Manuja Suran Wickramasinghe Mudiyanselage

1812914

Unified Explore - an All-in-One Communication System

B.Sc. (Hons) Computer Science and Software Engineering

Undergraduate Thesis Report

Department of Computer Science & Technology

University of Bedfordshire

Mr Iresh Bandara

2018/2019

Acknowledgments

First, I wish to show my appreciation to my assigned supervisor Mr Iresh Bandara for his professional guidance throughout the whole duration of the project. I was advised on how to complete documentation, as well as in the development of the proposed system to successfully complete my research. Next, my gratitude is extended to my lecturer in charge Ms Gayana Fernando for her suggestions and help in the different stages of this research. Furthermore, I want to give my colleagues at university special thanks for their support when working on assignments and relevant components regarding the project.

Abstract

The modernized world we currently live in is a complex one that is assisted with an enormous variety of mobile applications. These mobile applications help in individual areas like navigation, communication, socialization, or even entertainment. This sounds beneficial at first but proves to be inefficient at last. The issue rises in storing multiple applications on a personal device, relying on multiple updates that users can't even keep track of and in the long run draining the performance on your smartphone up to a point that the phone is not usable regards to any kind of background activities. The proposed system, entitled with the name Unified Explore is an all-in-one service application that combines important areas to assist the users daily life with one mobile based application. This new concept will approach to change how we use current navigation systems or existing social media platforms. Unified Explore will even gamify those prior mentioned individual services and provide an attempt in improving communicational behaviour and skills as we know of in modern society. This approach will hopefully result in building a new active and adventurous online community that makes use of Unified Explore. As its name implies, the system aims not only to unify people by exploring and bringing them together wherever situated they might be around the globe in the sense of communication between people but also enhance the communication between the device and the user by combining multiple features and services in regards to user assistance.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

DB – Database

IDE – Integrated development environment

SDK – Software development kit

UI – User interface

Wi-Fi – Wireless fidelity

WWW – World Wide Web

1. Introduction

1.1. Project Background

In this day and age, almost everyone owns a smartphone, and these users are said to spend 69% of their time on social media. In the rapid development within social media platforms, multiple improvements occurred that now gives users the optionality to select a great variety of applications (De Silva, 2018). Despite there being so many options, on a day-to-day basis only 9 applications are used, and 30 on a monthly basis. Using mobile applications to assist in one's daily task is no longer something we consider for the future to have in store, but it is already the reality of the present day (Perez 2017).

Recent mobile apps now offer many features and services that focus fields such as entertainment, assistance, e-business and more. These applications individually help users in a great way. For examples, navigation systems now not only show the shortest route but also the traffic hotspots along the way, alternate routes that the user might be interested in, add checkpoints along the way so it can determine the most convenient path to be taken and more. While individual applications hold the power to benefit its users to a great amount, an all-inclusive application, which combines a range of services would serve a huge purpose and can be considered an attractive solution to society and its demands.

A few of the most popular and frequently used platforms are Facebook and Google, and these platforms are very specific to their limited services, such as Facebook being a social media platform that can be used to simply interact with one another in the form of statuses, media, and direct messaging. Facebook is entitled to having the highest number of downloads in the year 2018. There were 4.119 billion downloads on Google Play store alone (Dan Price, 2018). Despite it being the most popular download, it also has nearly 10 million 1-star ratings, because the app includes several features that barely anyone uses, further suggesting that more useful features could have been added instead (MakeUseOf, 2019). Right after Facebook, Facebook Messenger comes in second with the highest number of downloads at a

rate of 3.408 billion. This is followed by Whatsapp at 2.979 billion and then Instagram following with a still an immense number of 1.843 billion downloads.

A trend that can be noticed in the top list is that all these apps serve the same purpose. They may have a few features that make them different from one another, but they all serve a key purpose, which is communication - in the form on media, direct messaging, and status updates.

It is safe to say that majority of the people who have a smartphone have all of the apps named above, even though they all do the same thing and serve the same purpose, all four apps will most likely be available. This makes it a little redundant, kills storage and also could degrade performance. If a user were to access the map in order to navigate to a location, one of the social media applications would need to be closed and the user would have to switch between these apps to utilize what is required at the point of time. This can be considered to be time inefficient and inconvenient regarding the overall performance. This issue can be solved by integrating multiple services, all into one application. This would aid in improving the sharing of information in real-time, such as sharing traffic information for crowdsourced navigation applications, as well as media sharing and more. The integration of this service would not strain the device's performance and storage as much, and it would also save the user's time.

1.2 Project Aims & Objectives

1.2.1 Aims

The aim of the project is to develop an android application to advance communication for people through supporting and facilitating real-world activities.

1.2.2 Objectives

The objectives of this project include -

- To identify a crowdsourcing technique and incorporate it with a gaming approach in order to receive feedback of locations in the form of a review for the system

- To identify a profile matching algorithm that would allow users to share common interests and who are geographically close by to connect through the application
- To compare similar applications to find inspiration for the system designs
- To identify a programming language that would best suit the implementation of the application
- To design a system that allows real-time data sharing and retrieval
- To evaluate a unique navigation system that would support user-defined activities and movement patterns and produce route calculations and directions

1.3 The Project Artefact

The artefact is an all-in-one communication system which includes a navigation system that provides the best route based on the user's itinerary for that day. For example, the user can store the locations that they need to go to in a particular day, and the app should be able to provide the most logical and convenient route for the user. The navigation section of the system will also be gamified. Users can upload pictures of the locations that they visit, and the most popular pictures are displayed on a scoreboard. The rankings will also be displayed on a different tab so users can keep up with the top trending posts and locations. Additionally, the application should be able to allow users to find friends, not only by common interests but also based on location. This would allow users to not only make friends online but to also meet them in person and bond over the interests that they share. Also, the user can keep in touch through simple text messaging once having found each other. In case a user wants to get information about the potential friend he or she is trying to contact or meet in person a user profile displays all relevant information about that person. The profile is customizable to change personal details anytime if needed. Lastly, the system gives a very user-friendly experience and provides a great visual look that can attract any users so that no one will get bored with the system in the long-run.

1.4 Report Structure

The thesis report consists of five main chapters -

- Introduction - This section will consist of a background of the project and discuss the current status of the project with regards to applications that are already on the market. The project aims and objectives will be introduced followed by an introduction to the proposed system.
- Literature Review - This section consists of the research carried out on relevant techniques and technologies along with existing systems similar to the proposed system. The research will be based on mostly research articles and journals published in recent years.
- Methodology - The SDLC model used will be discussed here. Each stage will be discussed thoroughly along with action steps and relevant diagrams.
- Results & Discussion - The results section will show the working functionalities along with any important code segments that have been implemented. The Discussion section will review any technical problems that have been experienced along with any changes that may have had to be made to accommodate the project. Additionally, the system's reliability and accuracy will also be discussed.
- Conclusion - This final section will discuss whether the set objectives have been achieved and the aim has been met. It will further consider the limitations of the researcher's study. Future work and recommendations will also be introduced.

2. Literature Review

2.1 Navigation Systems using Crowdsourcing Approaches

2.1.1 Crowdsourced Road Navigation

Road navigation systems that have been automated throughout time have increased in becoming a popular source for the development of mobile apps over the course of the past 20 years (Fan et al., 2017). Early models that utilized and focused on the service of navigation relied almost entirely on the support of major companies at the point of time. Examples for such products or companies are Magellan, TomTom, and Garmin. With the introduction of Third and Fourth Generation (3G/4G) Networking, came the opportunity for drivers everywhere to be well connected through means of accessing and sharing sensitive and important data information across the internet with a community. Whereas by today's standards, one of the greatest navigation services being Google Maps has struck billions of people on its daily usage meter. Features such as real-time traffic feed and real-time feed on locations that are under construction are available as well. The quality of the routes that the application suggests to its users is generally acceptable. However, it is very common that the suggested routes often times fail to be agreed upon by drivers because the routes don't represent the reality of the local driving conditions that the driver's experience (Ceikute and Jensen, 2013). This proves that there is potential in exploring crowdsourced navigation led by crowd intelligence.

Navigation first started off with standalone devices. The GPS Project was launched and worked to provide geo-locations and time information to a satellite receiver. Although high-sensitivity GPS chipsets have been taken up in recent years, the standalone version of GPS does not work as precisely in indoor and urban environments. This resulted in the use of complementary position systems in smartphones. Most smartphones are now GPS-enabled and additionally employ a technology called A-GPS (Assisted GPS) where an assistant server provides clock and satellite orbit data and information to increase accuracy and correct GPS signal deterioration. Wi-Fi is also being used to identify other existing Wi-Fi signals which aid to determine the specific location of a device.

The new generation of navigation has been making use of crowd intelligence. Companies such as Google and TomTom have taken advantage of crowdsourcing for updating and maintaining maps. Users submit any noticeable changes in the roads with regards to traffic, constructions or shortcuts. The resulting changes are then revised manually and finally integrated into the systems map. These systems heavily rely on their users to constantly update, maintain and improve the provided service. Existing platforms including Waze and OpenStreet, maintain maps that are crowdsourced and have attracted millions of users and contributors. Recent activities in this field are summarized in the table below (Table 1).

CHAPTER 2 TABLE 1: NAVIGATION APPLICATIONS WITH CROWDSOURCING

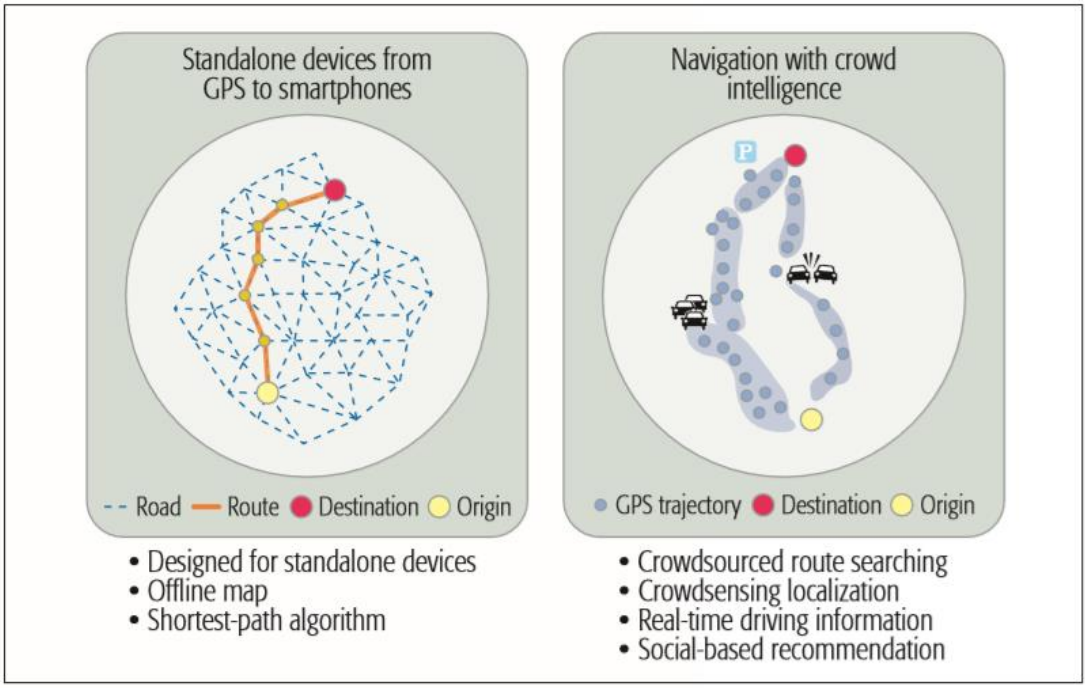
Authors	Catogries	Collected data	Technique summary
T-drive [6]	Route planning	GPS trajectories	Find fastest routes on historical trajectories
Wei *et al.* [7]	Route planning	GPS trajectories	Popular routes from uncertain trajectories
Zheng *et al.* [8]	Route planning	GPS trajectories	Propose a HITS-based inference model
CrowdNavi [2]	Route planning	GPS trajectories	Navigate in the last mile with crowdsourced driving information
CrowdPlanner [9]	Route planning	Crowdsourced answers	Determine the best route based on human worker answers
Zee [10]	Localization	Motion sensors	Fingerprint crowdsourcing-based indoor localizatoin
CrowdMap [11]	Digital map construction	Image and motion sensors	Producing floor plans based on image and motion sensors
CrowdInside [12]	Digital map construction	Motion sensors	Producing floor plans based on motion traces
CrowdAtlas [13]	Digital map construction	GPS trajectories	Infer and complement missing lanes
Yoon *et al.* [14]	Recommendation	GPS trajectories	Rank itinerary from user-generated GPS trajectories
Google Local Guide	Recommendation	Crowdsourced answers	Encourage users to help others with their local experience
Wang *et al.* [15]	Location authentication	WiFi APs	Use physical devices to complete proximity authentication

The above table discusses the following categories -

- Digital Map Construction - Recent work has led to the discovery of analysing user's movements to automatically update maps. CrowdMap uses in order to keep track of a user's movement a crowdsourced concept to retrieve images and sensor (Chen et al., 2015). It then uses the traces and image context to construct a detailed floor plan. CrowdInside uses sensors on smartphones and automatically generate precise motion traces (Alzantot and Youssef, 2012).
- Route Planning - GPS now allows users to mark down location histories based on their motion traces. This enables the monitoring of human behaviour and the user's route preferences of travel. The GPS then compares the routes taken by multiple users and looks for more convenient routes and shortcuts that could be suggested to the user next time around. CrowdPlanner is a route recommendation system that inquires users to assess the

recommended routes and establish the best route based on the feedback that was received. Human assessments are used over machine algorithms because of the real-world knowledge and experience that humans possess (Su et al., 2014).

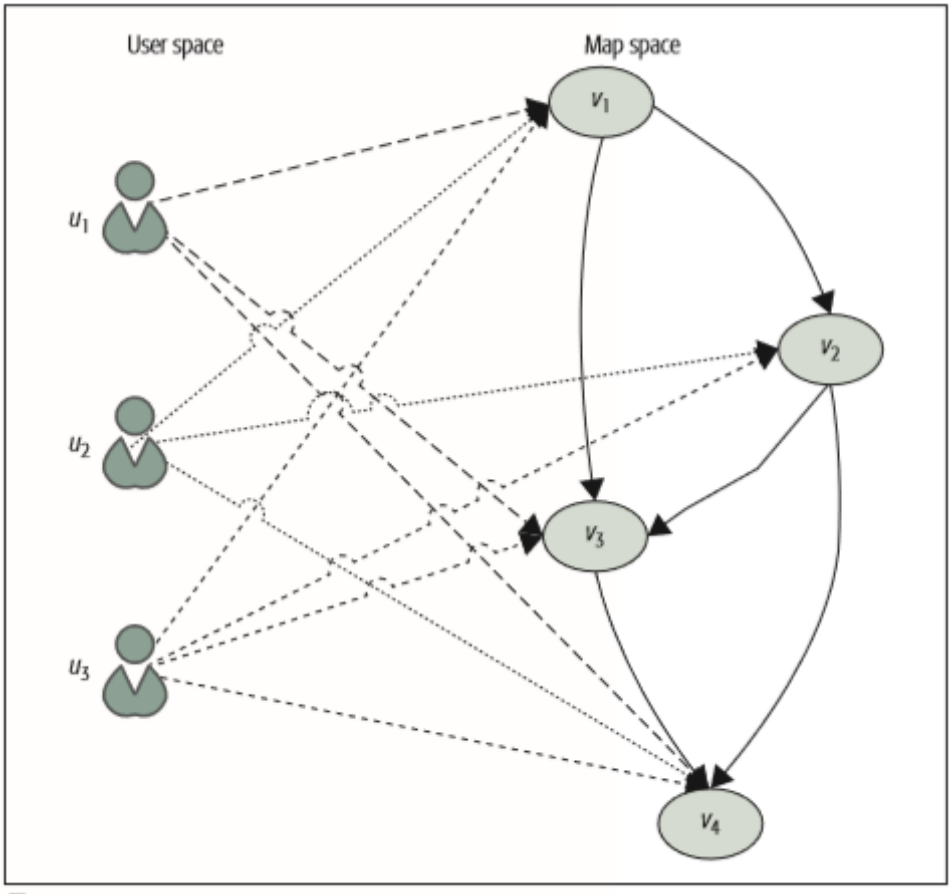
- Social Based Recommendations - Social networks have recently incorporated route recommendation systems. Social media now consists of elements such as geotagged photographs upon uploading, checking in at different locations. This kind of data is processed and used to identify patterns in regard to places that the user visits in an area frequently.



