application aim is to dive into a student's life and the importance so socialising through communicating with their peers. This could also lead to study groups, hobbies, and interests as well as an introduction to what the university has to offer such as, clubs, societies, and student's wellbeing.

Aims:

- The project is to deliver a mobile application targeting Android which is designed for students at the university of Westminster.
- The mobile application will deliver informational and social aspects for students along with usability, addressing ethical concerns and complies with the GDPR.
- Quality feedback to be gained regarding the student interaction with their peers and the condition of the application.
- The application will include a forum, direct messaging and private messaging and student union section.
- The user will have controls over their choices over their profile.

Legislation.gov.uk. 2018 Data Protection Act 2018 [online]
https://www.legislation.gov.uk/ukpga/2018/12/contents/enacted
[Accessed 25th Oct 2022]

Admin. Teaching Expertise. Students supporting each other [online]
https://www.teachingexpertise.com/articles/students-supporting-each-other/
[Accessed 25th Oct 2022]

2.3. Please outline the design and methodology of your study (include details of the selection and recruitment of participants (if any) and details of any invasive (e.g. blood samples, inhalation/ingestion of food and/or non-food products (in abnormally higher or lower levels than normal or a different form), or intrusive (e.g. questionnaires, focus groups, interviews, etc.) procedures [attach extra information as necessary] (400 words maximum in total):

The use of surveys will allow me to find and tackle the any problems and additional ideas survey participations may have. All survey participants will be from the University of Westminster and will be over the age of 18. There will be several prototypes of the mobile application, once it's presented to the participants, they will be able to give feedbacks for further adjustments before completion.

The first prototype will be available approximately the beginning of the second semester and a survey will be provided to participants for testing and feedback.

2.4. Timescales

Start Date (DD/MM/YY): 15/10/2022

Estimated duration of work: 7 months

Appendix VI

- Account has the meaning given in the Agreement or, if no such meaning is given, means Customer's account for the Services.

- Additional Product means a product, service or application provided by Google or a third party that: (a) is not part of the Services; and (b) is accessible for use within the user interface of the Services or is otherwise integrated with the Services.
- Additional Security Controls means security resources, features, functionality and/or controls that Customer may use at its option and/or as it determines, including the Admin Console and other features and/or functionality of the Services such as logging and monitoring and identity and access management.
- Adequate Country means:

  (a) for data processed subject to the EU GDPR: the EEA, or a country or territory recognized as ensuring adequate data protection under the EU GDPR;

  (b) for data processed subject to the UK GDPR: the UK or a country or territory recognized as ensuring adequate data protection under the UK GDPR and the Data Protection Act 2018; and/or

  (c) for data processed subject to the Swiss FDPA: Switzerland, or a country or territory that is (i) included in the list of the states whose legislation ensures adequate protection as published by the Swiss Federal Data Protection and Information Commissioner, or (ii) recognized as ensuring adequate data protection under the Swiss FDPA,

  in each case, other than on the basis of an optional data protection framework.
- Admin Console has the meaning given in the Agreement or, if no such meaning is given, means the online console(s) and/or tool(s) provided by Google to Customer for administering the Services.
- Affiliate has the meaning given in the Agreement or, if no such meaning is given, means any entity that directly or indirectly controls, is controlled by, or is under common control with, a party.
- Alternative Transfer Solution means a solution, other than SCCs, that enables the lawful transfer of personal data to a third country in accordance with European Data Protection Law, for example a data protection framework recognized as ensuring that participating local entities provide adequate protection.
- Audited Services means the then-current Services indicated as being in-scope for the relevant certification or report at https://firebase.google.com/support/privacy/#certifications, as may be updated by Google from time to time.
- Customer Data has the meaning given in the Agreement or, if no such meaning is given, means data provided by or on behalf of Customer or Customer End Users via the Services (except TSS and any other support services, if applicable) under the Account.

- Customer End Users means the individuals who are permitted by Customer to use the Services. For clarity, Customer End Users may include employees of Customer Affiliates and other authorized third parties.
- Customer Personal Data means the personal data contained within the Customer Data, including any special categories of personal data defined under European Data Protection Law.
- Customer SCCs means the SCCs (Controller-to-Processor), the SCCs (Processor-to-Processor), and/or the SCCs (Processor-to-Controller), as applicable.
- Data Incident means a breach of Google's security leading to the accidental or unlawful destruction, loss, alteration, unauthorized disclosure of, or access to, Customer Data on systems managed by or otherwise controlled by Google.
- European Data Protection Law means, as applicable: (a) the GDPR; and/or (b) the Swiss FDPA.
- EEA means the European Economic Area.
- EU GDPR means Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data, and repealing Directive 95/46/EC.
- European Law means, as applicable: (a) EU or EU Member State law (if the EU GDPR applies to the processing of Customer Personal Data); and (b) the law of the UK or a part of the UK (if the UK GDPR applies to the processing of Customer Personal Data).
- GDPR means, as applicable: (a) the EU GDPR; and/or (b) the UK GDPR.
- Google's Third Party Auditor means a Google-appointed, qualified and independent third party auditor, whose then-current identity Google will disclose to Customer.
- Instructions has the meaning given in Section 5.2.1 (Customer’s Instructions).
- ISO 27001 Certification means an ISO/IEC 27001:2013 certification or a comparable certification for the Audited Services.
- Non-European Data Protection Law means data protection or privacy laws in force outside the EEA, Switzerland, and the UK.
- Notification Email Address means the email address(es) designated by Customer in the Admin Console to receive certain notifications from Google. Customer is responsible for using the Admin Console to ensure that its Notification Email Address remains current and valid.
- SCCs means the Customer SCCs and/or SCCs (Processor-to-Processor, Google Exporter), as applicable.
- SCCs (Controller-to-Processor) means the terms at: https://firebase.google.com/terms/firebase-sccs-eu-c2p

- SCCs (Processor-to-Controller) means the terms at: https://firebase.google.com/terms/firebase-sccs-eu-p2c
- SCCs (Processor-to-Processor) means the terms at: https://firebase.google.com/terms/firebase-sccs-eu-p2p
- SCCs (Processor-to-Processor, Google Exporter) means the terms at: https://firebase.google.com/terms/firebase-sccs-eu-p2p-google-exporter
- Security Documentation means all documents and information made available by Google under Section 7.5.1 (Reviews of Security Documentation).
- Security Measures has the meaning given in Section 7.1.1 (Google's Security Measures).
- Services has the meaning given to "Paid Services", "APIs" or "Services" (as applicable) in the Agreement.
- SOC 2 Report means a confidential Service Organization Control (SOC) 2 report (or a comparable report) on Google's systems examining logical security controls, physical security controls, and system availability, as produced by Google's Third Party Auditor in relation to the Audited Services.
- Subprocessor means a third party authorized as another processor under these Terms to have logical access to and process Customer Data in order to provide parts of the Services and TSS (if applicable).
- Supervisory Authority means, as applicable: (a) a "supervisory authority" as defined in the EU GDPR; and/or (b) the "Commissioner" as defined in the UK GDPR and/or the Swiss FDPA.
- Swiss FDPA means the Federal Data Protection Act of 19 June 1992 (Switzerland).
- Term means the period from the Terms Effective Date until the end of Google's provision of the Services, including, if applicable, any period during which provision of the Services may be suspended and any post-termination period during which Google may continue providing the Services for transitional purposes.
- Terms Effective Date means the date on which Customer accepted, or the parties otherwise agreed to, these Terms.
- TSS means technical support services that Google has agreed to provide to Customer under an agreement that incorporates the Firebase Technical Support Services Guide available at https://cloud.google.com/terms/tssg/firebase/.
- UK GDPR means the EU GDPR as amended and incorporated into UK law under the UK European Union (Withdrawal) Act 2018, and applicable secondary legislation made under that Act.

Appendix VII

END USERS LICENCE AGREEMENT

Last updated 01, February 2023

UoWsa is licensed to You (End-User) by UoWsa for use only under the terms of this License agreement.

By signing up the Licensed Application from Google's software distribution platform ("Play Store"), and any update thereto (as permitted by this Licence Agreement) and You accept this Licence Agreement. Play Store is referred to this Licence Agreement as "Services". The licensor reserves all rights not expressly granted to You. UoWsa is to be used on devices that operate with Google's operating system ("Android").

The Licensor is solely responsible for providing any maintenance and support services for this Licensed Application. You can reach the Licensor at the email address listed in the Play Store Overview for this application.

UoWsa and End-User acknowledge the services have no obligation whatsoever to furnish any maintenance and support service with respect to the licensed application.