CHAPTER 2 FIGURE 1: NAVIGATION WITH STANDALONE DEVICES VS. CROWD INTELLIGENCE

CrowdNavi is a recently developed system that can be installed on drivers' physical devices such as mobile phones or internet-enabled car consoles. Once the application CrowdNavi has been launched and set enabled, it will run in the background of the device. It monitors the user's movements and reports the location information periodically to a backend server. When a user needs to find a destination, the application first identifies the last recorded location segment and then uses an external navigation system, such as Google Maps, to travel from that location segment to the desired destination. The route segment will then be

calculated from the server end using the driving patterns drawn from the crowd. This system consists of a Landmark Scoring Model; there are two node, landmarks, and users. The user can pass through several landmarks, and a landmark can be passed through several users. This is illustrated in the below figure (Figure 2).



CHAPTER 2 FIGURE 2: LANDMARK SCORING MODEL

In the figure, user U1 visits V1 → V3 → V4, where U1 points to V1, V3, and V4. This implies a good user can reveal several good landmarks, and a good landmark is revealed by several good users.

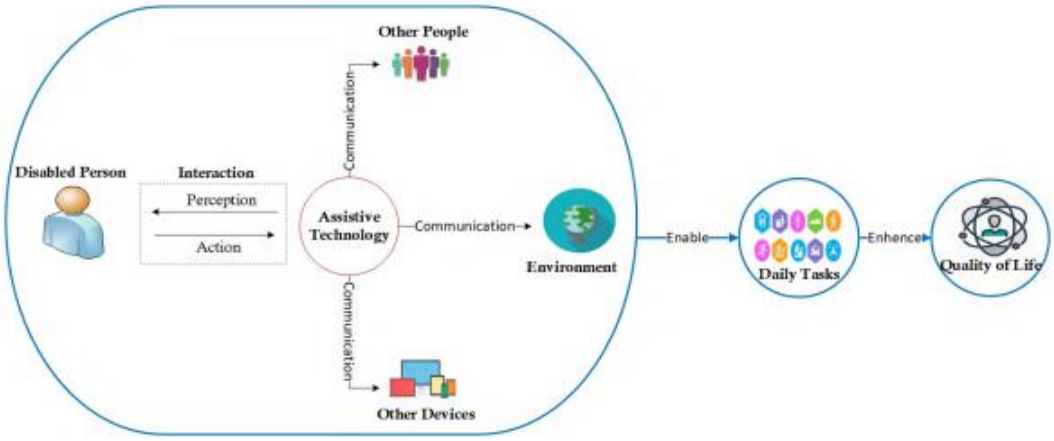
The popularity of a landmark can be determined by the frequency at which it has been visited by a good user. By analysing the increase in popularity of a specific landmark with regards to user sense, a function to determine landmark preference can be designed over time.

However, crowdsourced systems can tend to be vulnerable to malicious users who intend to rattle the system in order to disrupt the algorithms. This can be influenced

by competitor markets who may hire people to falsify their location to mislead the system. There are no definitive techniques that have been set up to authenticate a device's location. Users can also falsify traffic hotspots and mislead routes at any desired location by simulating device movement and location positioning. A multisensor, cross-validation method is said to be considered useful in order to make it difficult for users to falsify location information but is yet to be implemented.

2.1.2 Navigation for the visually impaired through Crowd Assistance

Assistive Technology has been designed in the form of systems, services, devices, and applications in order to facilitate daily activities for the disabled. Figure 3 illustrates the interaction that occurs between assistive technology and the user (Hersh and Johnson, 2008).



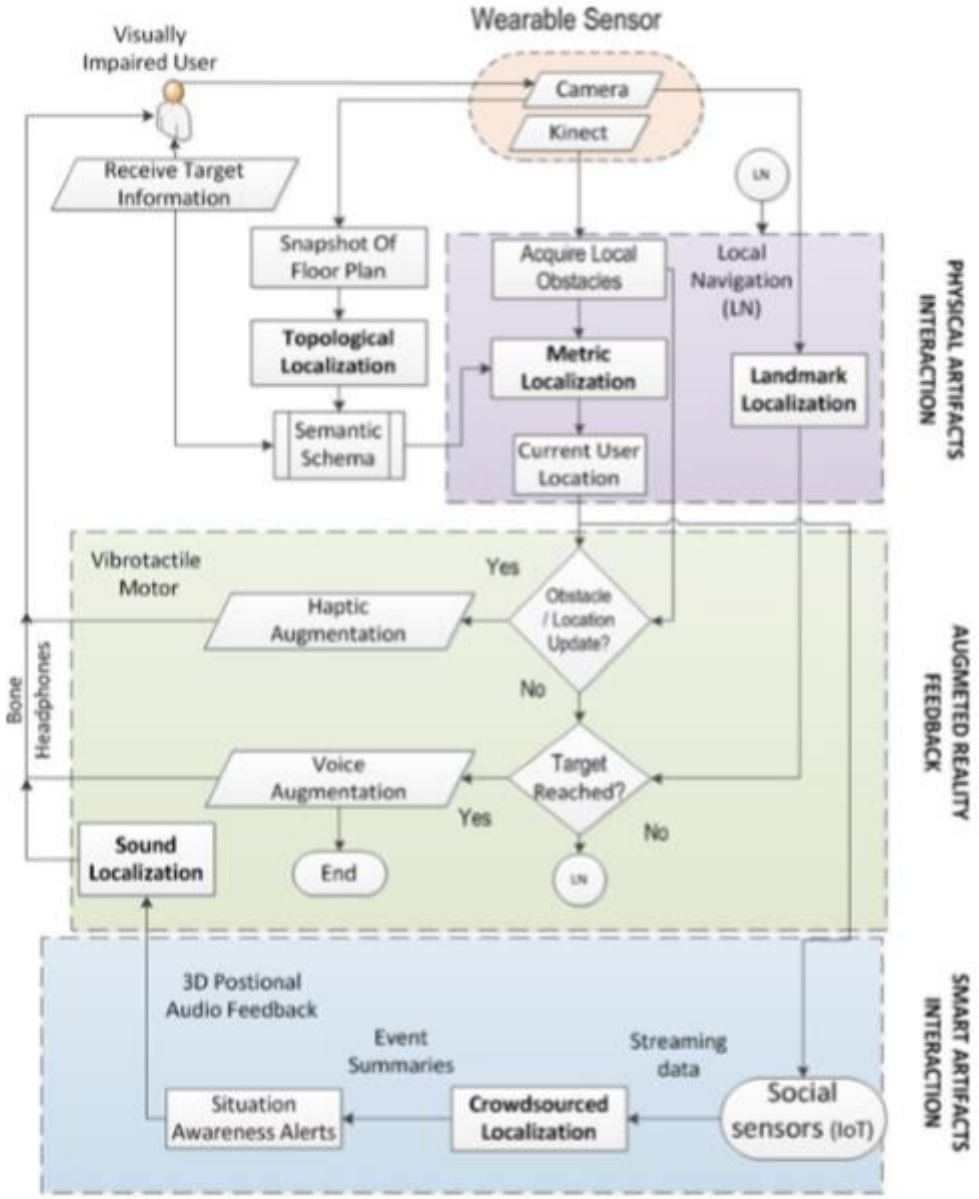
CHAPTER 2 FIGURE 3: INTERACTION BETWEEN ASSISTIVE TECHNOLOGY AND USER

Crowd-assisted navigation systems is a part of Assistive Technology, that has been designed to aid the visually impaired to navigate through streets to reach their desired destinations. Such navigation systems in an ambient environment implement the following -

- Line-of-sight interaction with the use of location-context awareness. This consists of object recognition and detection, the recognition of text and signatures, and the detection of mobile objects.

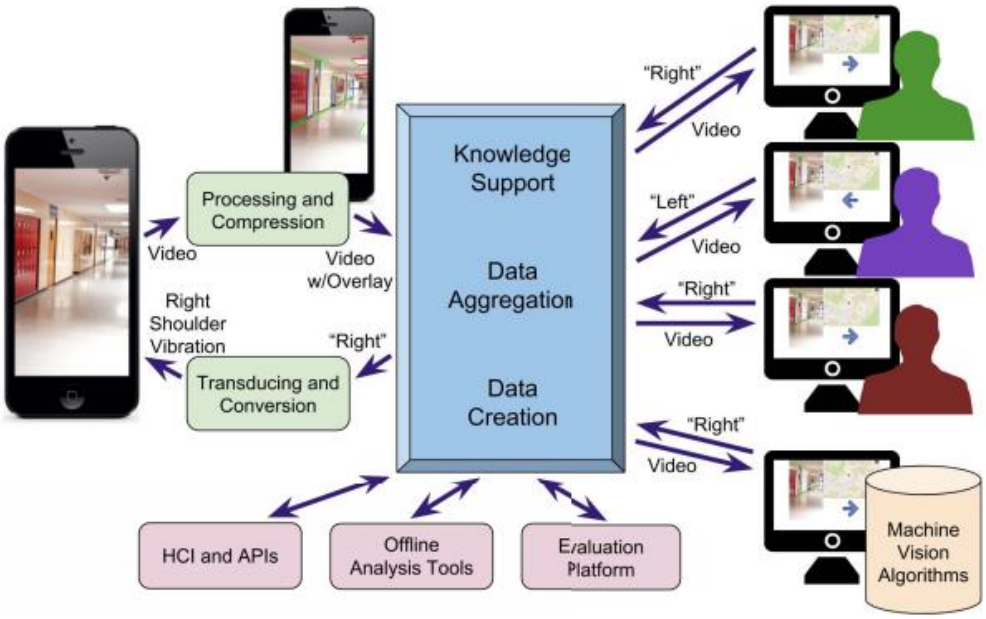
- Event-based artefacts with a crowdsourcing approach for discovering live events and social activities that may foster a crowd. Additionally, it provides warnings of any hazardous occurrences.
- An assistive SLAM (Simultaneous Localization and Mapping) based navigation system to plan custom routes and generate a map. This should reduce the user's need to remember landmarks and a general perception of their environment.
- A non-vision UI that could communicate map information, which includes audio guidance, spacial awareness updates, object locations, and a way-finding technique for the visually impaired.

The localization, the estimation of a user's location in their environment, is illustrated in Figure 4 with regards to real-time localization technologies.



CHAPTER 2 FIGURE 4: LOCALIZATION PARADIGM

A similar application works by retrieving directions from reliable volunteers online, who will assign directions to routes they are travelling in, left right, forward, backward, stop, etc. These indications will then be analysed and transmitted back to the visually impaired through an audio format (Olmschenk et al., 2015).



CHAPTER 2 FIGURE 5: CROWD ASSISTED NAVIGATION APPLICATION - AN OVERVIEW

The above Figure 5 illustrates how the data flows within the application in the form of an overview.

2.2 Social Friend Recommendation

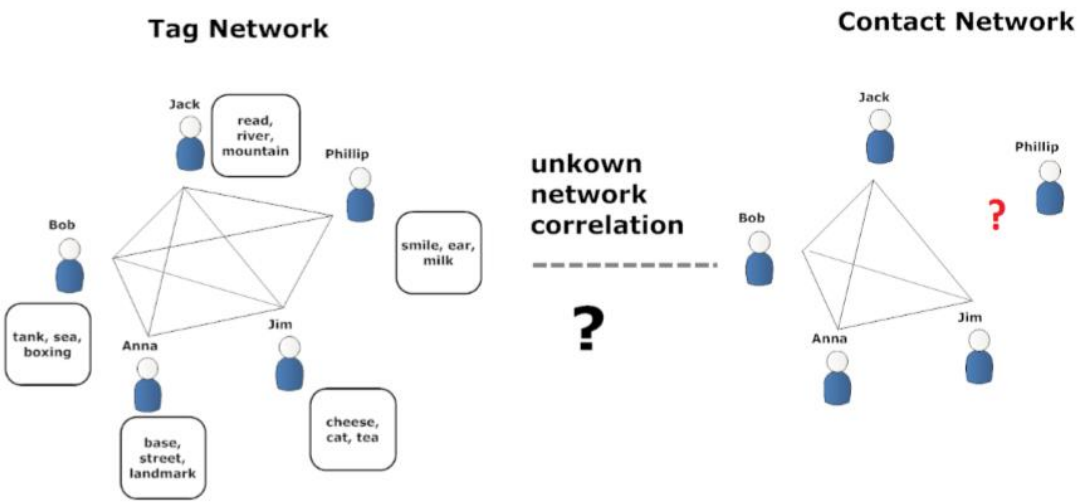
Social networks have grown explosively over the course of the last decade (Huang et al., 2016). There are billions of users online who are constantly looking to connect with someone new in order to grow their friend circles. To cater to this desire, friend recommender systems have been designed. One of the primary methods of recommending friends was based upon similarities in content, such as common the visual looks in the form of themes on a user's profile, images or statuses. However, it was argued that in order to build a reliable recommendation system that performs well and outputs accurate results several other variables and factors need to be considered thoroughly (Xie, 2010).

The following aspects can be considered when making friends in social aspects -

- Social Environment - This includes where one lives and works (Barnett and Casper, 2001)

- Social Behaviour - This may include habits such as shopping behaviours, work performance, hobbies and most importantly, how one interacts with another (Rummel, 1991; Weber, 1991)
- Social Status - This includes age, gender, designation, etc. (Brym and Lie, 2013)

To influence the correlation between different networks, a social network has been presented, where nodes on the graph represent the user and the edges represent relationships between these users. Different kinds of relationships can be illustrated using different topologies. For example, a user may upload pictured with tags for descriptions. If two users are in each other's "friend list", a contact network is built where the nodes represent users and edges represent whether these users are friends. Thus, a tag network is constructed based on this knowledge. Figure 6 demonstrates the contact and tag network for a group of users where the nodes in this structure represent the name of the users, but the edges, on the other hand, represent the correlation between the tags set which is used by each of those users.

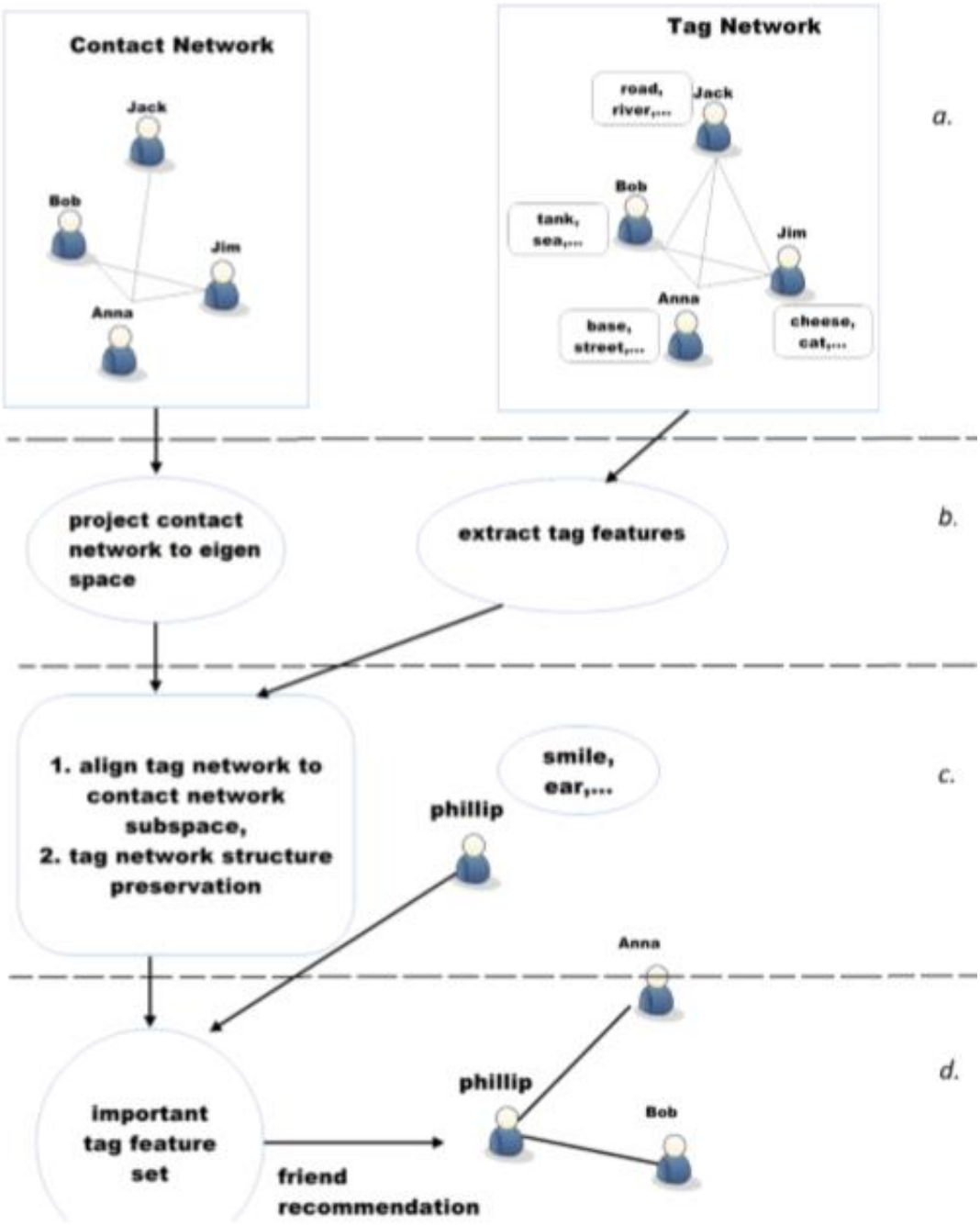


CHAPTER 2 FIGURE 6: CORRELATION OF TWO TYPES OF NETWORKS

Based on the correlations found on these networks, assumptions can be made, such as, if the tag network is strong between two nodes, it could also mean that these two users are most likely in each other's "Friend lists".