The licensed application may invite you to chat, contribute to, or participate in discussions with the opportunity to create, submit, post, display, transmit, perform, and distribute materials to the Licensed Application. This includes but not limited to text, writings, animations, photographs, comments, suggestions or personal information or other material. Contributions may be viewable by other users of the Licensed Application. As such, any contributions transmitted by you may be treated as non-confidential and non-proprietary. When you create or make available any contributions, you thereby represent and warrant that:

1. The creation, distribution, transmission, public display, performance, and the accessing, downloading, or copying of your contributions do not and will not infringe the proprietary right, including but not limited to the copyright, patent, trademark, trade secret or moral rights of any third party.
2. You are the creator and owner of or have the necessary licences, rights, consents, releases, and permissions to use and authorise us the, the Licensed application and other users of the licensed application to use your contributions in any manner contemplated by the Licensed Agreement.
3. You have written consent, release, and/or permission of each and every identifiable individual person in your contributions to use the name or likeness or each and every such identifiable individual application and this licence agreement.
4. Your contributions are not false, inaccurate, or misleading.

5. Your contributions are not unsolicited or unauthorised advertising, promotional materials, pyramid schemes, letter chains, spam, mass mailing, or other forms of solicitation.
6. Your contributions are not obscene, lewd, lascivious, filthy, violet, harassing, libellous, slanderous, or otherwise objectionable (as determined by us).
7. Your contributions do not ridicule, mock, disparage, intimidate, or abuse anyone.
8. Your contributions are not sued to harass or threaten (in the legal sense of those terms) any other person and to promote violence against a specific person or class of people.
9. Your contributions do not violate any applicable law, regulation, or rule.
10. Your contributions do not violate the privacy or publicity rights of any third party.
11. Your contributions do not violate any applicable law concerning child pornography, or otherwise intend to protect the health or well-being of minors.
12. Your contributions do not include any offense comments that are connected to race, national origin, gender, sexual preference, or physical handicap.
13. Your contributions do not otherwise violate, or link to material that violates any provision of this Licence agreement or any applicable law or regulation.

Any use of the Licenced Application in violation of the foregoing violates this Licence Agreement and many result in, among other things, termination, or suspension of your rights to use the Licensed Application.

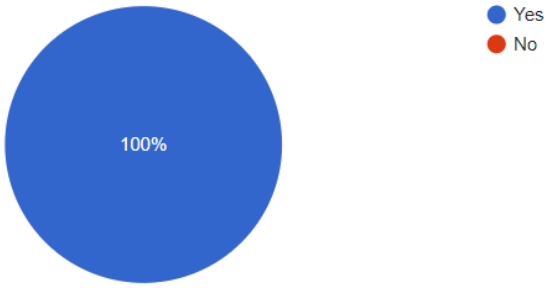
Appendix VIII

Concent Form

The participation information was explained to me.

Copy

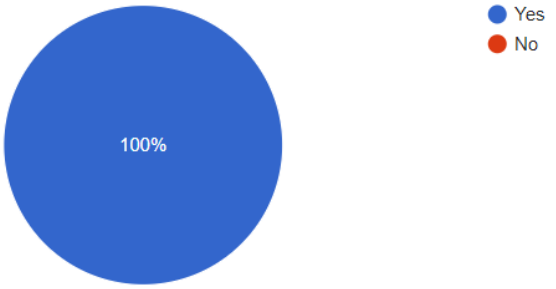
24 responses



I understand if I wish to withdraw my participation, the information I have provided will be permanently removed.

Copy

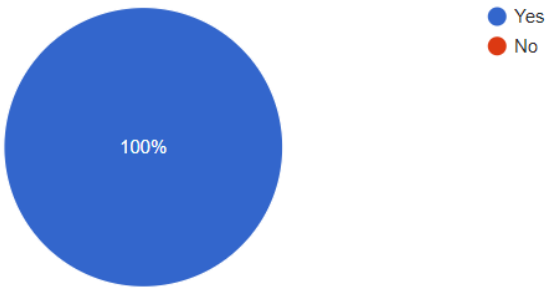
24 responses



I understand on submission the data provided cannot be removed and will be used by the lead researcher.

Copy

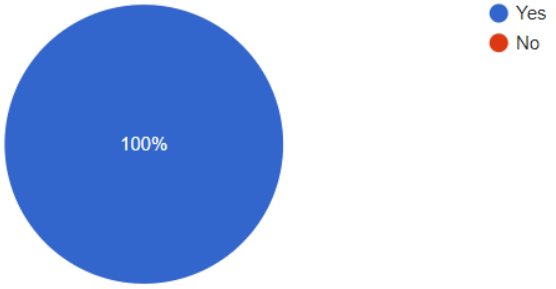
24 responses



I understand the information provided for this project will be treated with confidentiality.

Copy

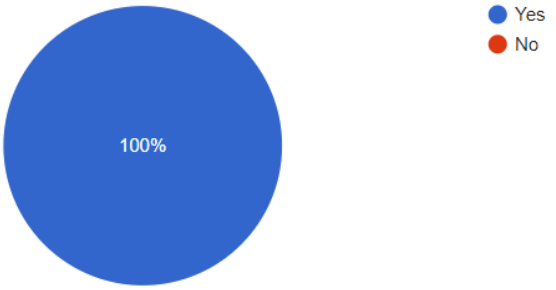
24 responses



I have the opportunity to ask the lead researcher any question regarding the research study and the project under development.

Copy

24 responses



Appendix IX

Timestamp	Consent Question	Consent Question	Consent Question	Consent Question	Consent Question	Question 1	Question 2	Question 3
12/11/2022 23:33:11	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Online
12/11/2022 23:33:29	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Online
12/11/2022 23:35:06	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Both
12/11/2022 23:39:45	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Online
12/12/2022 3:29:54	Yes	Yes	Yes	Yes	Yes	No	Yes	Physically Meet
12/12/2022 3:30:44	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Online
12/12/2022 3:46:08	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Online
12/12/2022 3:49:10	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Online
12/12/2022 3:51:19	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Online
12/12/2022 4:12:37	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Both
12/12/2022 4:48:06	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Online
12/12/2022 5:10:23	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Online
12/12/2022 5:33:47	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Online
12/12/2022 5:53:41	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Online
12/12/2022 8:23:10	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Online
12/12/2022 8:27:38	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Online
12/12/2022 12:09:54	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Online
12/12/2022 20:59:51	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Both
12/14/2022 2:59:37	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Both! It's good to have the options to remain in touch with peers online when in-person becomes difficult to schedule/plan out.
12/23/2022 7:14:49	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Online
12/23/2022 7:15:42	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Online
12/23/2022 7:16:35	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Online
12/23/2022 7:17:13	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Online
12/24/2022 12:49:43	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Physically Meet

Question 4	Question 5
No	Finding common grounds with fellow students, Student Life, Learning
No	Study Group, Making New Friends, Finding common grounds with fellow students, Student Life, Learning
No	Study Group, Making New Friends, Student Life
No	Making New Friends, Learning more about the university, Student Life, Learning
Yes	Making New Friends, Student Life, All of the Above
No	Making New Friends, Finding common grounds with fellow students
No	Study Group, Making New Friends, Student Life, Learning
Yes	Study Group, Making New Friends
Yes	All of the Above
Yes	Learning
Yes	Learning more about the university, Finding common grounds with fellow students, Student Life, Learning
Yes	Study Group, Making New Friends, Finding common grounds with fellow students, Student Life, Learning
Yes	All of the Above
Yes	Study Group, Student Life, Learning
No	Study Group, Making New Friends, Finding common grounds with fellow students, Student Life
No	Study Group, Making New Friends, Finding common grounds with fellow students, Student Life, Learning
Yes	Study Group, Learning more about the university, Student Life, Learning
No	All of the Above
No	None of the Above
No	All of the Above
Yes	All of the Above
Yes	All of the Above
Yes	All of the Above
Yes	Study Group, Making New Friends, Learning more about the university, Finding common grounds with fellow students, Student Life, Learning, All of the Above

Question 6	Question 7
All of the Above	Yes
Direct Messaging, Forum / Discussion Area, Maps for campus location, Subjects available for Undergraduates and Postgraduates, Student Wellbeing, Student Union, Events, About us	Yes
Direct Messaging, Forum / Discussion Area, Maps for campus location, Accommodation, Subjects available for Undergraduates and Postgraduates, Student Wellbeing, Events	No
All of the Above	Yes
Direct Messaging, Student Life, Student Wellbeing, Student Union, Events	No
Direct Messaging, Events	Yes
Direct Messaging, Forum / Discussion Area, Accommodation, Student Wellbeing, About us	Yes
Direct Messaging, Forum / Discussion Area, Events	No
Forum / Discussion Area, Maps for campus location, Student Life, Accommodation, Events, About us	Yes
All of the Above	Yes
Forum / Discussion Area, Student Life, Subjects available for Undergraduates and Postgraduates, Events	Yes
Direct Messaging, Forum / Discussion Area, Subjects available for Undergraduates and Postgraduates, Student Wellbeing, Student Union, Events, About us	Yes
All of the Above	Yes
Direct Messaging, Forum / Discussion Area, Student Life, Student Union	Yes
All of the Above	Yes
All of the Above	Yes
Direct Messaging, Maps for campus location, Accommodation, Subjects available for Undergraduates and Postgraduates, Student Union, Events	No
Direct Messaging, Forum / Discussion Area, Maps for campus location, Events	Yes
Direct Messaging, Forum / Discussion Area, Maps for campus location, Accommodation, Student Wellbeing, Events	Yes
Direct Messaging, Forum / Discussion Area, Maps for campus location, Student Wellbeing, Student Union	Yes
All of the Above	Yes
All of the Above	Yes
All of the Above	Yes
Direct Messaging, Forum / Discussion Area, Accommodation, Subjects available for Undergraduates and Postgraduates, Events, About us, All of the Above	Yes

Question 8
The ability to safely send and receive messages between two users, To add multiple users from direct messaging to group messaging
The ability to safely send and receive messages between two users, To add multiple users from direct messaging to group messaging, Direct messaging should only be available when a friend request is sent and accepted, Gifs and Emojis are available
All of the Above
The ability to safely send and receive messages between two users, To add multiple users from direct messaging to group messaging, Direct messaging should only be available when a friend request is sent and accepted, Gifs and Emojis are available
To add multiple users from direct messaging to group messaging, Direct messaging should only be available when a friend request is sent and accepted
The ability to safely send and receive messages between two users, Direct messaging should only be available when a friend request is sent and accepted
The ability to safely send and receive messages between two users
The ability to safely send and receive messages between two users, To add multiple users from direct messaging to group messaging, Gifs and Emojis are available
All of the Above
All of the Above
The ability to safely send and receive messages between two users, To add multiple users from direct messaging to group messaging
The ability to safely send and receive messages between two users, To add multiple users from direct messaging to group messaging, Accessibility options (font, size) of text, Gifs and Emojis are available, All of the Above
All of the Above
The ability to safely send and receive messages between two users, To add multiple users from direct messaging to group messaging, Direct messaging should only be available when a friend request is sent and accepted, Gifs and Emojis are available
All of the Above
The ability to safely send and receive messages between two users, Gifs and Emojis are available
The ability to safely send and receive messages between two users, To add multiple users from direct messaging to group messaging, Direct messaging should only be available when a friend request is sent and accepted
All of the Above
The ability to safely send and receive messages between two users, To add multiple users from direct messaging to group messaging, Accessibility options (font, size) of text, All of the Above
The ability to safely send and receive messages between two users, To add multiple users from direct messaging to group messaging, Direct messaging should only be available when a friend request is sent and accepted
The ability to safely send and receive messages between two users, To add multiple users from direct messaging to group messaging, Direct messaging should only be available when a friend request is sent and accepted, Gifs and Emojis are available
All of the Above
All of the Above
The ability to safely send and receive messages between two users, To add multiple users from direct messaging to group messaging, Gifs and Emojis are available

Question 9 Rank 1	Question 9 Rank 2	Question 9 Rank 3
Profile Name	Profile Image	Connectivity State (Online, Offline, Away, Busy, Appearing Offline)
Profile Name	Profile Image	Forum Post Counter
Connectivity State (Online, Offline, Away, Busy, Appearing Offline)	Profile Image	Profile Name
Profile Name	Profile Image	User Custom Status
Profile Name	Profile Image	Connectivity State (Online, Offline, Away, Busy, Appearing Offline)
Profile Name	Connectivity State (Online, Offline, Away, Busy, Appearing Offline)	Profile Image
Profile Name	Profile Image	User Custom Status
Profile Name	Profile Image	Connectivity State (Online, Offline, Away, Busy, Appearing Offline)
User Custom Status	Profile Image	Connectivity State (Online, Offline, Away, Busy, Appearing Offline)
Forum Post Counter	Connectivity State (Online, Offline, Away, Busy, Appearing Offline)	User Custom Status
Profile Name	Profile Image	Connectivity State (Online, Offline, Away, Busy, Appearing Offline)
Connectivity State (Online, Offline, Away, Busy, Appearing Offline)	Profile Name	Profile Image
Profile Name	Profile Image	User Custom Status
Profile Name	Profile Image	User Custom Status
Profile Image	Profile Name	Connectivity State (Online, Offline, Away, Busy, Appearing Offline)
Profile Name	User Custom Status	Connectivity State (Online, Offline, Away, Busy, Appearing Offline)
Profile Name	Connectivity State (Online, Offline, Away, Busy, Appearing Offline)	Profile Image
Profile Name	Profile Image	Forum Post Counter
Profile Name	Profile Image	Connectivity State (Online, Offline, Away, Busy, Appearing Offline)
Profile Name	Profile Image	Connectivity State (Online, Offline, Away, Busy, Appearing Offline)
Profile Name	Profile Image	User Custom Status
Profile Name	Profile Image	User Custom Status
Profile Name	Profile Image	Connectivity State (Online, Offline, Away, Busy, Appearing Offline)
Profile Name	Profile Image	Connectivity State (Online, Offline, Away, Busy, Appearing Offline)

Question 9 Rank 4	Question 9 Rank5	Question 10 Rank 1
Forum Post Counter	User Custom Status	Add, Remove and Block other users
User Custom Status	Connectivity State (Online, Offline, Away, Busy, Appearing Offline)	Add, Remove and Block other users
User Custom Status	Forum Post Counter	Add, Remove and Block other users
Forum Post Counter	Connectivity State (Online, Offline, Away, Busy, Appearing Offline)	Add, Remove and Block other users
User Custom Status	Forum Post Counter	Allow Direct Messaging from users who are not added as friends
User Custom Status	Forum Post Counter	Add, Remove and Block other users
Connectivity State (Online, Offline, Away, Busy, Appearing Offline)	Forum Post Counter	Allow Direct Messaging from users who are not added as friends
User Custom Status	Forum Post Counter	Add, Remove and Block other users
Forum Post Counter	Profile Name	Allow Direct Messaging from users who are not added as friends
Profile Name	Profile Image	Light and Dark Mode
User Custom Status	Forum Post Counter	Add, Remove and Block other users
Forum Post Counter	User Custom Status	Light and Dark Mode
Connectivity State (Online, Offline, Away, Busy, Appearing Offline)	Forum Post Counter	Add, Remove and Block other users
Connectivity State (Online, Offline, Away, Busy, Appearing Offline)	Forum Post Counter	Add, Remove and Block other users
User Custom Status	Forum Post Counter	Light and Dark Mode
Profile Image	Forum Post Counter	Allow Direct Messaging from users who are not added as friends
User Custom Status	Forum Post Counter	Add, Remove and Block other users
User Custom Status	Connectivity State (Online, Offline, Away, Busy, Appearing Offline)	Light and Dark Mode
User Custom Status	Forum Post Counter	Add, Remove and Block other users
User Custom Status	Forum Post Counter	Add, Remove and Block other users
Connectivity State (Online, Offline, Away, Busy, Appearing Offline)	Forum Post Counter	Add, Remove and Block other users
Connectivity State (Online, Offline, Away, Busy, Appearing Offline)	Forum Post Counter	Add, Remove and Block other users
Forum Post Counter	User Custom Status	Add, Remove and Block other users
Forum Post Counter	User Custom Status	Add, Remove and Block other users