Research has also been carried out for friend recommendation based on the multimedia environment. Different approaches to social data are used for integrated recommendation. Several methods have been researched upon. One of these researches takes into account user-generated content on social media platforms, such as Twitter, and develops an algorithm instantaneously for real-time recommendations that have similarities based upon user-to-user or follower-follower (Hannon, Bennett and Smyth, 2010). Another research looks into applications that make recommendations based on location and interesting words and their co-occurrence (Li and Chen, 2009).

In a more recent study, there are discussions about the initial formation of three networks; tag network, image content network, and user friendship network. Relations are then defined between these networks. Based on these relations an algorithm is designed for multipurpose recommendations, such as friend/item recommendation. It uses a mechanism for picking out important features. This gives a new perspective for the interpretation of the properties on a social network. Correlations on network alignment are mostly based off of behaviour patterns and predictions.



CHAPTER 2 FIGURE 7: FRAMEWORK OF A SAMPLE ALGORITHM

a) Original tag and contact
b) Contact network projection and tag feature extraction
c) Align tag network to contact network
d) Feature selection and friend recommendation

A sample algorithm is listed below in Figure 8, in relation to the research conducted by Huang and his team (Huang et al., 2016)

Algorithm 1 Proposed NC based SFR

Input:

tag feature matrix $\mathbf{X}$, contact matrix $\mathbf{K}$, tag feature vector of the new user $\mathbf{x}$, number of friends K

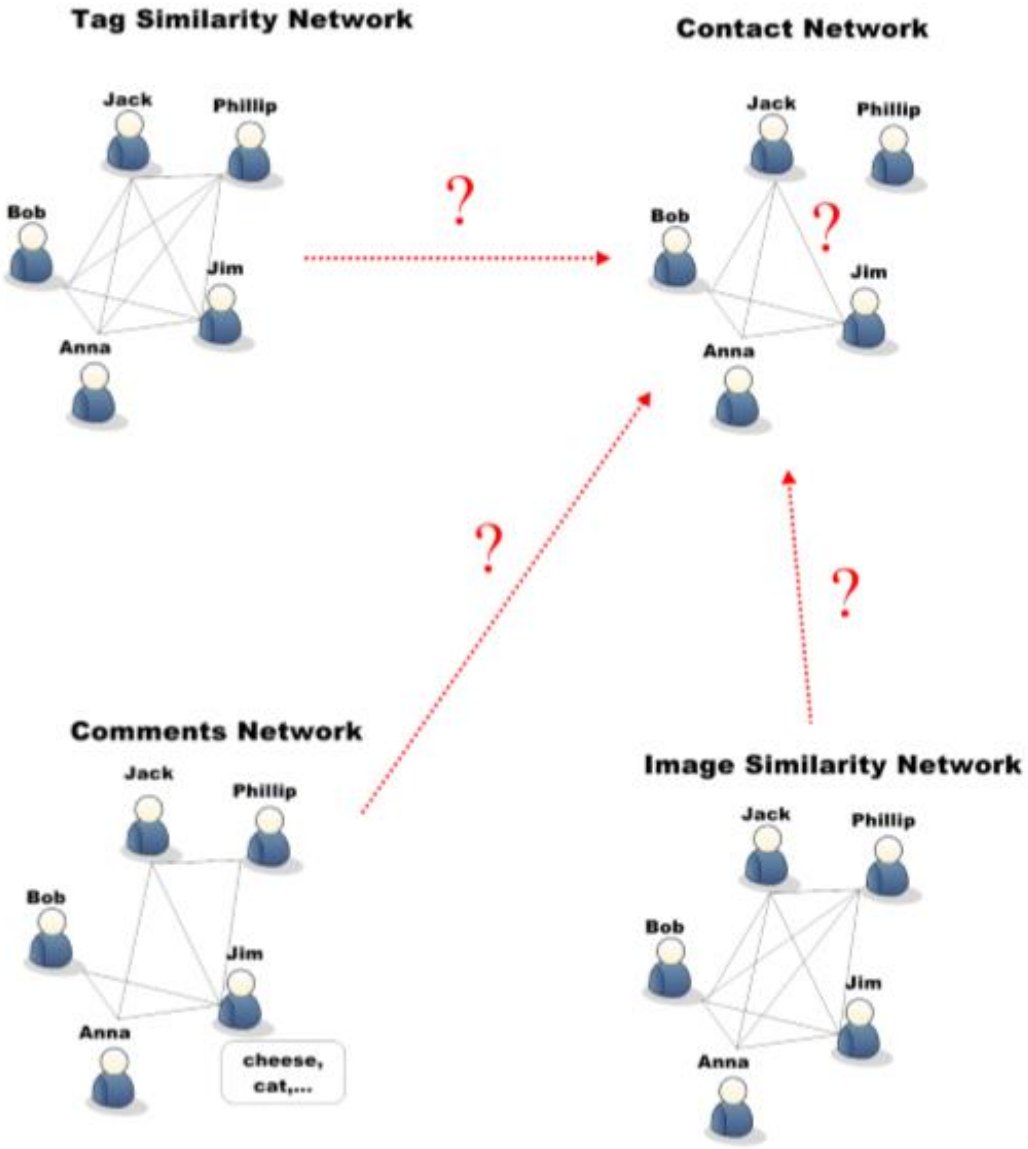
Output:

Friend recommendation list

1: Determine λ, μ and p via cross validation on training set
2: Calculate the tag relationship matrix $\mathbf{L}$
3: Calculate $\mathbf{V}$ by eigen-decomposition of Laplacian of $\mathbf{K}$
4: Initialize $\mathbf{B}$ with identity matrix $\mathbf{I}$
5: **repeat**
6: Calculate $\mathbf{W}$ by eq.(12) or eq.(13)
7: Calculate $\mathbf{B}$ by eq.(10)
8: **until** $Convergence$
9: Calculate the norm of each row of $\mathbf{W}$. Rank the norms in a descending order.
10: Choose important features from top of the ranking list.
11: Calculate the similarities between the important features of the new user and those of the existing users.
12: Top K similar users are recommended as friends to the new user.

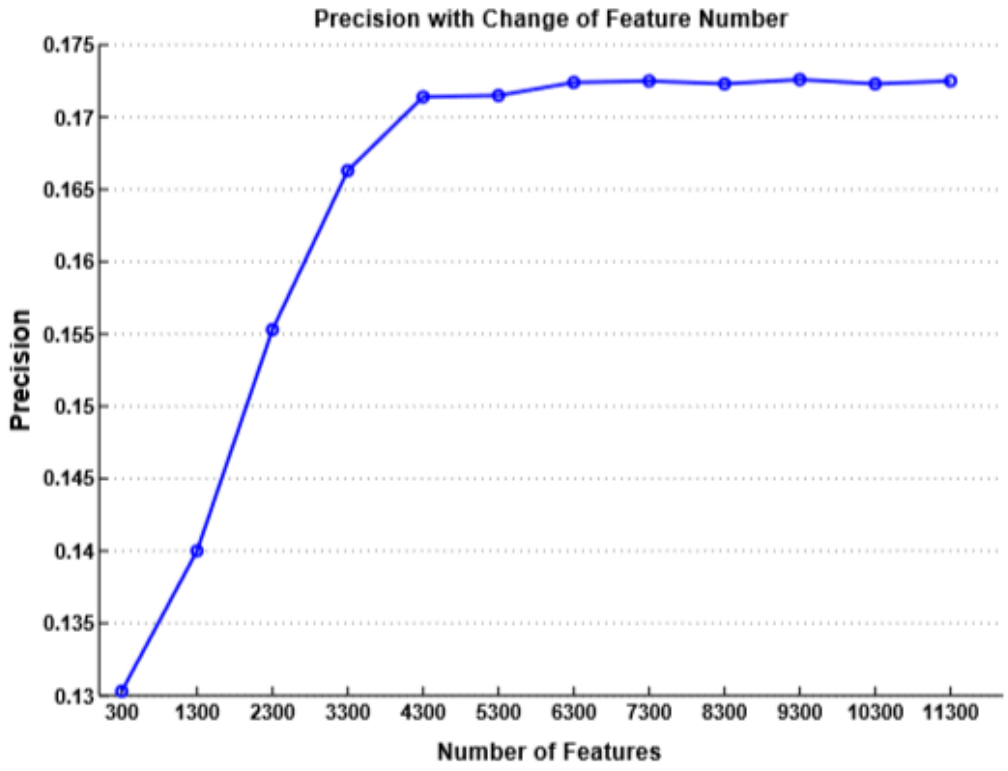
CHAPTER 2 FIGURE 8: PROPOSED ALGORITHM

Huang and his team all designed a system model, as illustrated below in Figure 9. Taking the Flickr network as a prime example, there are several features that they offer, including tag features, comment features, geo features, image features, and favourite picture features. These different features create different networks. Based off of these networks, new user friendships can be predicted and more networks can be connected to create a combined interpretation.



CHAPTER 2 FIGURE 9: HOW CAN MULTIPLE NETWORKS ARE CORRELATED

The experimental results are as follows. First, the algorithm runs on a different number of features. Then the selected number of features is changed from a starting point of 300 up to 12000, as illustrated in Figure 10. Until 4300 there is a rapid increase in the rate of precision, but later onwards the precision rate varies only slightly until the endpoint of 12000 is reached. These features are tested and for each user on each feature, the first twenty similar users are recommended. The accuracy of recommendation increases distinctively with the increase of features. After a certain point, it becomes stable.



CHAPTER 2 FIGURE 10: EXPERIMENTAL RESULTS

2.3 Gamification Concepts

2.3.1 Gamification in Education

Nowadays, most methods for traditional schooling are existing but very much considered as boring and ineffective. Lecturers teaching at any level of schooling are continuously looking for new ways to engage and motivate students while maintaining learning standards and personal improvement in the skillset (Lee and

Hammer, 2011). While attempting to find effective teaching methods, it was observed that students react well to competition and games. Games have a strange motivational power due to its active engagement and it encourages pupils to participate in activities, for not only the joy of playing but also for the desire to win. Gamification, especially in education is considered as one of the popular and top growing trends (Dicheva et al., 2015).

A careful and systematic mapping design was carried out to answer two questions; "What educational contexts has Gamification been applied to?" and "What game elements have been used in gamifying educational systems?". Gamification design principles in the educational sector have been identified and the game mechanisms that have been used to identify them are demonstrated in Table 2 below.

CHAPTER 2 TABLE 2: GAMIFICATION DESIGN PRINCIPLES WITH REGARD TO EDUCATION - 1

Design Principles	**Used Game Mechanics**	**Papers**
Goals: specific, clear, moderately difficult, immediate goals		(Lee & Hammer, 2011) (Kapp, 2012)
Challenges and quests: clear, concrete, actionable learning tasks with increased complexity		(Lee & Hammer, 2011) (Zichermann & Cunningham, 2011) (Deterding, 2013) (Simões, Díaz, & Fernández, 2013)
Customization: personalized experiences, adaptive difficulty; challenges that are perfectly tailored to the player's skill level, increasing the difficulty as the player's skill expands		(Lee & Hammer, 2011) (Zichermann & Cunningham, 2011) (Simões, Díaz, & Fernández, 2013) (Gordon, Brayshaw, & Grey, 2013)
Progress: visible progression to mastery	Points, Progress bars, Levels, Virtual Goods/ Currency	(Zichermann & Cunningham, 2011)
Feedback: immediate feedback or shorten feedback cycles; immediate rewards instead of vague long-term benefits		(Lee & Hammer, 2011) (Nah, et al., 2014) (Zichermann & Cunningham, 2011) (Kapp, 2012) (Simões, Díaz, & Fernández, 2013)

CHAPTER 2 TABLE 3: GAMIFICATION DESIGN PRINCIPLES WITH REGARD TO EDUCATION - 2

		(Gordon, Brayshaw, & Grey, 2013)
Competition and cooperation / social engagement loops	Badges, Leaderboards, Levels, Avatars	(Zichermann & Cunningham, 2011) (Iosup & Epema, 2014) (Deterding, 2013) (Simões, Díaz, & Fernández, 2013)
Accrual grading	Points	(Simões, Díaz, & Fernández, 2013)
Visible status: reputation, social credibility and recognition	Points, Badges, Leaderboards, Avatars	(Lee & Hammer, 2011) (Deterding, 2013) (Simões, Díaz, & Fernández, 2013)
Access/Unlocking content		(Iosup & Epema, 2014)
Freedom of choice: multiple routes to success, allowing students to choose their own sub-goals within the larger task		(Lee & Hammer, 2011) (Iosup & Epema, 2014) (Deterding, 2013) (Simões, Díaz, & Fernández, 2013)
Freedom to fail: low risk from submission, multiple attempts		(Lee & Hammer, 2011) (Kapp, 2012) (Deterding, 2013) (Gordon, Brayshaw, & Grey, 2013)
Storytelling	Avatars	(Nah, et al., 2014) (Kapp, 2012) (Simões, Díaz, & Fernández, 2013)
New identities and/or roles	Avatars	(Lee & Hammer, 2011) (Simões, Díaz, & Fernández, 2013)
Onboarding		(Zichermann & Cunningham, 2011) (Iosup & Epema, 2014)
Time restriction	Countdown clock	(Kapp, 2012)

Majority of the researches conducted have concluded that students were observed to be significantly much more engaged in projects, forums, and other daily learning activities. With that, attendance had also increased. Students' contributions increased all while holding up the quality as well. More students were succeeding with higher grades. Researches have also concluded that students consider the gamified versions much more interesting, motivating and easier to learn compared to the other traditional courses.

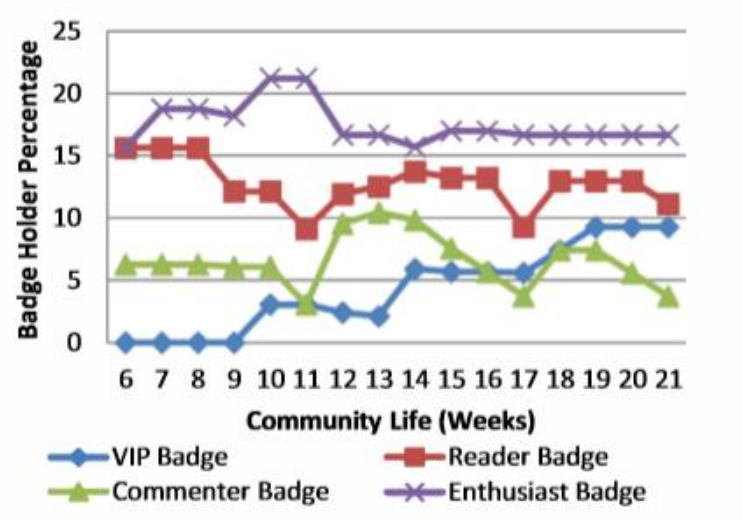
2.3.2 Gamification in an Online Community

To ensure that all members of the community contribute equally to the duties, a gamified website was designed. Members of the website were assigned tasks and duties and were rewarded badges based on whether or not they have successfully completed the tasks. A few of the tasks are listed below in Figure 11.

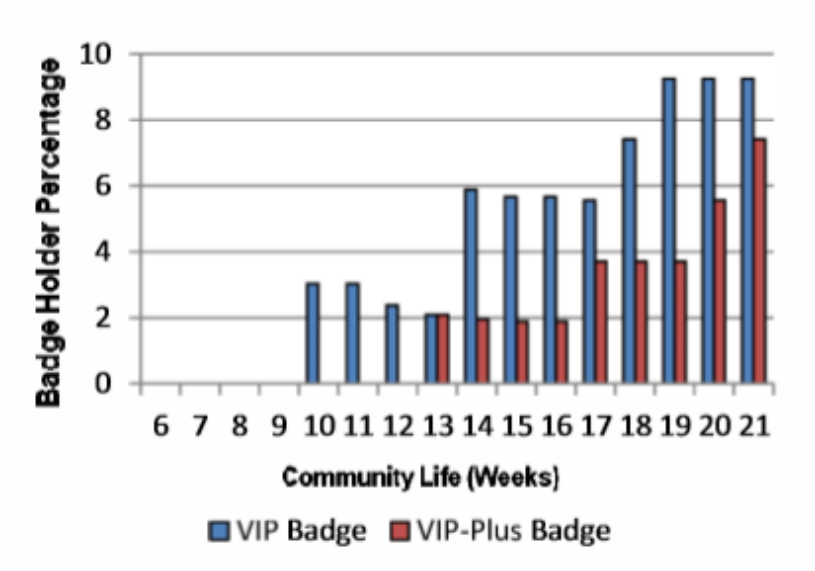
Gamification Context (c)	Action (a)
Registration	Complete Registration Early Bird registration
Visit	Unique Sign-In
Forum	Commenting Reading Rating
Buddy	Send Invitation Accept Invitation
Activity	Complete Standalone Activities Complete Collaborative Activities
Toolkit	Visit Resources Rate Resources Feedback on Resources Comment on Resources Comment Ratings
Live Chat	Participating in Live Chat Suggesting Topics for Live chat
Myth Buster	Visit Myth Buster Page Myth Buster Rating, Feedback and Commenting

CHAPTER 2 FIGURE 11: GAMIFICATION CONTEXTS AND ACTIONS

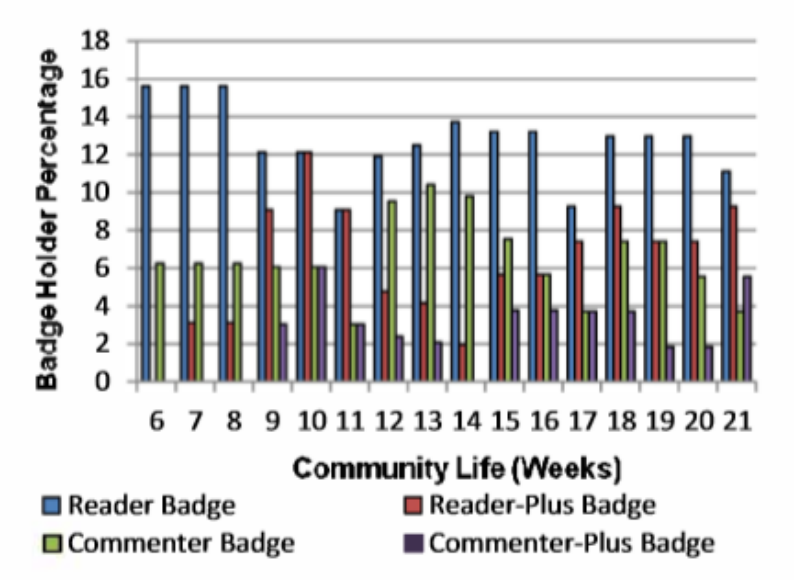
The following diagrams illustrate the statistics the researchers have received since the start of the gamified website.



CHAPTER 2 FIGURE 12: BADGES AND BADGE TYPES EARNED DURING THE FIRST 4 MONTHS



CHAPTER 2 FIGURE 13: REINFORCEMENT BADGE COMPARISON; VIP-VIP PLUS

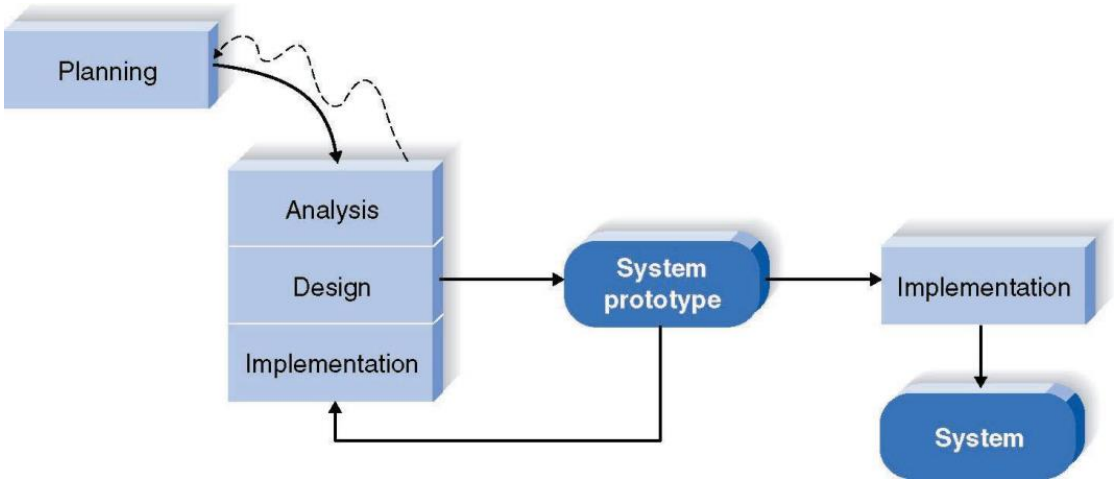


CHAPTER 2 FIGURE 14: COMPARISON OF REINFORCEMENT BADGE POPULATION

The badge data in the above graphs show how the activities were picked and presents the status of the members; new or recurrent. From these results, it can be observed that Gamification data has proved to be useful in monitoring the community's behaviour. Active and passive behaviours of the users can be monitored and analysed using this data. The results also portray the percentage of the community that maintains a specific behaviour, which is important when it comes to analysing online community sustainability.

3. Methodology

A formalized approach has been used to develop and implement the system. Several methodologies were considered and the most viable option was narrowed down. The Prototype methodology was used in this scenario. The reason behind following this particular methodology is the need for users to engage in the process of feedback and evaluation sessions, in order to gain constructive criticism and use it to improve the system. This methodology allows for constant evaluation and improvement. Additionally, users can feel constantly engaged and reassurance is given constantly to them that their feedback is crucial for the success of the system. The figure below (Figure 1) illustrates the stages of the prototyping model.



CHAPTER 3 FIGURE 1: PROTOTYPE METHODOLOGY

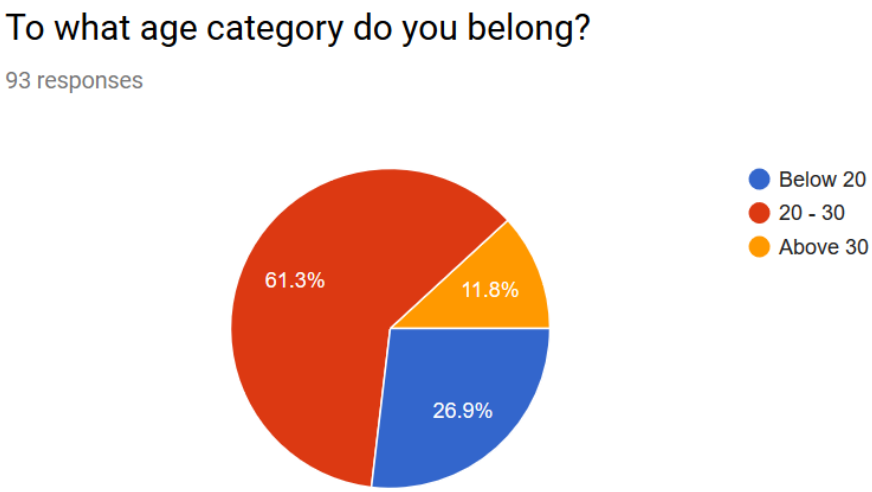
3.1 Requirement Gathering & Analysis

Prior to actually designing and implementing the system, user requirements and expectations had to be gathered. This was done in two forms, primary and secondary. Primary data gathering was done in the form of a market research via a questionnaire, while secondary data gathering was done in the form of a literature review by analysing relevant research journals and articles, and is explored in section two of the thesis.

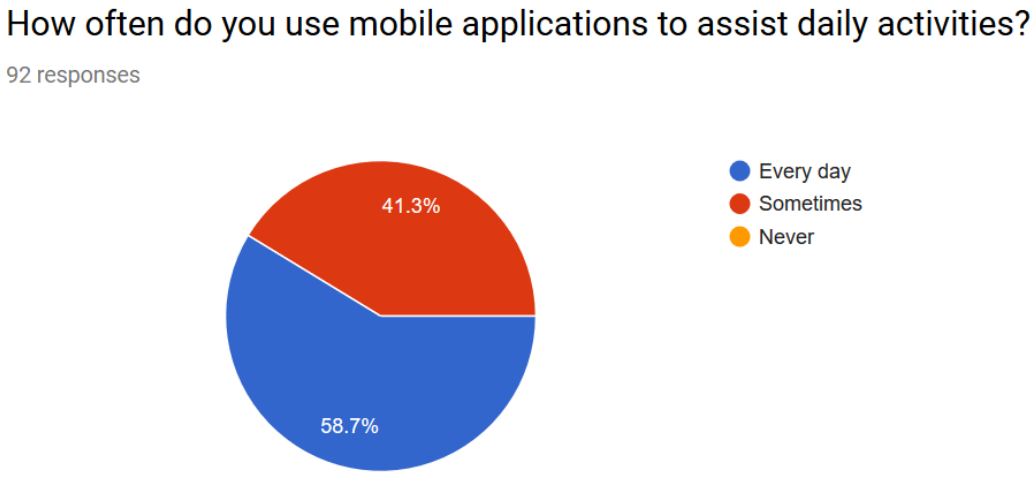
Below is a segment of the questionnaire that was distributed and analysed in order to process the requirement more clearly. The questionnaire is based on research for

an all-in-one mobile application, which would consist of features such as communication, navigation, and entertainment.

The responses to these, questions will help to determine the demand for such a product.



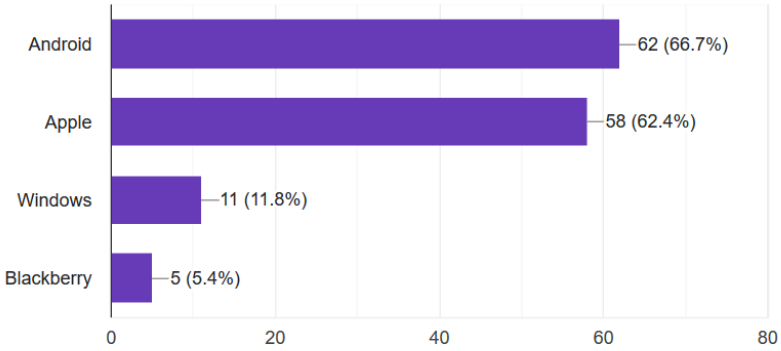
CHAPTER 3 FIGURE 2: QUESTIONNAIRE RESULT 1



CHAPTER 3 FIGURE 3: QUESTIONNAIRE RESULT 2

What is/are your operating system preference(s) for your mobile device?

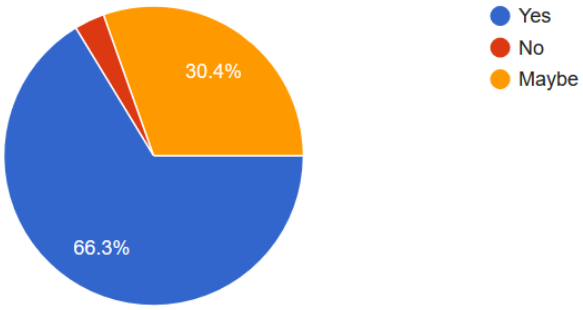
93 responses



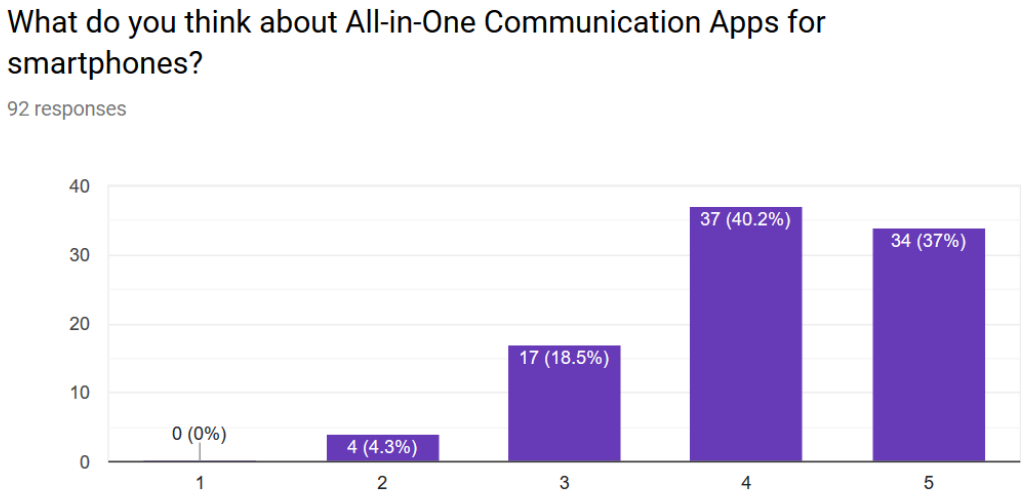
CHAPTER 3 FIGURE 4: QUESTIONNAIRE RESULT 3

Would you consider using a mobile application that gives directions based on your daily schedule?

92 responses



CHAPTER 3 FIGURE 5: QUESTIONNAIRE RESULT 4



CHAPTER 3 FIGURE 6: QUESTIONNAIRE RESULT 5

Majority of the users took part in this questionnaire were aged between 20-30, at a rate of 61.3%, followed by 26.9% of people who were younger than 20, as can be seen in Figure 2. It is safe to say that the people who answered the next few questions were young, impressionable adults. From this sample space, more than half, 58.7%, claimed to use mobile applications often to facilitate their daily activities (Figure 3). The remaining of the sample space said they only use these apps sometimes. However, it is noteworthy that there was no one who claimed to never use apps to assist in their daily tasks. This goes to prove how much, in this day and age, one depends on mobile applications on a daily basis to go about their day. People also voted Android as their most preferred operating system, which was closely followed by Apple (Figure 4). Windows and Blackberry were two of the least preferred comparatively. When the users were asked whether they would be open to using a navigation system that would study the user's daily movements and give directions based off of that, 66.3% of the sample space said they would definitely consider using such an application, 30.4% answered "Maybe" and the remaining small amount answered with a "No" (Figure 5). The people were finally asked what they think of an All-in-One communication system and were asked to rate it on a scale of 1 to 5; 1 being not helpful at all, and 5 being extremely helpful (Figure 6). As illustrated in the figure, it can be seen that the majority identify such an application to be Very Helpful (40.2%) and Extremely Helpful (37%). There were no votes for

"Not Helpful". The remaining sample claimed the application to be either moderately helpful (18.5%) or slightly helpful (4.3%).

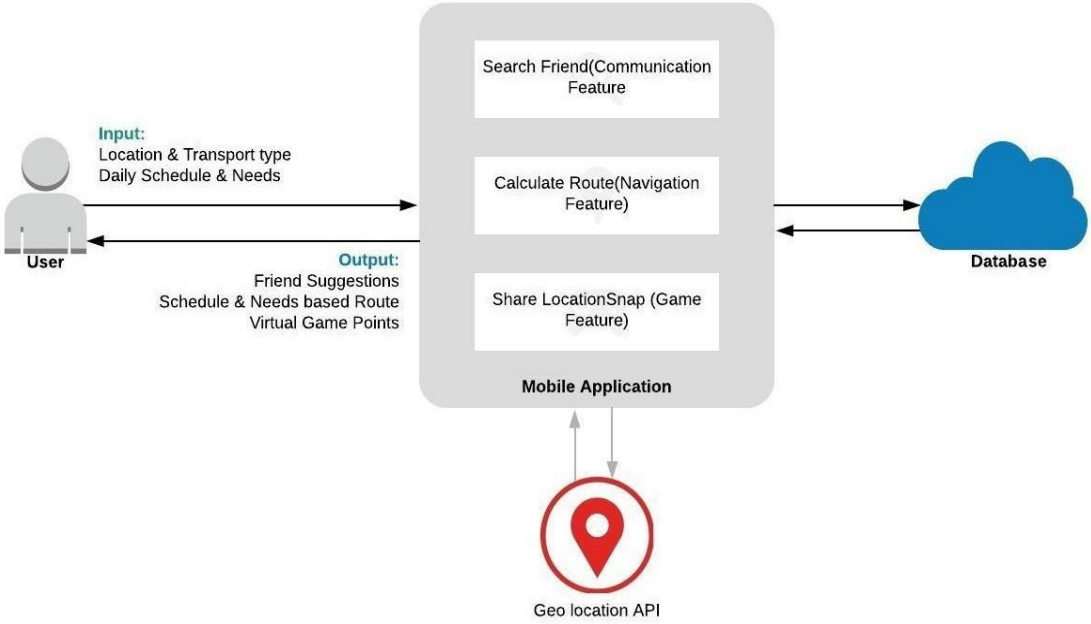
From this market research, it was clear that society would indeed benefit from an all-in-one communication system. The platform to implement this system, by popular vote, would be Android, and the targeted users are mostly teenagers and young adults.