Question 10 Rank 2	Question 10 Rank 3	Question 11
Allow Direct Messaging from users who are not added as friends	Light and Dark Mode	Navy and Vanilla [current prototype] (figure 1)
Light and Dark Mode	Allow Direct Messaging from users who are not added as friends	Navy and Vanilla [current prototype] (figure 1)
Allow Direct Messaging from users who are not added as friends	Light and Dark Mode	Blue and White
Allow Direct Messaging from users who are not added as friends	Light and Dark Mode	Pink
Light and Dark Mode	Add, Remove and Block other users	Blue and Pink
Allow Direct Messaging from users who are not added as friends	Light and Dark Mode	Blue and Pink
Add, Remove and Block other users	Light and Dark Mode	Navy and Vanilla [current prototype] (figure 1)
Allow Direct Messaging from users who are not added as friends	Light and Dark Mode	Blue and White
Light and Dark Mode	Add, Remove and Block other users	Mustard, Sage & Forest Green
Add, Remove and Block other users	Allow Direct Messaging from users who are not added as friends	Blue and Pink
Allow Direct Messaging from users who are not added as friends	Light and Dark Mode	Blue and Pink
Allow Direct Messaging from users who are not added as friends	Add, Remove and Block other users	Blue and White
Allow Direct Messaging from users who are not added as friends	Light and Dark Mode	Navy and Vanilla [current prototype] (figure 1)
Light and Dark Mode	Allow Direct Messaging from users who are not added as friends	Navy and Vanilla [current prototype] (figure 1)
Add, Remove and Block other users	Allow Direct Messaging from users who are not added as friends	Blue and Pink
Add, Remove and Block other users	Light and Dark Mode	Blue and White
Allow Direct Messaging from users who are not added as friends	Light and Dark Mode	Blue and Pink
Add, Remove and Block other users	Allow Direct Messaging from users who are not added as friends	Royal Blue and Peach
Light and Dark Mode	Allow Direct Messaging from users who are not added as friends	Blue and Pink
Light and Dark Mode	Allow Direct Messaging from users who are not added as friends	Navy and Vanilla [current prototype] (figure 1)
Allow Direct Messaging from users who are not added as friends	Light and Dark Mode	Navy and Vanilla [current prototype] (figure 1)
Light and Dark Mode	Allow Direct Messaging from users who are not added as friends	Navy and Vanilla [current prototype] (figure 1)
Allow Direct Messaging from users who are not added as friends	Light and Dark Mode	Navy and Vanilla [current prototype] (figure 1)
Allow Direct Messaging from users who are not added as friends	Light and Dark Mode	Navy and Vanilla [current prototype] (figure 1)

Appendix X

What colours do you prefer when using this mobile application? (If other, choose 2 or more colours) *

- ○ Navy and Vanilla [current prototype] (figure 1)

- ○ Blue and Pink

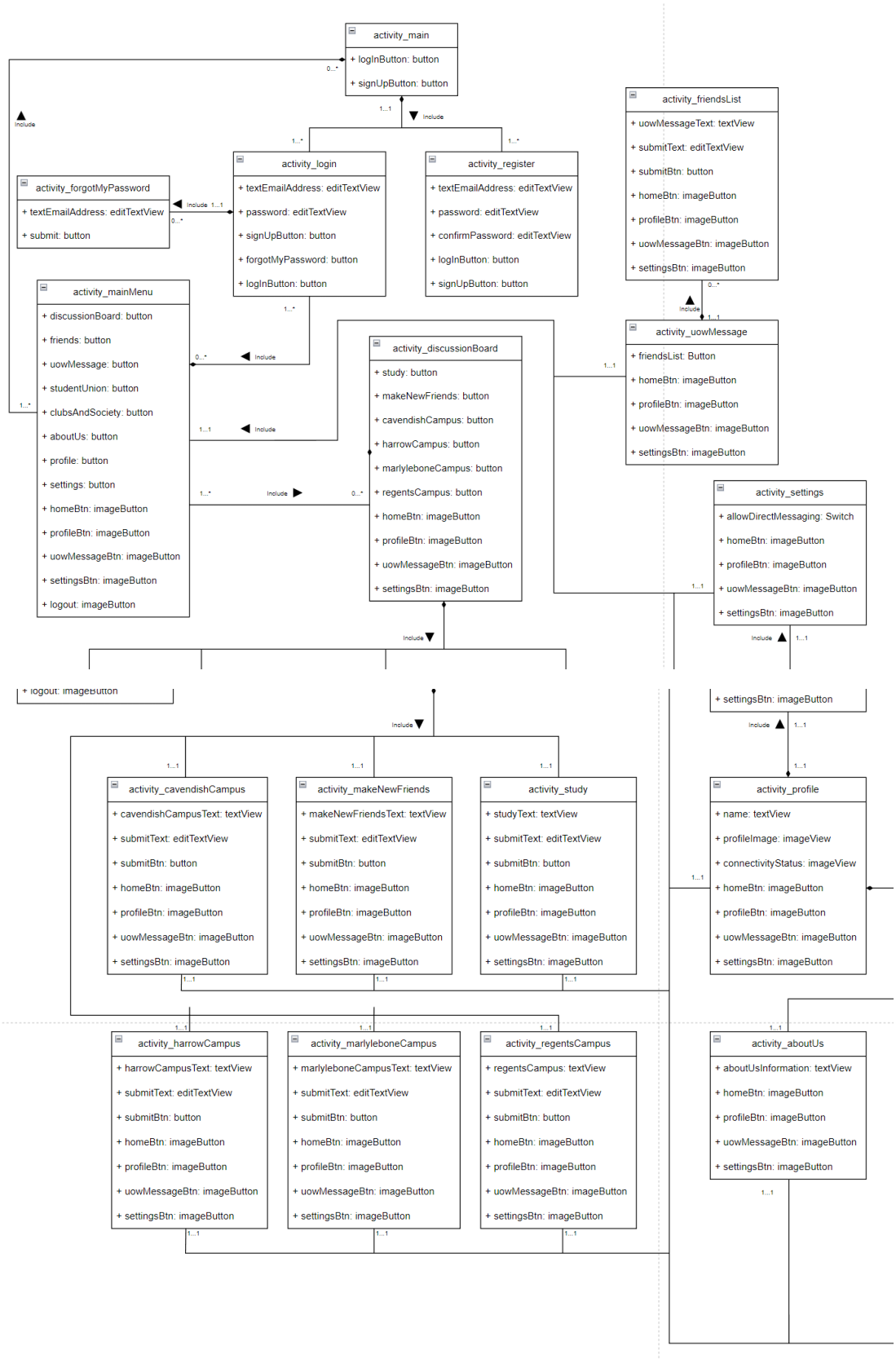
- ○ Royal Blue and Peach

- ○ Blue and White

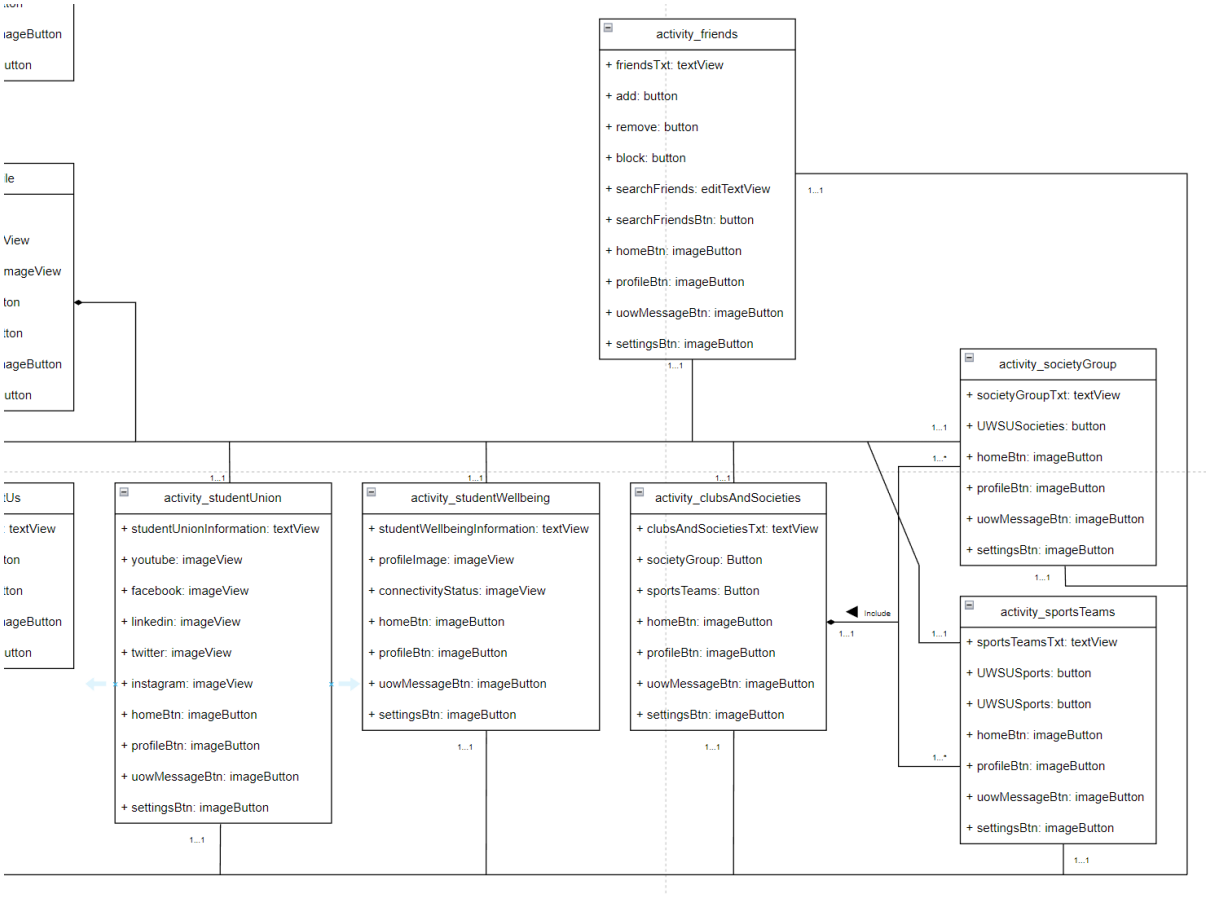
- ○ Mustard, Sage & Forest Green

- ○ Other...