3.2 Design

The Design phase consists of a high-level architecture, along with diagrams such as Use Case, Class and Entity Relationship Diagrams to illustrate the overall functionality of the system.

3.2.1 High-Level Design

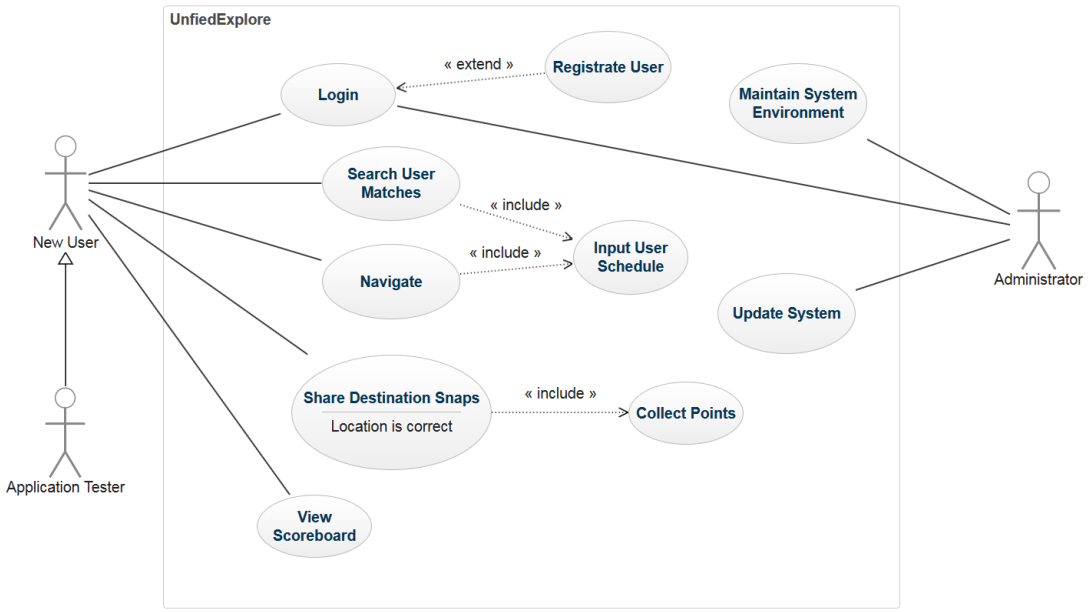
Figure 7 below delivers a brief overview of the proposed system while identifying all the main components that are intended to be developed for the final product. With the development the design started to vary from its initial thought. Few variables changed regarding the input and output that include the transport type and daily schedule and needs.



CHAPTER 3 FIGURE 7: HIGH-LEVEL ARCHITECTURE DIAGRAM

3.2.2 Use Case Diagram

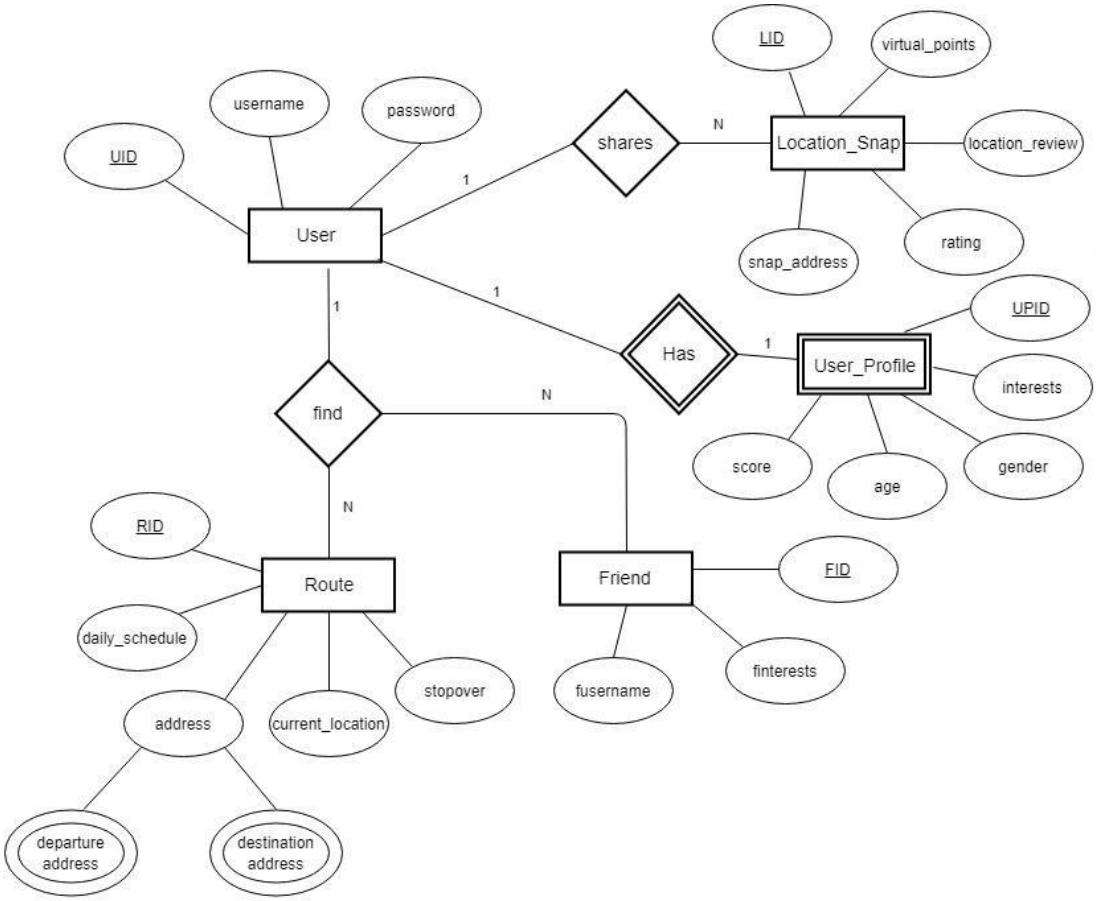
The below figure 8 shows a simple representation of how the users interact with the system by showing the relationships between the actors(users) and the different use cases in which all the actors are involved.



CHAPTER 3 FIGURE 8: USE CASE DIAGRAM

3.2.3 Entity Relationship Diagram

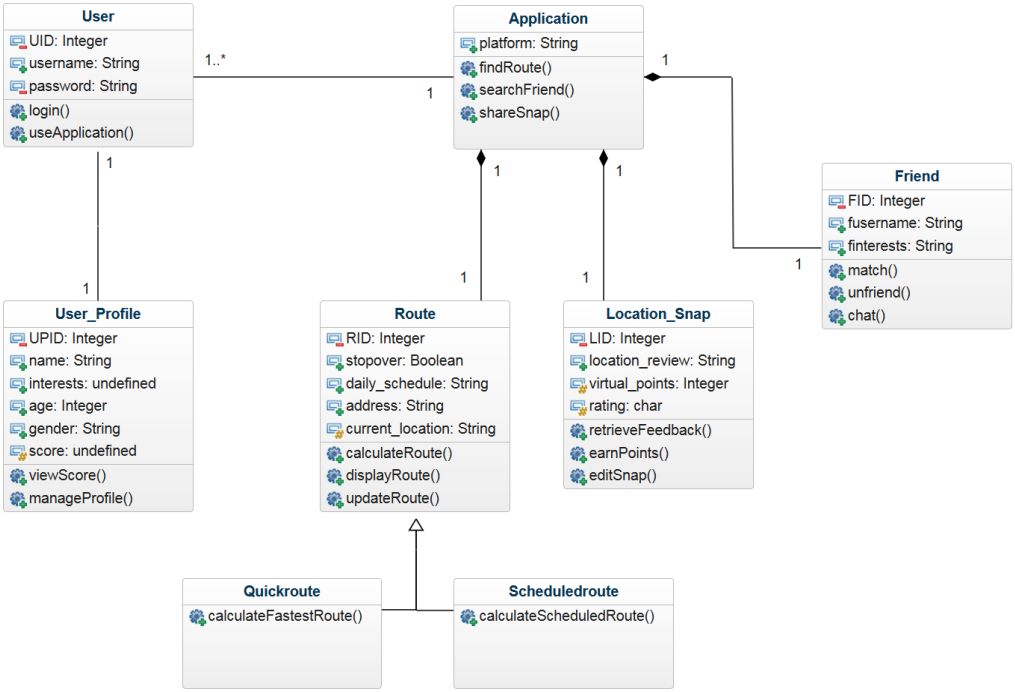
The bellows figure (Figure 9) describes and illustrates how the entities are interrelated within the Unified Explore domain and states the relationships existing with those entities.



CHAPTER 3 FIGURE 9: ENTITY RELATIONSHIP DIAGRAM

3.2.4 Class Diagram

This following figure (Figure 10) describes the structure of the system in showing the systems classes, their operations, various attributes it contains, and the relationships among them.



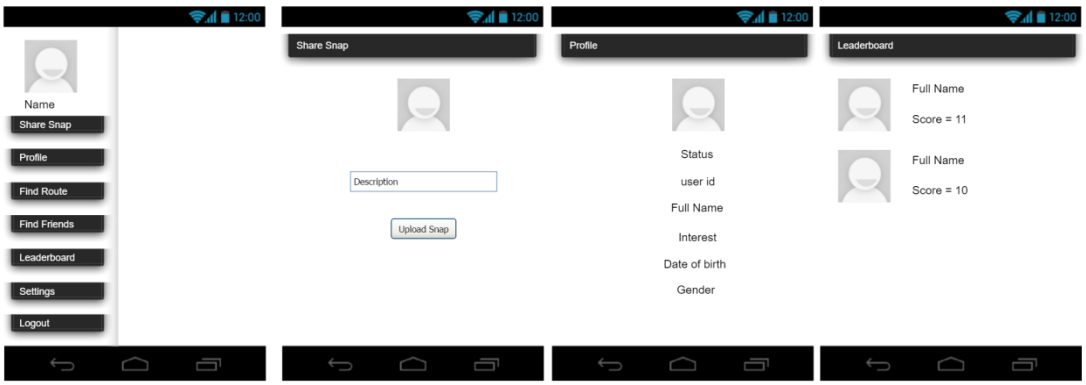
CHAPTER 3 FIGURE 10: CLASS DIAGRAM

3.2.5 Interface Design

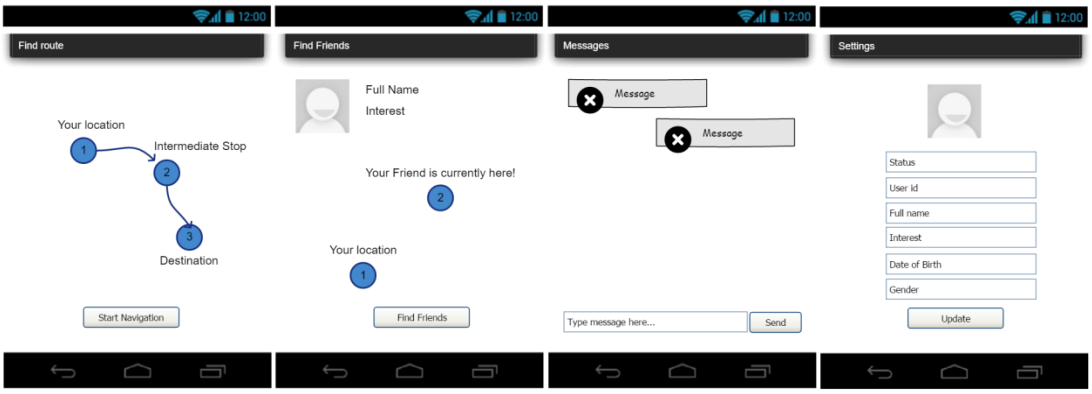
The proposed system runs on an android based platform with the aim to be used as a mobile application. Therefore, the interfaces are used to manage and navigate throughout the system. The designs of the interfaces were done at the start in a minimalistic and simple way for the ease of understanding. Later on the actual interfaces for the system oriented on the following displayed and were designed similarly but with a more creative and user-friendly visual look on Android Studio.

CHAPTER 3 FIGURE 11: DESIGN INTERFACES – 1

The above interfaces (Figure 11) show the initial design for the login procedure to access the system and sign up procedure for the creation and profile set up on a new account.



CHAPTER 3 FIGURE 12: DESIGN INTERFACES – 2



CHAPTER 3 FIGURE 13: DESIGN INTERFACES – 3

Figure 12 to 13 show example views of the entire interfaces that will be available when the user either created and set up an account successfully or simply logs in to the system using his or her credentials. The navigation bar on the interface on the left-hand side displays the users set profile picture and his full name. Below the name, the entire feature the system provides is listed inside the navigation bar. These features are as follows: Share Snap, Profile, Find Route, Find Friends, Leaderboard, Settings, and Logout.

3.3 Implementation

3.3.1 Technologies used for the implementation

In the following various technologies will be listed and explained that were utilized in order to implement the proposed system.

- Java

  Java is up-to-date one of the most desired and recognised programming languages. It was first developed by Sun Microsystems but now is owned by Oracle. Java is object-oriented and incorporates features such as security and platform independency. Multiple android applications these days are developed using the programming language Java or make use of its principles. Since android relies heavily on Java fundamentals, the Android SDK includes many standard java libraries that were helpful when developing the proposed system (W3schools.com, 2019).

- Android Studio

  This is an IDE designed explicitly for the development of Android applications and based on IntelliJ IDEA by JetBrains. It provides following advantages that aided the development of the system, such as a virtual emulator, assistance in integrating Firebase to the application or by simply formatting code automatically as work was done. Moreover, Android Studio is available for Windows Operating System with the capability to use Java Development Kid and fulfilled all the requirements as a platform for the systems development (Android Developers, 2019).

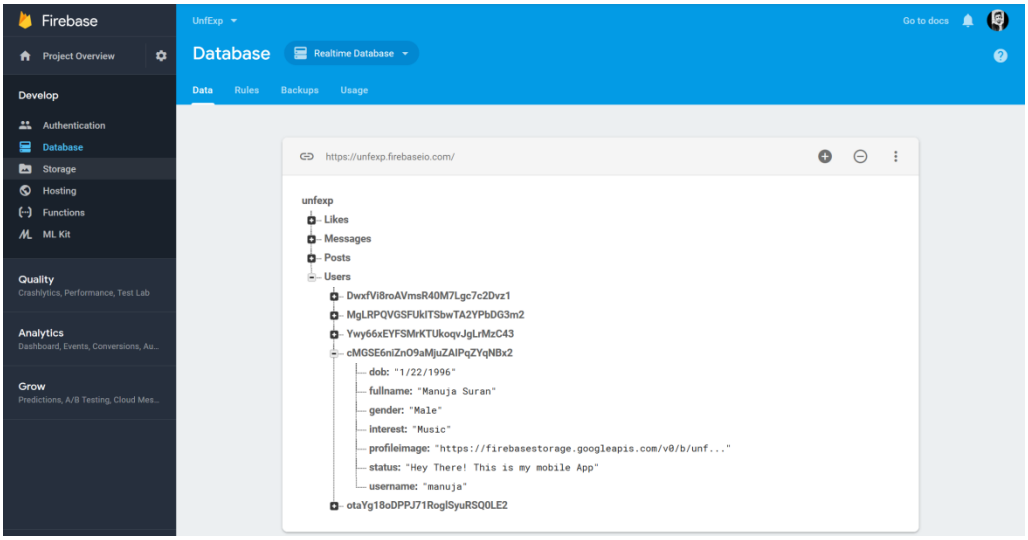
CHAPTER 3 FIGURE 14: ANDROID STUDIO LAUNCH INTERFACE

- Firebase

  Firebase is a platform that is owned by Google and allows building mobile applications without worrying about a server-side programming language. It allows storing user's data on its real-time database in which data changes can be done immediately. Furthermore, Firebase provides additional useful features that were utilized during the implementation.

  - Authentication – to manage users efficiently
  - Real-time database – to handle changes in data in immediately when requested
  - Storage – to store data by using firebase as a backend
  - Crash reporting – notifies whenever the app crashes

  As going in-depth with the development of the software artefact Firebase provided these most convenient solutions since it was simple to integrate and use with the application (Firebase, 2019).



CHAPTER 3 FIGURE 15: FIREBASE DATABASE SCREENSHOT

3.3.2 APIs and other libraries used

APIs and Libraries that were used to support and implement the functionalities of the system are listed and explained below.

- Mapbox API

  Mapbox APIs allow accessing Mapbox tools and services. Hence, the Navigation service and its Mapbox Directions API are utilized within the system to find directions and generate matching navigation instructions in order to build one of the functionalities of the system.