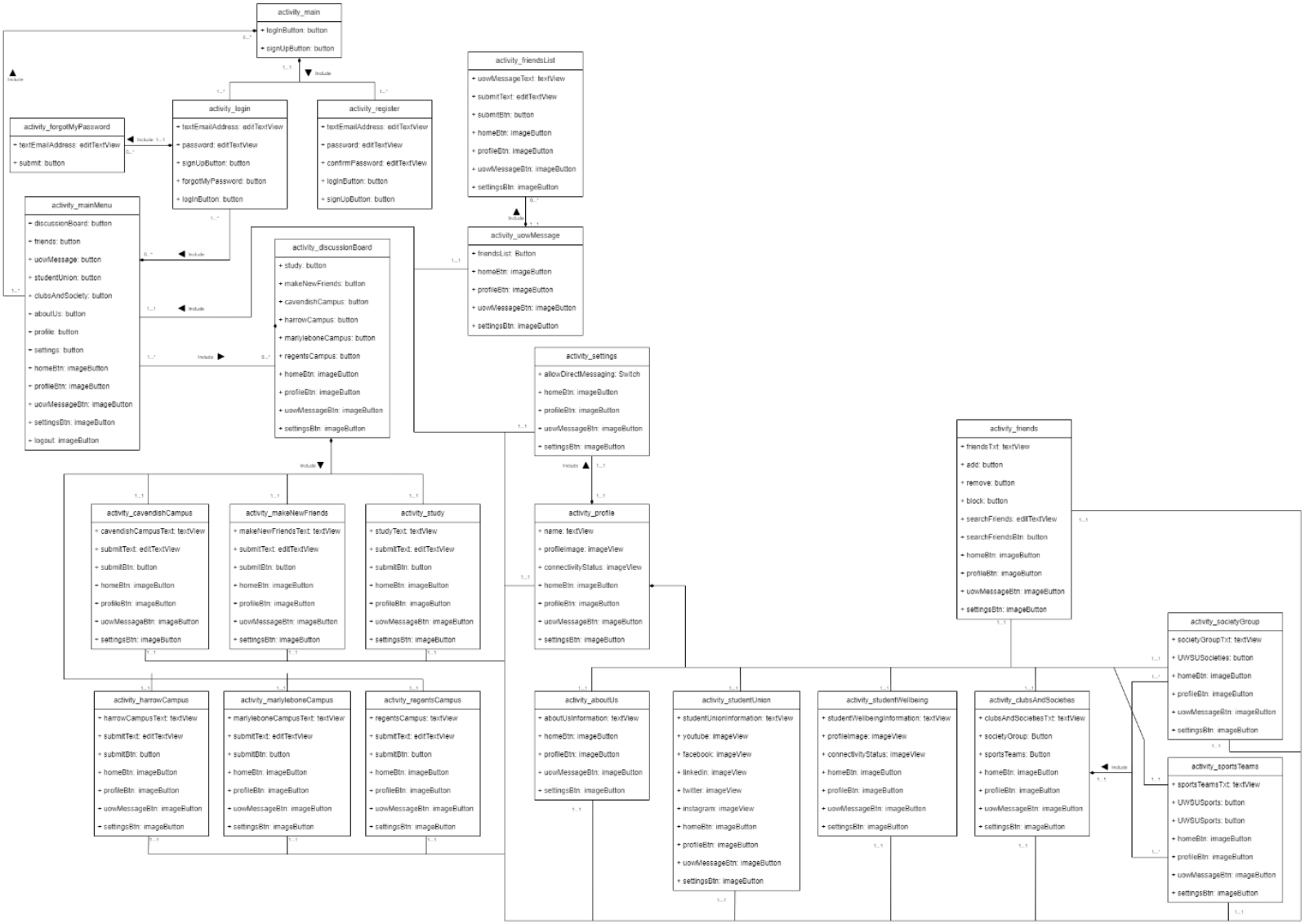
Appendix XI



Class Diagram Part 1

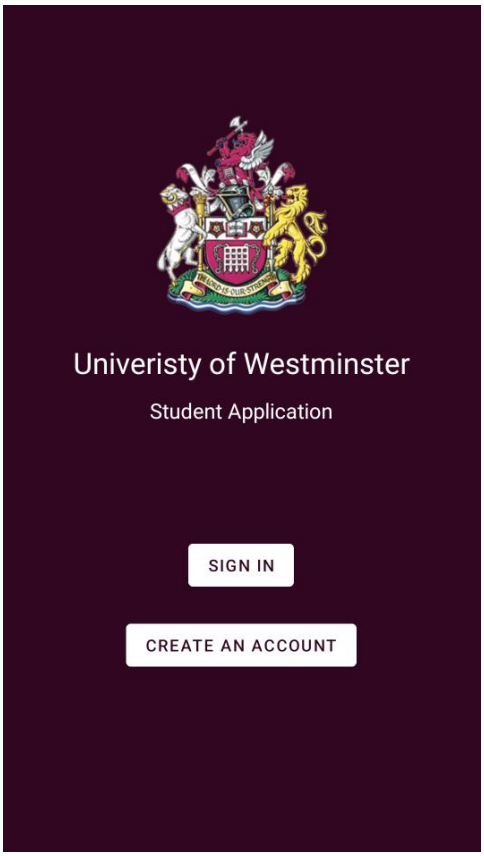


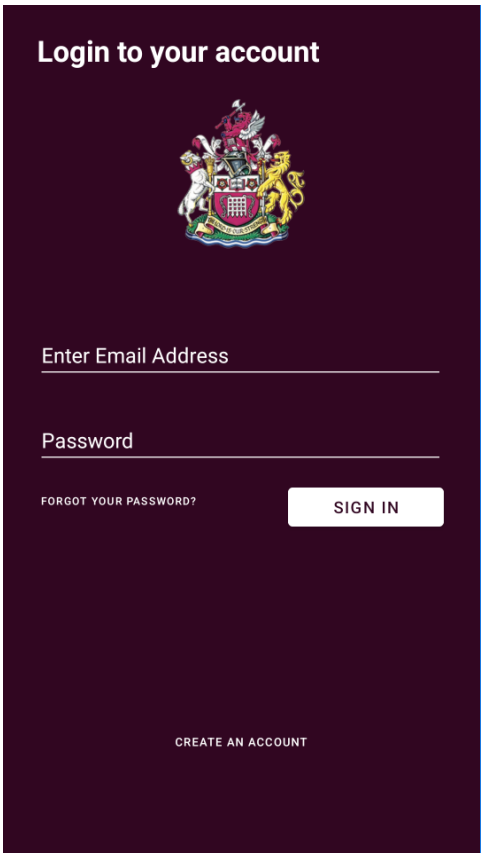
Class Diagram Part 2

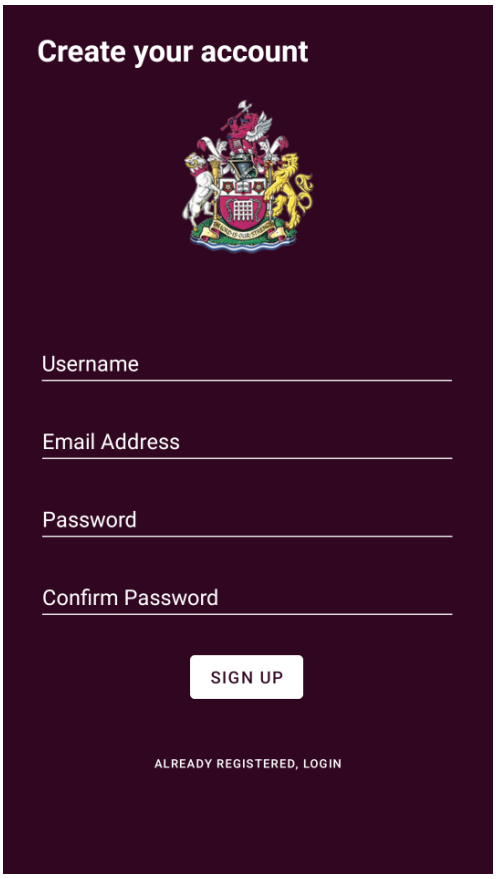


The complete Class Diagram.