- Google Maps API

  Using the Maps SDK in this particular case for android, maps based on Google Maps latest available data is added to the application. The API manages access to servers of Google Maps and handles map display and response to map gestures. Hence, it is utilized to implement another function of the proposed system.

- Picasso Library

  This library mainly aids in terms of image processing. It gives the possibility to retrieve images from external sources and utilize them. It is very practical to use and can be implemented within a few lines of code. Hence, the Picasso library came very handy whenever it was needed to process and handle an image using the proposed system.

- Material Library

  This is an open source library that makes use of material design components on android. This helped me in implementing the checkbox feature in the Login page of the proposed system.

- Android Image-Cropper Library

  This is simple open source library that provides a cropping functionality for the optimization of images retrieved from the Gallery or Camera of a mobile device. Thus, images can be costumed by shape, zoom level and more.

3.3.3 Relevant code segments

In the Code segment below (Figure 16) a method is written to validate under what conditions the user is allowed to log in to the system. This method is called if the user intends to login to the system. In total this validation includes six conditions.

1. A Condition in case the user has selected the checkbox to retrieve the email and password from the General.class (Figure 37) to set its stored values which act as a key as input to the relevant email and password field in the PermitUserToLogin() Method
2. Two conditions to check, if either the email field or password field is left empty and in that case to prompt a message accordingly, such as “Please enter <name of field>”
3. The last condition is executed in case all other previous conditions are false. Here there are two options. One, it allows the user to login to the system if the credentials match with the values stored in the database. Otherwise, the system prompts a message that the credentials are wrong and to try again

```
private void PermitUserToLogin() {
    String email = UserEmail.getText().toString();
    String password = UserPassword.getText().toString();

    if (UserCheckBox.isChecked()) {
        Paper.book().write(General.UserEmailKey, email);
        Paper.book().write(General.UserPasswordKey, password);
    }

    if (TextUtils.isEmpty(email)) {
        Toast.makeText(this, "Please enter email.", Toast.LENGTH_SHORT).show();
    } else if (TextUtils.isEmpty(password)) {
        Toast.makeText(this, "Please enter password.", Toast.LENGTH_SHORT).show();
    } else {
        LoadingBar.setTitle("Login");
        LoadingBar.setMessage("Please wait, Login is in process...");
        LoadingBar.show();
        LoadingBar.setCanceledOnTouchOutside(true);

        mAuth.signInWithEmailAndPassword(email, password)
                .addOnCompleteListener(new OnCompleteListener<AuthResult>() {
                    @Override
                    public void onComplete(@NonNull Task<AuthResult> task) {
                        if (task.isSuccessful()) {
                            SendUserToMainActivity();
                            Toast.makeText(LoginActivity.this, "You are now logged in
successfully.", Toast.LENGTH_SHORT).show();
                            LoadingBar.dismiss();
                        } else {
                            String message = task.getException().getMessage();
                            Toast.makeText(LoginActivity.this, "Credentials are wrong. Please
try again.", Toast.LENGTH_SHORT).show();
                            LoadingBar.dismiss();
                        }
                    }
                });
    }
}
```

CHAPTER 3 FIGURE 16: LOGIN METHOD (LOGIN ACTIVITY)

```
public class General {

    public static final String UserEmailKey = "UserEmail";
    public static final String UserPasswordKey = "UserPassword";
}
```

CHAPTER 3 FIGURE 17: GENERAL CLASS

Figure 38 below is a method that is written for the purpose to create a new account that is called once the “Create Account” Button is pressed on the UI of the system. The conditions check if either the email, password or confirm password are left empty and prompt messages accordingly if true. Another condition checks if both passwords entered match. In case the password do not match the user is prompt a message to check once again. The last condition is executed in case all other previous conditions are false. Here there are two options. One, it allows the user to create the account successfully so that the data entered by the user is stored to the database. Otherwise, the system prompts an appropriate error message.

```
private void CreateANewAccount() {
    String email = UserEmail.getText().toString();
    String password = UserPassword.getText().toString();
    String confirmPassword = UserConfirmPassword.getText().toString();

    if (TextUtils.isEmpty(email)) {
        Toast.makeText(this, "Please enter your email.", Toast.LENGTH_SHORT).show();
    } else if (TextUtils.isEmpty(password)) {
        Toast.makeText(this, "Please enter your password.", Toast.LENGTH_SHORT).show();
    } else if (!password.equals(confirmPassword)) {
        Toast.makeText(this, "Passwords do not match, check again.",
Toast.LENGTH_SHORT).show();
    } else if (TextUtils.isEmpty(confirmPassword)) {
        Toast.makeText(this, "Please confirm your password.", Toast.LENGTH_SHORT).show();
    } else {
        LoadingBar.setTitle("Creating new Account");
        LoadingBar.setMessage("Please wait, while your account is created...");
        LoadingBar.show();
        LoadingBar.setCanceledOnTouchOutside(true);

        mAuth.createUserWithEmailAndPassword(email, password)
                .addOnCompleteListener(new OnCompleteListener<AuthResult>() {
                    @Override
                    public void onComplete(@NonNull Task<AuthResult> task) {
                        if (task.isSuccessful()) {
                            SendUserToSetupActivity();
                            Toast.makeText(RegisterActivity.this, "Account created
successfully.", Toast.LENGTH_SHORT).show();
                            LoadingBar.dismiss();
                        } else {
                            String message = task.getException().getMessage();
                            Toast.makeText(RegisterActivity.this, "Error: " + message,
Toast.LENGTH_SHORT).show();
                            LoadingBar.dismiss();
                        }
                    }
                });
    }
}
```

CHAPTER 3 FIGURE 18: CREATE ACCOUNT METHOD (REGISTRATION ACTIVITY)

This method shown in Figure 19 is called for the purpose of the customization of the user's profile. The username, full name, and the interest are retrieved and through if conditions validations are given so that the fields cannot be left as empty when trying to complete the account creation. Messages are displayed accordingly if the user tries to leave a field empty.

If all fields have values and the method is called the else part of the condition will be executed. Using HashMap the user entered data is saved to the real-time database under the database reference that is assigned as UsersRef. If the task is successful a message is prompt to indicate that the account creation is successful. Otherwise, if the task is not successful and appropriate error messages are displayed.

```
private void SaveAccountSetupInformation() {
    String username = UserName.getText().toString();
    String fullname = FullName.getText().toString();
    String interest = Interest.getSelectedItem().toString();

    if (TextUtils.isEmpty(username)) {
        Toast.makeText(this, "Please enter your Username.", Toast.LENGTH_SHORT).show();
    }
    if (TextUtils.isEmpty(fullname)) {
        Toast.makeText(this, "Please enter your Full name.", Toast.LENGTH_SHORT).show();
    }
    if (TextUtils.isEmpty(interest)) {
        Toast.makeText(this, "Please enter your Interest.", Toast.LENGTH_SHORT).show();
    } else if (!TextUtils.isEmpty(username) && !TextUtils.isEmpty(fullname) &&
!TextUtils.isEmpty(interest)) {
        LoadingBar.setTitle("Saving Information");
        LoadingBar.setMessage("Please wait, while we are creating your new account...");
        LoadingBar.show();
        LoadingBar.setCanceledOnTouchOutside(true);

        HashMap userMap = new HashMap();
        userMap.put("username", username);
        userMap.put("fullname", fullname);
        userMap.put("interest", interest);
        userMap.put("status", "Hey There! This is my status");
        userMap.put("gender", "");
        userMap.put("dob", "");

        UsersRef.updateChildren(userMap).addOnCompleteListener(new OnCompleteListener() {
            @Override
            public void onComplete(@NonNull Task task) {
                if (task.isSuccessful()) {
                    SendUserToMainActivity();
                    Toast.makeText(SetupActivity.this, "Your account is created
successfully.", Toast.LENGTH_LONG).show();
                    LoadingBar.dismiss();
                } else {
                    String message = task.getException().getMessage();
                    Toast.makeText(SetupActivity.this, "Error Occured: " + message,
Toast.LENGTH_SHORT).show();
                    LoadingBar.dismiss();
                }
            }
        });
    }
}
```

CHAPTER 3 FIGURE 19: SAVE SETUP INFORMATION METHOD (SETUP ACTIVITY)

Figure 20 shows a so-called “On click listener” that is executed once the user presses the Image Button to select a LocationSnap. Once pressed an alert dialog box is used to display two options the user can select from. Depending on the selected option refer to Figure 21 for further explanation.

```
SelectPostImage.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        CharSequence options[] = new CharSequence[]{
                "Upload from Gallery",
                "Use Camera"
        };
        AlertDialog.Builder builder = new AlertDialog.Builder(PostActivity.this);
        builder.setTitle("Select Option");

        builder.setItems(options, new DialogInterface.OnClickListener() {
            @Override
            public void onClick(DialogInterface dialog, int which) {
                if(which == 0){
                    OpenGallery();
                }
                if(which==1){
                    openCamera();
                }
            }
        });
        builder.show();
    }
});
```

CHAPTER 3 FIGURE 20: IMAGE SELECTION PART1 (POST ACTIVITY)

If the user selected the first option (Upload from Gallery) the openGallery() Method is called with its intent. Otherwise, if the user selected the second option (Use Camera) the openCamera() Method is called instead with its intent. Next, the onActivityResult() Method is called once the image is retrieved. Depending on the condition if it's either an image selected using the gallery or an image captured by the camera will be determined through an if-else statement. The retrieved image is then inputted to the ImageView named "SelectedPostImage".

```
private void openGallery() {
    Intent galleryIntent = new Intent(Intent.ACTION_GET_CONTENT);
    galleryIntent.setType("image/*");
    startActivityForResult(galleryIntent, IMAGE_PICK);
}

private void openCamera() {
    ContentValues values = new ContentValues();
    values.put(MediaStore.Images.Media.TITLE,"New Picture");
    values.put(MediaStore.Images.Media.DESCRIPTION,"From the Camera");
    ImageUri =
getContentResolver().insert(MediaStore.Images.Media.EXTERNAL_CONTENT_URI,values);
    Intent cameraIntent = new Intent(MediaStore.ACTION_IMAGE_CAPTURE);
    cameraIntent.putExtra(MediaStore.EXTRA_OUTPUT, ImageUri);
    startActivityForResult(cameraIntent, CAMERA_PICK);
}

@Override
protected void onActivityResult(int requestCode, int resultCode, Intent data) {
    super.onActivityResult(requestCode, resultCode, data);

    if (requestCode == IMAGE_PICK && resultCode == RESULT_OK && data != null) {
        ImageUri = data.getData();
        SelectPostImage.setImageURI(ImageUri);
    }
    else {
        SelectPostImage.setImageURI(ImageUri);
    }
}
```

CHAPTER 3 FIGURE 21: IMAGE SELECTION PART2 (POST ACTIVITY)

From the onMapClick() method displayed in Figure 22, we retrieve a Latitude/Longitude coordinate point. In this method when the user taps on a specific location a destination position is set, the origin position is retrieved and a marker is set on that particular destinations point.

Additionally, conditions are set to handle intermediate destinations that are specified here as waypoints. Therefore if more than one destination point exists the getRoute() function is called that retrieves additionally a waypoint as well to set the final route. Otherwise, in the else-statement, a similar getRoute() function is called that retrieves only the origin position and the destination position instead to set the final route.

```
@Override
public void onMapClick(@NonNull LatLng point) {

    boolean hasMoreThanOnePoint = false;
    if (destinationMarker != null) {
        currentWayPoints.add(destinationPosition);
        map.removeMarker(destinationMarker);
        hasMoreThanOnePoint = true;
    }

    destinationMarker = map.addMarker(new MarkerOptions().position(point));

    destinationPosition = Point.fromLngLat(point.getLongitude(), point.getLatitude());
    originPosition = Point.fromLngLat(originLocation.getLongitude(),
originLocation.getLatitude());
    if (hasMoreThanOnePoint) {
        getRoute(originPosition, destinationPosition, currentWayPoints);
    } else {
        getRoute(originPosition, destinationPosition);
    }

    startButton.setEnabled(true);
}
```

CHAPTER 3 FIGURE 22: SET DESTINATION POINT (FIND ROUTE ACTIVITY)

Below figure 23 shows the method to receive the specific location of a friend (another user) that has found an explorer (the current user). The latitude and longitude coordinates are retrieved of the friend and a marker is displayed on that friend's destination on a map with the description "This user has found you!".

```
private void GetFriendLocation() {
    AssignedFriendLocationRef = FirebaseDatabase.getInstance().getReference().child("Friend
Requests")
            .child(friendID).child("l");

    AssignedFriendLocationRefListner = AssignedFriendLocationRef.addValueEventListener(new
ValueEventListener() {
        @Override
        public void onDataChange(@NonNull DataSnapshot dataSnapshot) {
            if (dataSnapshot.exists()) {
                List<Object> friendLocationMap = (List<Object>) dataSnapshot.getValue();
                double LocationLat = 0;
                double LocationLng = 0;

                if (friendLocationMap.get(0) != null) {
                    LocationLat = Double.parseDouble(friendLocationMap.get(0).toString());
                }
                if (friendLocationMap.get(1) != null) {
                    LocationLng = Double.parseDouble(friendLocationMap.get(1).toString());
                }
                LatLng TravelerLatLng = new LatLng(LocationLat, LocationLng);
                PickUpMarker = mMap.addMarker(new
MarkerOptions().position(TravelerLatLng).title("This user has found
you!").icon(BitmapDescriptorFactory.fromResource(R.drawable.user)));
            }
        }

        @Override
        public void onCancelled(@NonNull DatabaseError databaseError) {

        }
    });
}
```

CHAPTER 3 FIGURE 23: GET FRIENDS LOCATION (EXPLORER MAP ACTIVITY)

This method is to find nearby Explorers (other users, which activated the explorer feature). To do so this method will get the unique id and location of that user from the reference that points to the "Explorers Available" node in the real-time database. If a unique id in the "Explorer Available" node exists, this indicates a user who activated the Explorer feature is currently online.

Furthermore, the GetClosestExplorers() method is a recursive function that calls itself if an Explorer is not nearby and his coordinates can't be found. Thus, the radius is incremented by 1 and the GetClosestExplorers() method is called again until an available Explorer is found. If the Explorer is found, the unique id is retrieved and another method GettingExplorerLocation() is called to retrieve the Explorers location, which is a similar method as the GetFriendLocation() which is displayed in the previous in Figure 24.

```
private void GetClosestExplorers() {
    GeoFire geoFire = new GeoFire(ExplorersAvailableRef);
    GeoQuery geoQuery = geoFire.queryAtLocation(new GeoLocation(FriendPickUpLocation.latitude,
FriendPickUpLocation.longitude),radius);
    geoQuery.removeAllListeners();

    geoQuery.addGeoQueryEventListener(new GeoQueryEventListener() {
        @Override
        public void onKeyEntered(String key, GeoLocation location) {
            if(!explorerFound && requestType)
            {
                explorerFound = true;
                explorerFoundID = key;

                ExplorersRef =
FirebaseDatabase.getInstance().getReference().child("Users").child(explorerFoundID);
                HashMap explorerMap = new HashMap();
                explorerMap.put("FriendMatchID", friendID);
                ExplorersRef.updateChildren(explorerMap);

                GettingExplorerLocation();
                FindFriendsButton.setText("Looking for Explorer Locations...");
            }
        }
        @Override
        public void onKeyExited(String key) {
        }
        @Override
        public void onKeyMoved(String key, GeoLocation location) {
        }
        @Override
        public void onGeoQueryReady() {
            if(!explorerFound){
                //check this statement
                radius = radius + 1;
                FindFriendsButton.setText("Search Radius is increased...");
                GetClosestExplorers();
            }
        }
        @Override
        public void onGeoQueryError(DatabaseError error) {
        }
    });
}
```

CHAPTER 3 FIGURE 24: GET CLOSEST EXPLORERS (EXPLORER ACTIVITY)

Once a user has typed a message and presses the send button/icon the SendMessage() method is being called. In this method, we, first of all, retrieve the message and next using and if else condition we check whether the text field is empty or not. It will prompt a message if the field is empty. Otherwise, the else-statement is entered. The message is stored under the Messages node in the database for both the Receiver of the message (messageRecieverID) and the Sender of the message (messagSenderID). Additionally, the date and time for the message are saved to the saveCurrentDate and saveCurrentTime variables. Then using a HashMap the text message is saved accordingly with other details, such as the date and time the message was sent, the type of the message and from whom the message was sent. Lastly, the if-else statement determines if the message was successfully sent and stored to the database under the correct reference or not and finally prompts a success or error message accordingly.

```
private void SendMessage() {
    String messageText = userMessageInput.getText().toString();

    if (TextUtils.isEmpty(messageText)) {
        Toast.makeText(this, "Please type a message first.", Toast.LENGTH_SHORT).show();
    } else {
        String message_sender_ref = "Messages/" + messageSenderID + "/" + messageRecieverID;
        String message_receiver_ref = "Messages/" + messageRecieverID + "/" + messageSenderID;

        DatabaseReference user_message_key = RootRef.child("Messages").child(messageSenderID)
                .child(messageRecieverID).push();
        String message_push_id = user_message_key.getKey();

        Calendar calFordDate = Calendar.getInstance();
        SimpleDateFormat currentDate = new SimpleDateFormat("dd-MMMM-yyyy");
        saveCurrentDate = currentDate.format(calFordDate.getTime());

        Calendar calFordTime = Calendar.getInstance();
        SimpleDateFormat currentTime = new SimpleDateFormat("HH:mm aa");
        saveCurrentTime = currentTime.format(calFordDate.getTime());

        Map messageTexBody = new HashMap();
        messageTexBody.put("message", messageText);
        messageTexBody.put("time", saveCurrentTime);
        messageTexBody.put("date", saveCurrentDate);
        messageTexBody.put("type", "text");
        messageTexBody.put("from", messageSenderID);

        Map messageBodyDetails = new HashMap();
        messageBodyDetails.put(message_sender_ref + "/" + message_push_id, messageTexBody);
        messageBodyDetails.put(message_receiver_ref + "/" + message_push_id, messageTexBody);

        RootRef.updateChildren(messageBodyDetails).addOnCompleteListener(new OnCompleteListen-
er() {
            @Override
            public void onComplete(@NonNull Task task) {
                if (task.isSuccessful()) {
                    Toast.makeText(ChatActivity.this, "Message send.",
Toast.LENGTH_SHORT).show();
                    userMessageInput.setText("");
                } else {
                    String message = task.getException().getMessage();
                    Toast.makeText(ChatActivity.this, "Error: " + message,
Toast.LENGTH_SHORT).show();
                    userMessageInput.setText("");
                }
            }
        });
    }
}
```

CHAPTER 3 FIGURE 25: SEND TEXT MESSAGES (CHAT ACTIVITY)

Once the user selects the Logout option in the navigation bar of the homepage menu the showLogoutPopup() method is called as shown in Figure 26. This method calls an alert dialog box that will appear asking the user if he or she is sure about wanting to Logout from the system. If “Logout” in the popup is selected the system will destroy the values stored in the General.class as shown in figure 17(chapter 3) so that when the user is sent back to the login page by calling the SendUserToLoginActivity() method the email and password field is set as empty. If the user selects the “Cancel” option in the popup, the user remains logged in the system and the popup closes.

```
private void showLogoutPopup() {
    AlertDialog.Builder builder = new AlertDialog.Builder(MainActivity.this,
R.style.Theme_MaterialComponents_Light_Dialog);
    builder.setMessage("Are you sure you want to Logout?")
            .setPositiveButton("Logout", new DialogInterface.OnClickListener() {

                public void onClick(DialogInterface dialog, int which) {

                    mAuth.signOut();
                    Paper.book().destroy();
                    SendUserToLoginActivity();

                }
            }).setNegativeButton("Cancel", null);

    AlertDialog alert = builder.create();
    alert.setIcon(R.drawable.logout);
    alert.show();
}
```

CHAPTER 3 FIGURE 26: LOGOUT METHOD (MAIN ACTIVITY)

Following Figure 27 shows the build version configuration that applies to the android project. The minimum required SDK version to run the application is 16 but can be updated if required, whereas the targeted SDK version is the current latest 28. Additionally, multidex had to be enabled to allow for further third-party libraries.

```
android {
    compileSdkVersion 28
    defaultConfig {
        applicationId "com.example.unfexp"
        minSdkVersion 16
        targetSdkVersion 28
        versionCode 1
        versionName "1.0"
        testInstrumentationRunner "android.support.test.runner.AndroidJUnitRunner"
        multiDexEnabled true
    }
    buildTypes {
        release {
            minifyEnabled false
            proguardFiles getDefaultProguardFile('proguard-android-optimize.txt'), 'proguard-
rules.pro'
        }
    }
    compileOptions {
        sourceCompatibility JavaVersion.VERSION_1_8
        targetCompatibility JavaVersion.VERSION_1_8
    }
}
```

CHAPTER 3 FIGURE 27: GRADLE APP MODULE

3.4 Testing and Results

This chapter will clarify the methods of testing and describe several test cases that were done for the system and their outcomes. To show the obtained results as evidence screen captures for test case scenarios are taken during the testing phase.

3.4.1 Methods of Testing

During the project duration, multiple procedures for the purpose of testing were done.

User acceptance testing was chosen as the first method for testing after the requirements for the system were gathered. Since a rough schedule and time plan was assigned with an enormous amount of workload only black box testing was utilized.

Secondly, Integration testing was used during the development stage of the prototype. After the results were analysed and the suggestions from the supervisor was obtained, the prototype was changed based on the feedback and the final prototype version was developed.

After finalizing the software artefact up to its completion, system testing will be used and functionality testing is done in order to check if the features and functions of the proposed system work accordingly. In the following, the test cases and their results are included.

3.4.2 Test cases

CHAPTER 3 TABLE 1: LOGIN - TC01

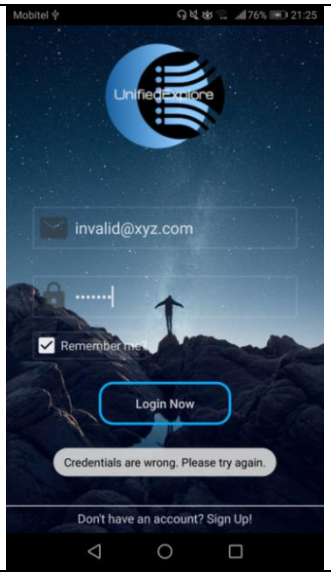
Test Case ID	Login – TC01
Test Case Description	Test the Login Functionality

S #	Prerequisites:
1	Launch Application
2	Launch Application

S #	Test Input
1	Email: invalid@xyz.com, Password: 1234567
2	Email: test@gmail.com, Password: 1234567

Test Scenario	Verify on entering relevant email and password, if the user can log in

Sequence #	Action	Expected Result(s)	Actual Result	Test Result
1	Enter invalid email & valid password and press login button	A message prompts: "Credentials are wrong. Please try again", User remains on the Login page	As Expected	Pass
2	Enter valid email & password and press login button	Login success, User is sent to home page interface	As Expected	Pass
Screenshot for Sequence #				
1				



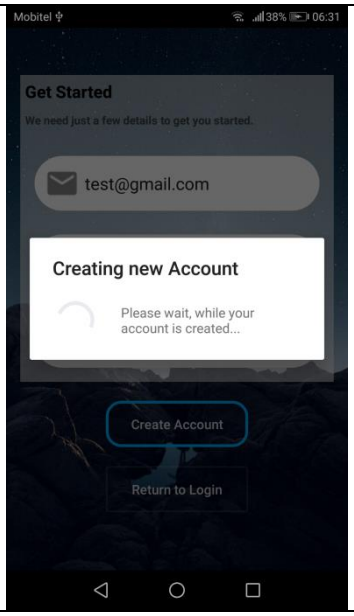
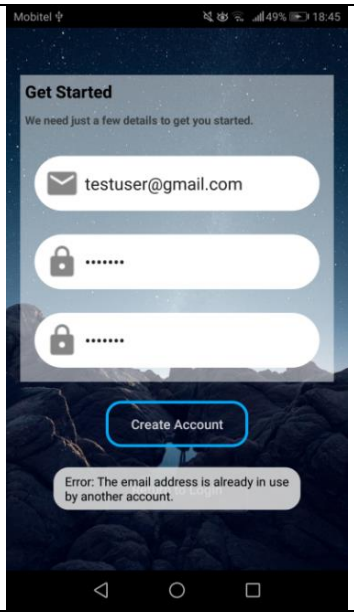
Test Case ID	Sign Up – TC02
Test Case Description	Test the Sign Up Functionality

S #	**Prerequisites:**
1	Launch Application
2	Launch Application, email assigned for an existing account

S #	**Test Input**
1	Email: test@gmail.com Password: 1234567-
2	Email: testuser@gmail.com Password: 1234567

Test Scenario	Verify on entering relevant user details, if an account is created

Sequence #	**Action**	**Expected Result(s)**	**Actual Result**	**Test Result**
1	Enter Test input and press the “Create Account” button	Prompts message that new account is creating	As Expected	Pass
2	Enter the email, password, confirm password and press “Create Account” button	If an account already exists, an error message is prompt	As Expected	Pass

Screenshot for Sequence #	
1	2
	

CHAPTER 3 TABLE 3: SETUP PROFILE - TC03

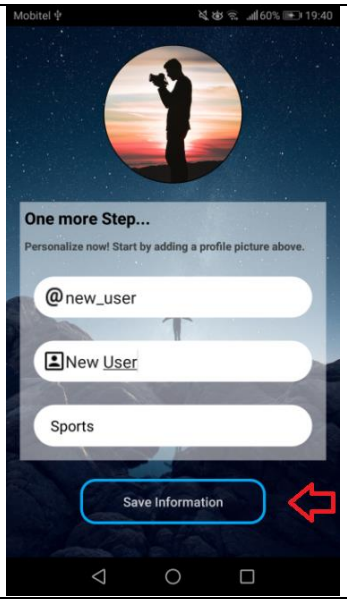
Test Case ID	Setup Profile – TC03
Test Case Description	Test the Setup Profile Functionality

S #	**Prerequisites:**
1	email assigned for an existing account
2	email not assigned for an existing account

S #	**Test Input**
1	Profile Picture: any
2	Profile Picture: any Username: new_user Full Name: New User Interest: Sports

Test Scenario	Verify on entering user personal details, if the account creation is being completed

Sequence #	**Action**	**Expected Result(s)**	**Actual Result**	**Test Result**
1	Enter Profile Picture, leave one field as empty, press "Save Information" button	Prompts message to input relevant field	As Expected	Pass
2	Enter test data and press "Save Information" button	The account is created successfully, User is sent to home page interface	As Expected	Pass

Screenshot for Sequence #
2


CHAPTER 3 TABLE 4: UPLOAD LOCATIONSNAP - TC04

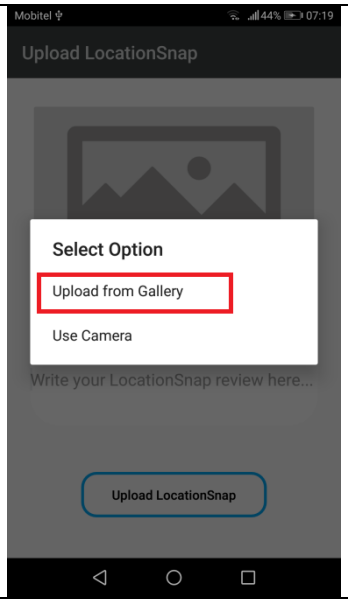
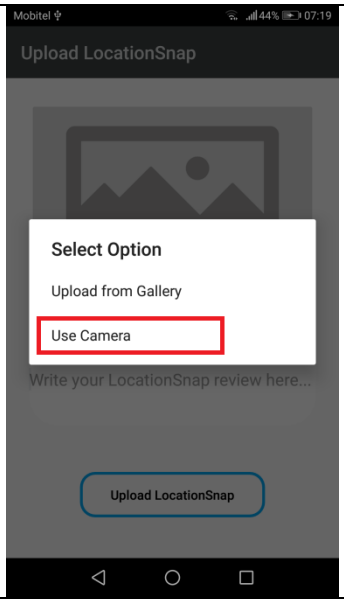
Test Case ID	Upload LocationSnap – TC04
Test Case Description	Test the Upload LocationSnap Functionality

S #	Prerequisites:
1	User logged in to account, Camera permission is enabled on the device
2	User logged in to account

S #	Test Input
1	Image, Description
2	Image

Test Scenario	Verify, if LocationSnaps are uploaded correctly

Sequence #	Action	Expected Result(s)	Actual Result	Test Result
1	Add LocationSnap using Gallery, enter description and press “Upload LocationSnap” button	success message is prompt, user is sent to the homepage where LocationSnap is displayed	As Expected	Pass
2	Add LocationSnap using Camera, enter description and press “Upload LocationSnap” button	success message is prompt, user is sent to the homepage where LocationSnap is displayed	As Expected	Pass

Screenshot for Sequence #	
1	2
	

CHAPTER 3 TABLE 5: PROFILE AND SETTINGS - TC05

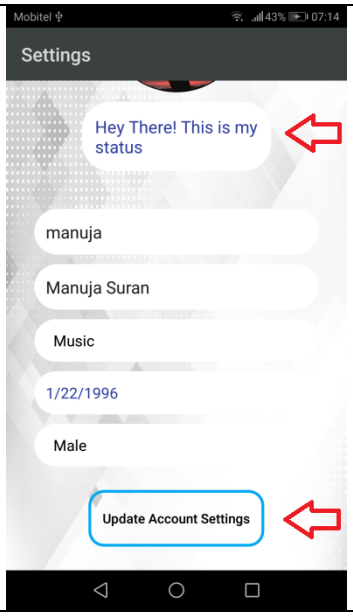
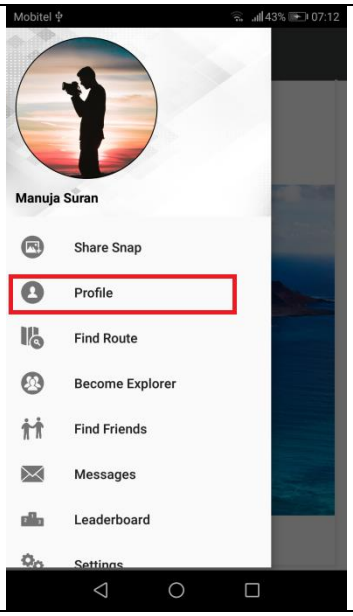
Test Case ID	Profile and Settings – TC05
Test Case Description	Test the Profile and Settings Functionality

S #	Prerequisites:
1	User logged in to account
2	User logged in to account

S #	Test Input
1	Personal information
2	none

Test Scenario	Verify, if Profile is displayed correctly after changing the Settings

Sequence #	Action	Expected Result(s)	Actual Result	Test Result
1	Change status field in Settings page and press update settings button	Prompts success message and User is sent to the homepage	As Expected	Pass
2	Select Profile on the navigation bar in the homepage and check if information is updated accordingly	Profile displays user details accordingly	As Expected	Pass

Screenshot for Sequence #	
1	2
	

CHAPTER 3 TABLE 6: FIND ROUTE - TC06

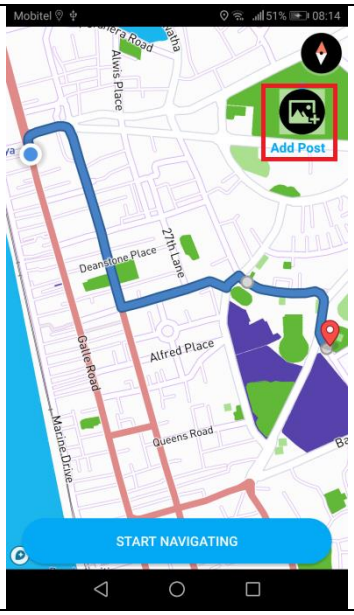
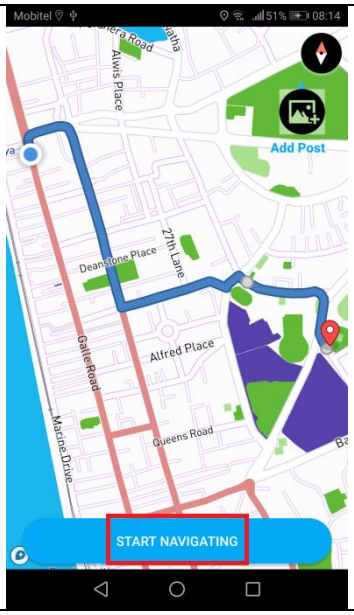
Test Case ID	Find Route – TC06
Test Case Description	Test the Find Route Functionality

S #	Prerequisites:
1	User logged in to account, GPS is enabled on the device
2	User logged in to account, GPS is enabled on the device

S #	Test Input
1	User Click
2	User Click

Test Scenario	Verify, if the user can set a custom route and use the navigation feature, as well as add posts

Sequence #	Action	Expected Result(s)	Actual Result	Test Result
1	Press “Add Post” icon on the map	User is sent to Upload LocationSnap feature	As Expected	Pass
2	Set a route by clicking on the map and press “Start Navigating” button	User is sent to Navigation UI, where route directions will be provided	As Expected	Pass