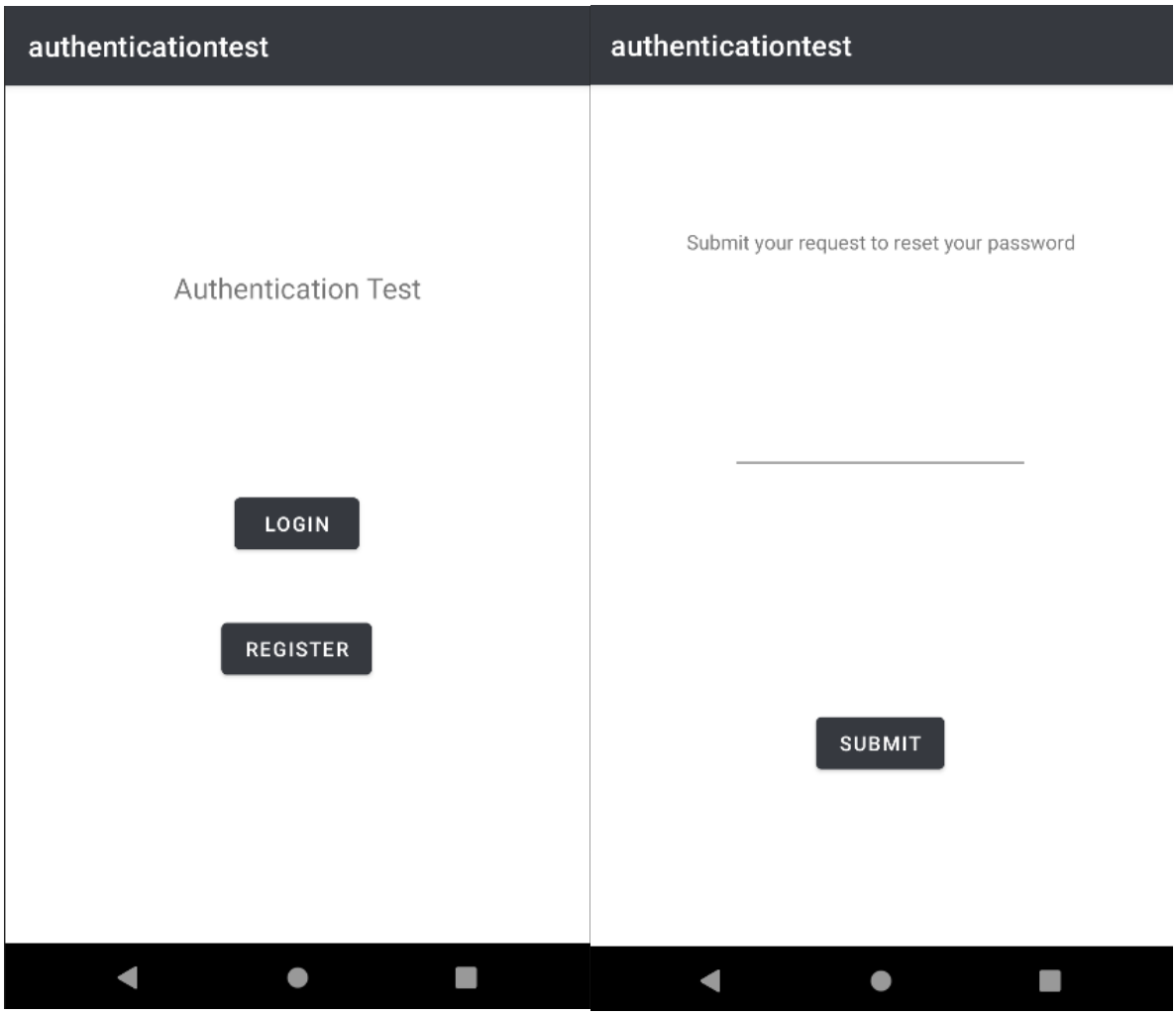
Appendix XII



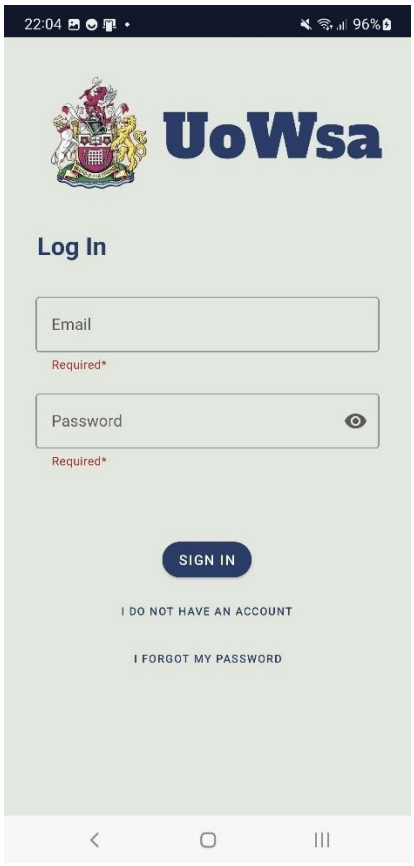


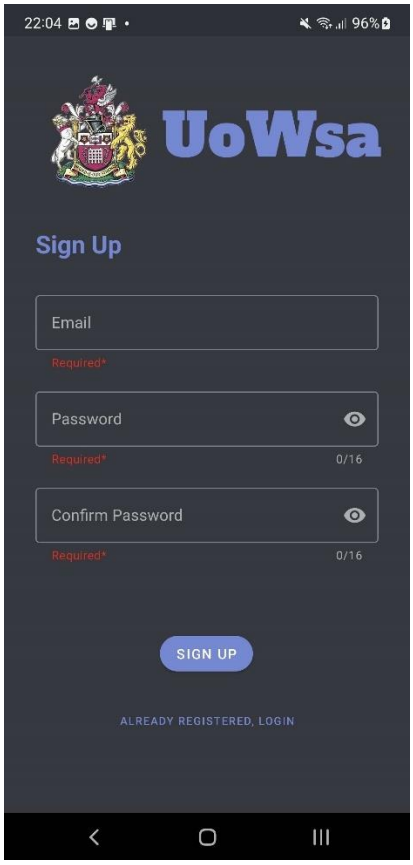


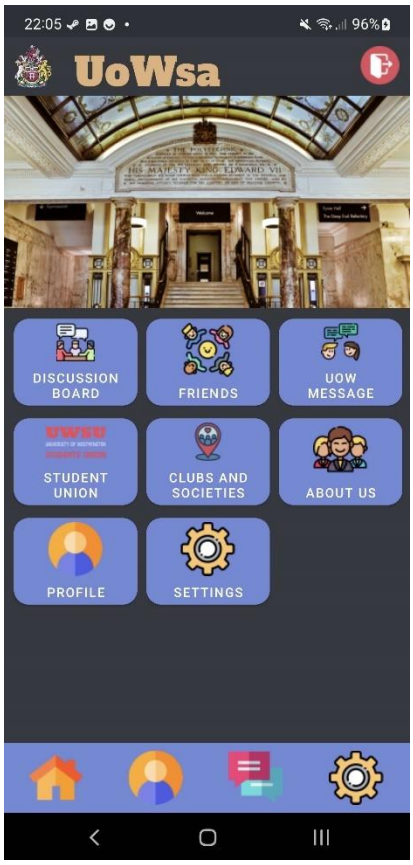
Appendix XIII



Appendix XIV








22:05 96%
Student Union
We're the University of Westminster Students' Union (UWSU), or SU if you'd like to keep it short.
We're a membership-led organisation, that exists under the Education Act 1994. Our sole purpose is to improve your life whilst at the University of Westminster; from providing free advice and support, to bangin' nights out and inclusive student groups, there's a little something for every Westminster student.
We represent you
Our core priority is to ensure you are heard and represented, and get the very best experience during your time at Westminster. Aside from carrying your voice across the University to make positive changes, we also want to make sure you get the most out of your Westminster experience by participating in plenty of diverse activities, build your networks and communities, become change-makers, and so much more!
We support you
We have so much on offer to make you feel valued and supported, and being part of a community is the first step towards building that support. Our sports clubs and society groups are the perfect space for you to meet like-minded people,


22:05 96%
supported, and being part of a community is the first step towards building that support. Our sports clubs and society groups are the perfect space for you to meet like-minded people, share experiences with, and maybe even become 'forever friends'.
We also have a team of advisors that offer free, independent, and impartial academic advice to guide and support you through any academic-related issues such as plagiarism, mitigating circumstances and lots more.
We're led by you
Our strategic direction and goals are led by you! – five student Officers are elected each year to lead the strategic direction of the Union, each striving to make fundamental changes to your student experience across the University and the SU.
We're a charity
Just like any charity, we exist to represent a specific group of people, and for us that is the students at Westminster. We are governed by a Trustee Board – these individuals hold the Union accountable by ensuring that our work is in the best interest for all students. We work within the laws and regulations governed by the Charity Commission; our board ensures that we stay within these laws and regulations.
We fight with you
We are passionate about liberation and student activism and believe that all students should have equal opportunities whilst at Westminster. it is an important stance for us and we will do everything we can to ensure that this happens and that all students are treated fairly and as equals.
Get in touch!
02079115000
su-info@westminster.ac.uk
Where to find us
35 Marylebone Rd, Marylebone, London NW1 5LS (Room MG02)
Office opening hours
10am - 5pm | Monday - Friday


22:06 95%
Student Community
THANKS for being inclusive.
Reasons to get involved
Looking to get more out of your university experience? There are hundreds of student groups for you to be part of at UWSU, all for free!
Sports Clubs
Whether you're looking to compete at a high level or try something completely new, we have something for you.
Active Lifestyle
Take part in our free and open Dragons Active session this term.
Societies
Browse our range of societies and find your community here! If you don't see anything for you, check out the societies waiting to be revived or start something new!
Committee Resources
Explore our guides and resources for committees to help with the running of your student group.


22:06 95%
UWSU Sports
Athletics
We're a small but friendly athletics team, hoping to welcome many new faces in the next year. From pushing our physical abilities to having a chat at the track, we make sure being on the team is a fun and rewarding experience for all our members. There are no requirements to join, so even if you've never run before in your life, you're welcome here! All you have to do before you come along to a session is register yourself to the club via the University of Westminster Student Union site, which is completely free. If you require any further information, feel free to send us an email.
Badminton
Welcome to Westminster's Badminton Club! Badminton is a great sport that works every part of the body and keeps you fit. This sport improves your reaction time, agility, and overall movement. Not only that, playing badminton is great for your mental health which is especially important as a student.
We have both a men's and a women's team, competing in singles, doubles, and mixed doubles in the BUCS league. These matches will be against other universities, with some of the matches being played at home or away depending on the matchup and the circumstances. Whether you've played badminton before, or just want to give it a go, get in touch or come to one of our sessions to get a feel for the sport. We look forward to meeting you!

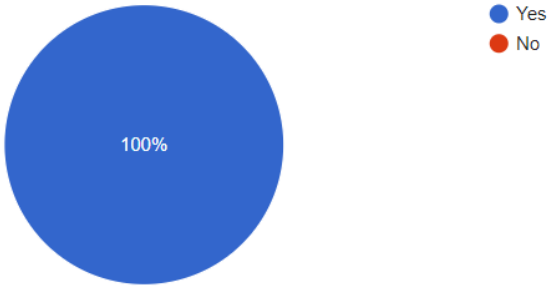
Appendix XV

Concent Form

The participation information was explained to me.

Copy

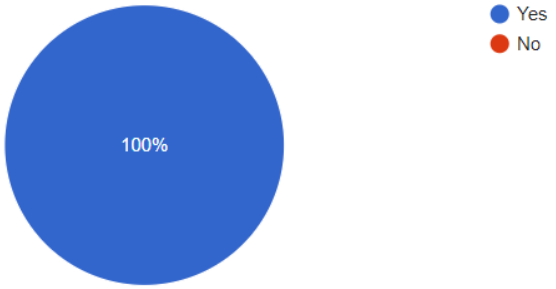
27 responses



I understand on submission the data provided cannot be removed and will be used by the lead researcher.

Copy

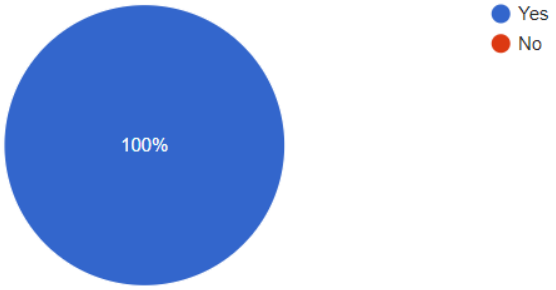
27 responses



I understand if I wish to withdraw my participation, the information I have provided will be permanently removed.

Copy

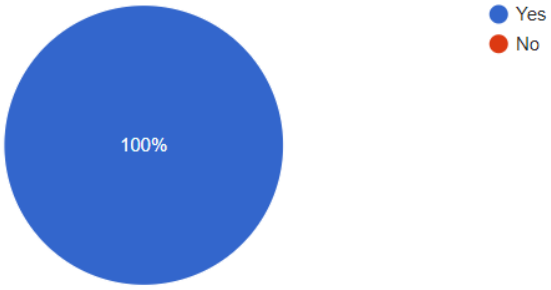
27 responses



I have the opportunity to ask the lead researcher any question regarding the research study and during testing of the mobile application project.

Copy

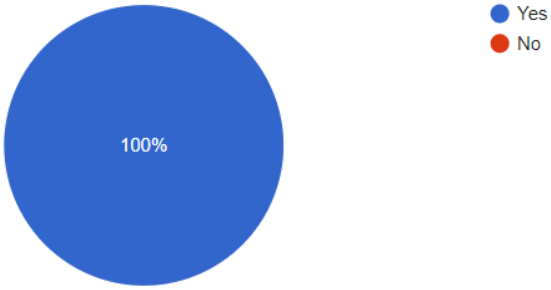
27 responses



I understand the information provided for this project will be treated with confidentiality.

Copy

27 responses



Appendix XVI

Timestamp	Consent Q1	Consent Q2	Consent Q3	Consent Q4	Consent Q5	Q1	Q2	Q3
4/26/2023 21:05:51	Yes	Yes	Yes	Yes	Yes	I believe that the colors used in the mobile application could be improved.	Very Simple	Very Simple
4/26/2023 21:37:25	Yes	Yes	Yes	Yes	Yes	I believe that the colors used in the mobile application could be improved.	Very Simple	Neutral
4/26/2023 22:09:16	Yes	Yes	Yes	Yes	Yes	I find the color scheme of the mobile application to be visually appealing and complementary.	Very Simple	Somewhat Simple
4/26/2023 22:11:53	Yes	Yes	Yes	Yes	Yes	I find the color scheme of the mobile application to be visually appealing and complementary.	Very Simple	Very Simple
4/26/2023 22:15:54	Yes	Yes	Yes	Yes	Yes	I find the color scheme of the mobile application to be visually appealing and complementary.	Very Simple	Very Simple
4/26/2023 22:45:06	Yes	Yes	Yes	Yes	Yes	I find the color scheme of the mobile application to be visually appealing and complementary.	Very Simple	Somewhat Simple
4/27/2023 2:15:06	Yes	Yes	Yes	Yes	Yes	I find the color scheme of the mobile application to be visually appealing and complementary.	Very Simple	Very Simple
4/27/2023 2:18:35	Yes	Yes	Yes	Yes	Yes	I have no strong opinion on the color scheme of the mobile application.	Very Simple	Somewhat Simple
4/27/2023 2:21:50	Yes	Yes	Yes	Yes	Yes	I find the color scheme of the mobile application to be visually appealing and complementary.	Very Simple	Very Simple
4/27/2023 2:23:28	Yes	Yes	Yes	Yes	Yes	I find the color scheme of the mobile application to be visually appealing and complementary.	Very Simple	Very Simple
4/27/2023 2:24:16	Yes	Yes	Yes	Yes	Yes	I believe that the colors used in the mobile application could be improved.	Very Simple	Somewhat Simple
4/27/2023 2:26:05	Yes	Yes	Yes	Yes	Yes	I believe that the colors used in the mobile application could be improved.	Very Simple	Somewhat Simple
4/27/2023 2:34:02	Yes	Yes	Yes	Yes	Yes	I find the color scheme of the mobile application to be visually appealing and complementary.	Neutral	Very Simple
4/27/2023 2:44:40	Yes	Yes	Yes	Yes	Yes	I find the color scheme of the mobile application to be visually appealing and complementary.	Very Simple	Somewhat Simple
4/27/2023 2:50:27	Yes	Yes	Yes	Yes	Yes	I have no strong opinion on the color scheme of the mobile application.	Very Simple	Very Simple
4/27/2023 3:00:52	Yes	Yes	Yes	Yes	Yes	I find the color scheme of the mobile application to be visually appealing and complementary.	Very Simple	Very Simple
4/27/2023 3:03:26	Yes	Yes	Yes	Yes	Yes	I find the color scheme of the mobile application to be visually appealing and complementary.	Very Simple	Very Simple
4/27/2023 3:18:38	Yes	Yes	Yes	Yes	Yes	I find the color scheme of the mobile application to be visually appealing and complementary.	Somewhat Simple	Very Simple
4/27/2023 3:20:34	Yes	Yes	Yes	Yes	Yes	I find the color scheme of the mobile application to be visually appealing and complementary.	Very Simple	Very Simple
4/27/2023 3:32:13	Yes	Yes	Yes	Yes	Yes	I have no strong opinion on the color scheme of the mobile application.	Very Simple	Very Simple
4/27/2023 3:51:01	Yes	Yes	Yes	Yes	Yes	I find the color scheme of the mobile application to be visually appealing and complementary.	Very Simple	Very Simple
4/27/2023 3:51:02	Yes	Yes	Yes	Yes	Yes	I have no strong opinion on the color scheme of the mobile application.	Somewhat Simple	Very Simple
4/27/2023 3:51:09	Yes	Yes	Yes	Yes	Yes	I have no strong opinion on the color scheme of the mobile application.	Very Simple	Neutral
4/27/2023 3:53:28	Yes	Yes	Yes	Yes	Yes	I find the color scheme of the mobile application to be visually appealing and complementary.	Very Simple	Very Simple
4/27/2023 3:54:03	Yes	Yes	Yes	Yes	Yes	I have no strong opinion on the color scheme of the mobile application.	Somewhat Simple	Very Simple
4/27/2023 3:54:20	Yes	Yes	Yes	Yes	Yes	I find the color scheme of the mobile application to be visually appealing and complementary.	Very Simple	Very Simple
4/27/2023 3:55:29	Yes	Yes	Yes	Yes	Yes	I find the color scheme of the mobile application to be visually appealing and complementary.	Very Simple	Very Simple

Q4	Q5	Q6	Q7	Q8	Q9	Q10
Very Useful	Somewhat Useful	Very Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile needs improvement.	Very Effective
Very Useful	Somewhat Useful	Very Useful	Yes	The process of blocking and unblocking users on the application needs improvement.	The user profile needs improvement.	Very Effective
Very Useful	Somewhat Useful	Very Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile needs improvement.	Very Effective
Very Useful	Very Useful	Very Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile is excellent.	Somewhat Effective
Very Useful	Very Useful	Very Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile is excellent.	Very Effective
Very Useful	Very Useful	Very Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile is excellent.	Somewhat Ineffective
Very Useful	Very Useful	Very Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile is excellent.	Very Effective
Somewhat Useful	Somewhat Useful	Very Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	I have no opinion on the implementation of the user profile.	Somewhat Effective
Somewhat Useful	Somewhat Useful	Somewhat Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile needs improvement.	Very Effective
Very Useful	Very Useful	Somewhat Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile needs improvement.	Very Effective
Neutral	Very Useful	Somewhat Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile is excellent.	Very Effective
Very Useful	Neutral	Very Useful	Yes	The process of blocking and unblocking users on the application needs improvement.	The user profile needs improvement.	Very Effective
Very Useful	Very Useful	Very Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile needs improvement.	Very Effective
Somewhat Useful	Somewhat Useful	Neutral	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile is excellent.	Very Effective
Very Useful	Very Useful	Very Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile needs improvement.	Very Effective
Very Useful	Very Useful	Very Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	I have no opinion on the implementation of the user profile.	Very Effective
Very Useful	Very Useful	Very Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile is excellent.	Very Effective
Somewhat Useful	Very Useful	Very Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile is excellent.	Very Effective
Very Useful	Very Useful	Very Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile is excellent.	Very Effective
Very Useful	Very Useful	Very Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile is excellent.	Very Effective
Very Useful	Very Useful	Very Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile is excellent.	Very Effective
Very Useful	Very Useful	Very Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile is excellent.	Somewhat Effective
Somewhat Useful	Very Useful	Neutral	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile needs improvement.	Very Effective
Very Useful	Very Useful	Very Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile is excellent.	Very Effective
Somewhat Useful	Very Useful	Neutral	Yes	No, the process of blocking and unblocking users on the application was confusing.	The user profile is excellent.	Very Effective
Very Useful	Very Useful	Very Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile is excellent.	Very Effective
Very Useful	Very Useful	Very Useful	Yes	Yes, the process of blocking and unblocking users on the application was straightforward.	The user profile is excellent.	Very Effective

Q11	Q12	Q13
No, creating a forum on the application was straightforward and easy.	I believe that the settings section of the application has enough features.	Yes
Yes, I faced some difficulties when creating a forum on the application.	I think that the settings section of the application has too few features.	Yes
No, creating a forum on the application was straightforward and easy.	I believe that the settings section of the application has enough features.	Yes
No, creating a forum on the application was straightforward and easy.	I believe that the settings section of the application has enough features.	Yes
No, creating a forum on the application was straightforward and easy.	I believe that the settings section of the application has enough features.	Yes
No, creating a forum on the application was straightforward and easy.	I feel that the settings section of the application has too many features.	Yes
No, creating a forum on the application was straightforward and easy.	I believe that the settings section of the application has enough features.	Yes
No, creating a forum on the application was straightforward and easy.	I believe that the settings section of the application has enough features.	Yes
No, creating a forum on the application was straightforward and easy.	I believe that the settings section of the application has enough features.	Yes
Yes, I faced some difficulties when creating a forum on the application.	I believe that the settings section of the application has enough features.	Yes
No, creating a forum on the application was straightforward and easy.	I believe that the settings section of the application has enough features.	Yes
I did not create a forum on the application, so I cannot comment on any challenges.	I think that the settings section of the application has too few features.	Yes
No, creating a forum on the application was straightforward and easy.	I believe that the settings section of the application has enough features.	Yes
No, creating a forum on the application was straightforward and easy.	I believe that the settings section of the application has enough features.	Yes
I did not create a forum on the application, so I cannot comment on any challenges.	I believe that the settings section of the application has enough features.	Yes
No, creating a forum on the application was straightforward and easy.	I believe that the settings section of the application has enough features.	Yes
No, creating a forum on the application was straightforward and easy.	I believe that the settings section of the application has enough features.	Yes
No, creating a forum on the application was straightforward and easy.	I believe that the settings section of the application has enough features.	Yes
No, creating a forum on the application was straightforward and easy.	I believe that the settings section of the application has enough features.	Yes
No, creating a forum on the application was straightforward and easy.	I believe that the settings section of the application has enough features.	Yes
No, creating a forum on the application was straightforward and easy.	I believe that the settings section of the application has enough features.	Yes
No, creating a forum on the application was straightforward and easy.	I believe that the settings section of the application has enough features.	Yes
Yes, I faced some difficulties when creating a forum on the application.	I believe that the settings section of the application has enough features.	Yes
No, creating a forum on the application was straightforward and easy.	I believe that the settings section of the application has enough features.	Yes
Yes, I faced some difficulties when creating a forum on the application.	I believe that the settings section of the application has enough features.	Yes
No, creating a forum on the application was straightforward and easy.	I believe that the settings section of the application has enough features.	Yes
No, creating a forum on the application was straightforward and easy.	I believe that the settings section of the application has enough features.	Yes

Q14
The application was very easy to use and understand. I especially liked the forum section. I think the application can benefit from having group direct messages.
I think the application can be improved to make the design more visually appealing and stylish, with high-quality graphics and animations
Overall the design is positive for user experience. Perhaps the notifications can be expanded a little more throughout the application.
The application looks great and very responsive.
The app overall looks great for the first version, there are improvements to made for future improvements, I assume.
I believe the direct messaging system as it is currently would need to have a way to display when a message is deleted as it feels like a bad actor would be able to abuse messages just being gone, and also adding a sort of preview of what the contents of a direct message would be as seeing just a number however big it
Feels very professional. Would love to have an app like it!
Would love a way to pin coversations
Would like to see more customised options on peoples profile (age, pursued degree clubs)
profile need to improve , and adding more feature
Would like to see more colours
I think the block and unblock need to have more detail
when the user send a message, at least have notification because it important to know someone sending to us
The format is really easy to understand.
Would like more customisation
Didn't encounter any problems using the app
Easy way to communicate and find information on events
Found it really easy to make forums
Would like a way to filter forums
Made it easy to get in contact with students without the need of their phone numbers
Love how easy the forum section is
Overall the work was well done, the profile page could add a few more details
the profile to improving , need to have more feature
Easy to get important information
The application need to have notification , when someone send to me
Easy to use
Love the dark mode