Screenshot for Sequence #	
1	2
	

CHAPTER 3 TABLE 7: FRIEND MATCHING - TC07

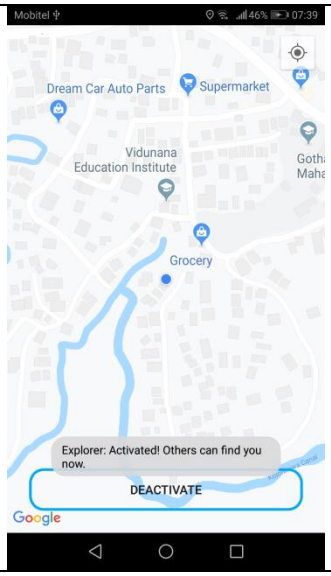
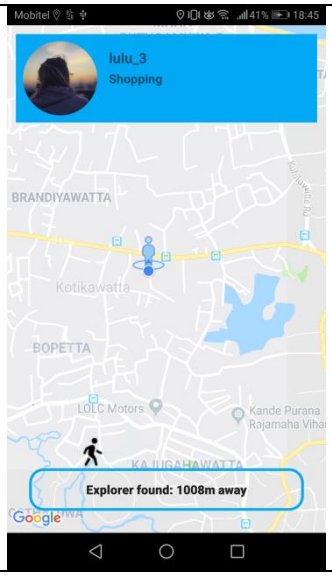
Test Case ID	Friend Matching – TC07
Test Case Description	Test the Friend Matching Functionality

S #	Prerequisites:
1	User logged in to account, GPS is enabled on the device
2	User logged in to account, GPS is enabled on the device

S #	Test Input
1	(Account 1)email: user1@gmail.com password: 1234567
2	(Account 2)email: user2@gmail.com password: 1234567

Test Scenario	Verify, if users can become Explorers and be found by other users with the Find Friend Feature

Sequence #	Action	Expected Result(s)	Actual Result	Test Result
1	Access Explorer feature on the navigation panel of the homepage	User is sent to the Explorer Interface, Message is prompt that the feature is activated	As Expected	Pass
2	Using another device select find Friends feature and press the “Find Friends” button	Other users location and personal details are displayed accordingly in each device	As Expected	Pass

Screenshot for Sequence #	
1	2
	

CHAPTER 3 TABLE 8: LEADERBOARD - TC08

Test Case ID	Leaderboard – TC08
Test Case Description	Test the Leaderboard Functionality

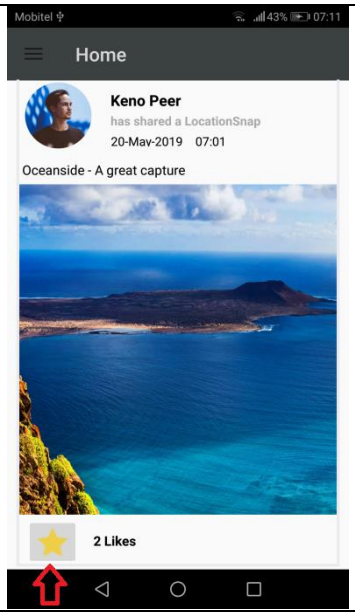
S #	Prerequisites:
1	User logged in to account

S #	Test Input
1	User presses like button

Test Scenario	Verify, if user posts with the top 5 highest scores are displayed correctly in the Leaderboard

Sequence #	Action	Expected Result(s)	Actual Result	Test Result
1	Like a post with the highest score in the homepage and check if the score is updated correctly in the Leaderboard	Leaderboard Score for the relevant post is updated correctly	As Expected	Pass

Screenshot for Sequence #
1



CHAPTER 3 TABLE 9: FRIEND MESSAGING - TC09

Test Case ID	Friend Messaging– TC09
Test Case Description	Test the Friend Messaging Functionality

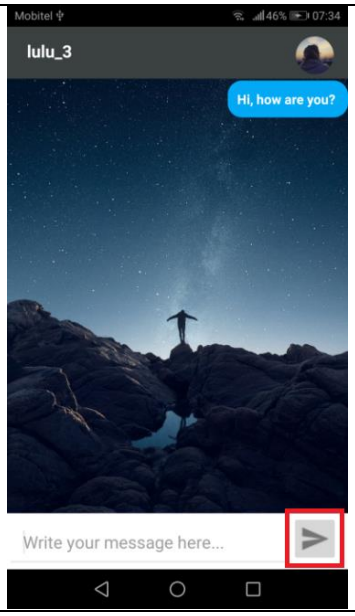
S #	**Prerequisites:**
1	User logged in to account, Different users profile was accessed and “Send Message” Button was pressed

S #	**Test Input**
1	Message text

Test Scenario	Verify, if the user can send messages to other users correctly and if messages are displayed in the Friend Messages Page

Sequence #	**Action**	**Expected Result(s)**	**Actual Result**	**Test Result**
1	Enter a text message in the input field and press the “Send” Icon in the bottom right corner	A text message is sent to the other user successfully	As Expected	Pass

Screenshot for Sequence #
1



CHAPTER 3 TABLE 10: LOGOUT - TC10

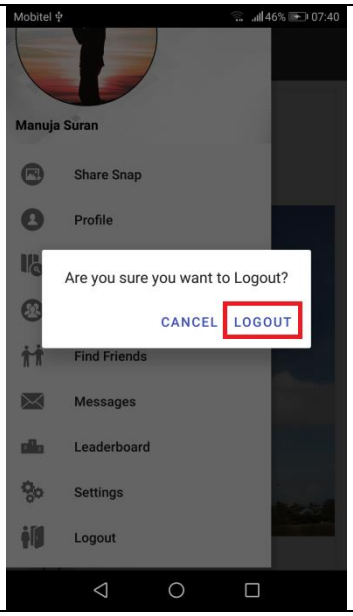
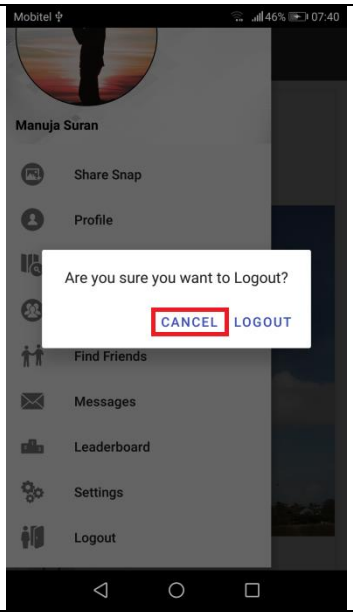
Test Case ID	Logout– TC10
Test Case Description	Test the Logout Functionality

S #	**Prerequisites:**
1	User logged in to account
2	User logged in to account

S #	**Test Input**
1	User presses Logout in the popup
2	User presses Cancel in the popup

Test Scenario	Verify, if is logged out correctly from the system

Sequence #	**Action**	**Expected Result(s)**	**Actual Result**	**Test Result**
1	Press Logout on the navigation bar in the homepage, Select Logout in the popup	User exits account successfully and is sent to the Login page	As Expected	Pass
2	Press Logout on the navigation bar in the homepage, Select Cancel in the popup	User remains logged in and the popup is closed	As Expected	Pass

Screenshots for Sequence #	
1	2
	

3.5 Evaluation

The critical evaluation procedure of the software artefact is accomplished with the help and engagement of my supervisor. Regardless, the proposed system (Unified Explore) is developed and built for daily use and has no specific audience other than its actual users.

Right after the completion of the final system, a final survey was given to the users and the feedback was retrieved. In the below Table 3 a summary of the retrieved user feedback is listed.

CHAPTER 3 TABLE 11: SUMMARY OF USER FEEDBACK RESULTS

Features/ Functions	Excellent	Good	Average	Poor
Registration process with the system		✓		
Profile Setup procedure	✓			
Login	✓			
User Profile Display		✓		
User Settings	✓			
Upload LocationSnaps		✓		
Like feature for LocationSnaps	✓			
Find Routes (Navigation)			✓	
Friend Matching			✓	
Leaderboard		✓		

Logout	✓			
Overall Efficiency		✓		
Overall Security			✓	

Below comments regarding the system are listed that some users have given in form of feedback. Also, anonymity for the feedback was requested and names were generalized by the common term “System User”.

CHAPTER 3 TABLE 12: SYSTEM USER COMMENTS

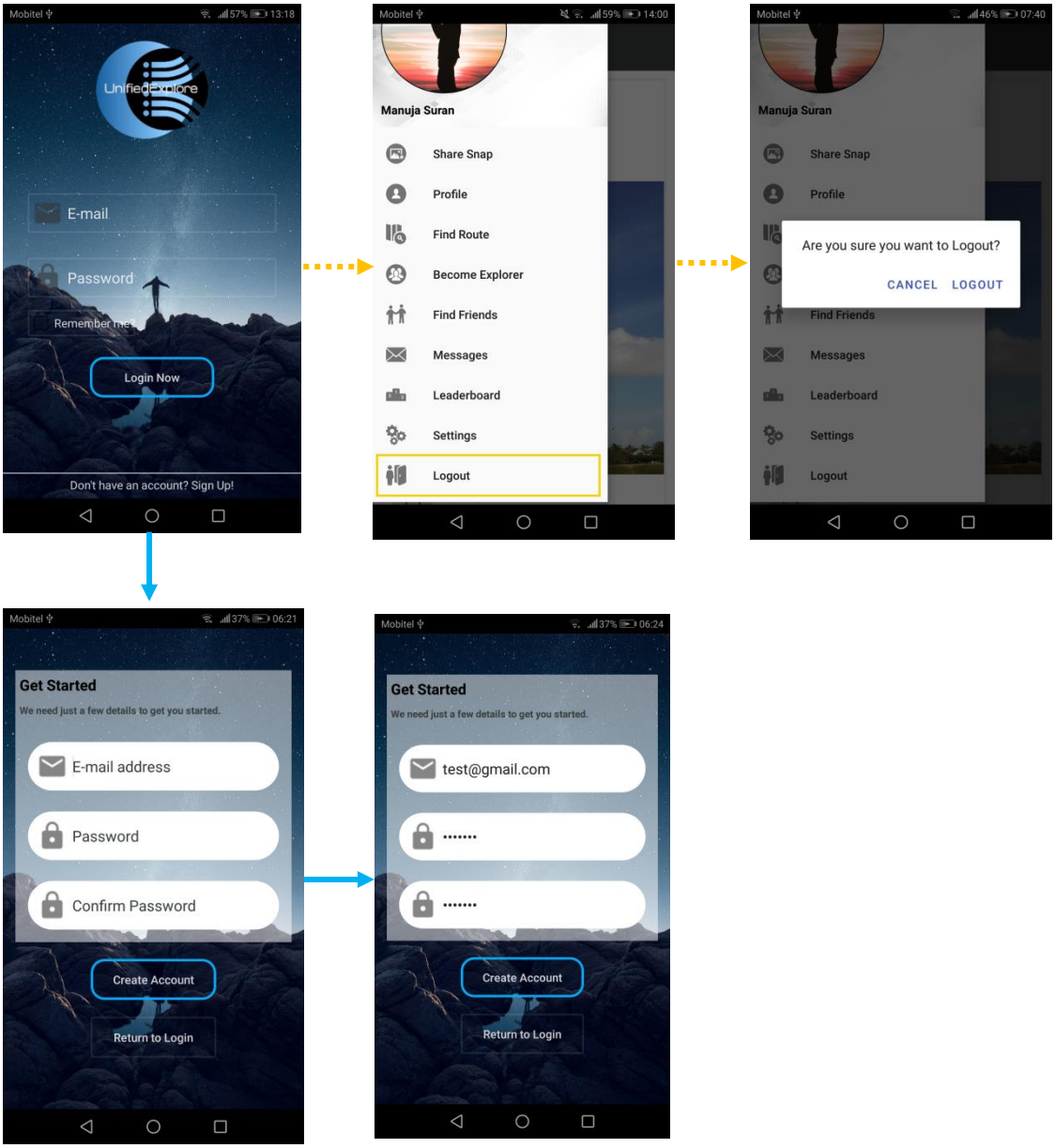
System User	Comments
User 1	The software is very user-friendly and all in all has a good concept that offers various types of services.
User 2	If the system is implemented further this could be an alternative to existing social media applications.
User 3	This whole idea saves time by allowing me to use multiple features within the same application.
User 4	It’s practical and easy to use. Being a social media blogger who travels a lot for various reasons, the LocationSnap and Navigation feature comes very handy for my daily commute.
User 5	The new way of using the navigation by setting destinations is quite simple. I was looking for this feature in other applications as well but without any success.
User 6	The system has a really attractive design. I was from the first usage very satisfied with the application.

4. Research Results and Discussion

The prior chapter discusses the procedure of the project development according to the concept of the SDLC and the testing and its results are analysed using test cases within section 3.4 this particular chapter focuses on the research outcomes that equal to the achievements and discusses reliability and accuracy rates in the form of statistics to determine the success scenarios of functionality components.

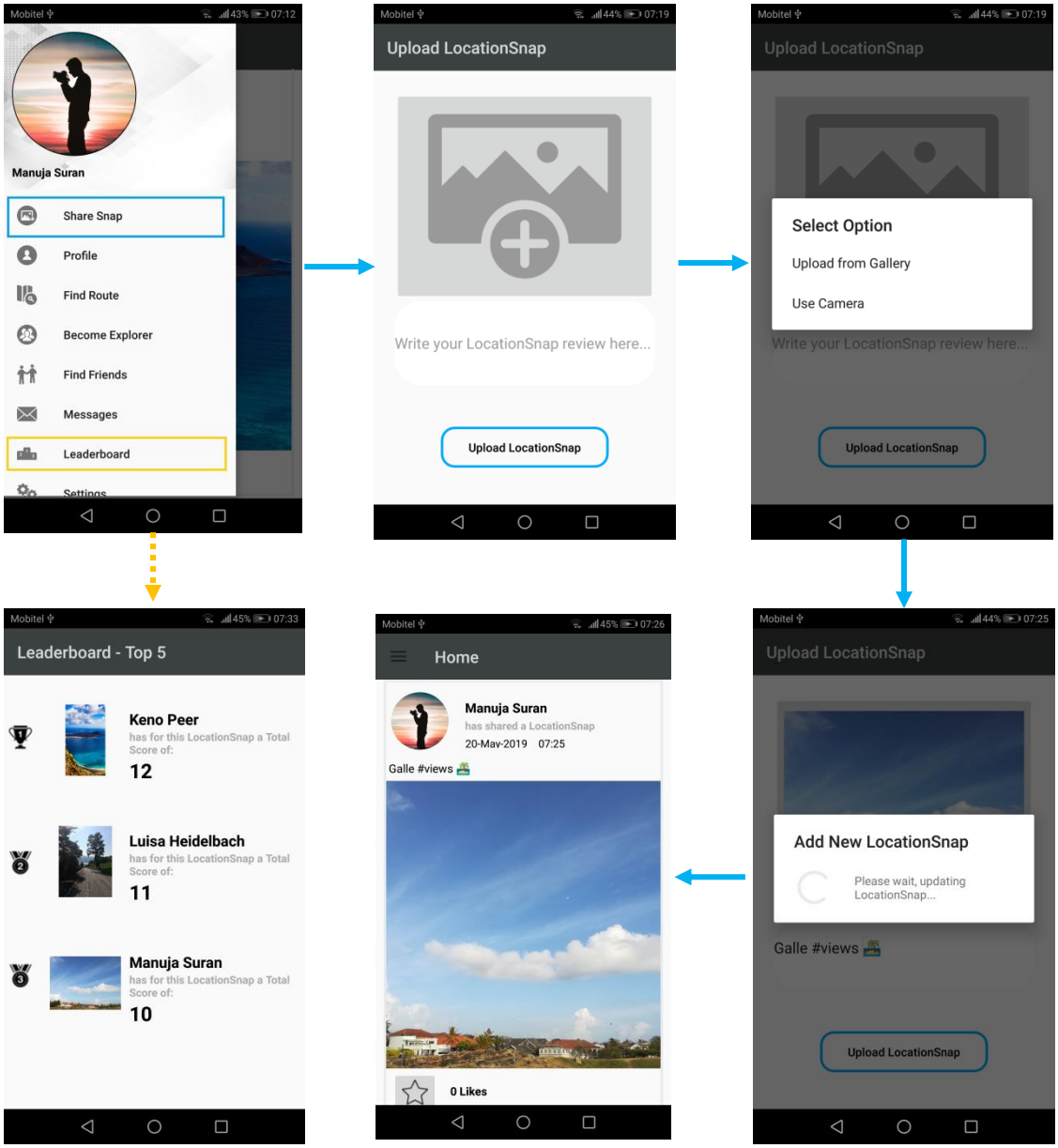
4.1 Achievements

As an important part of the research outcomes that are also stated as the achievements of the research, screens captures describing the functionalities of the system are displayed below. Since the implementation section clarified how the individual features work in detail, the focus for this section will be to give a rough idea about how the users will experience the system.



CHAPTER 4 FIGURE 1: USER ACCOUNT HANDLING

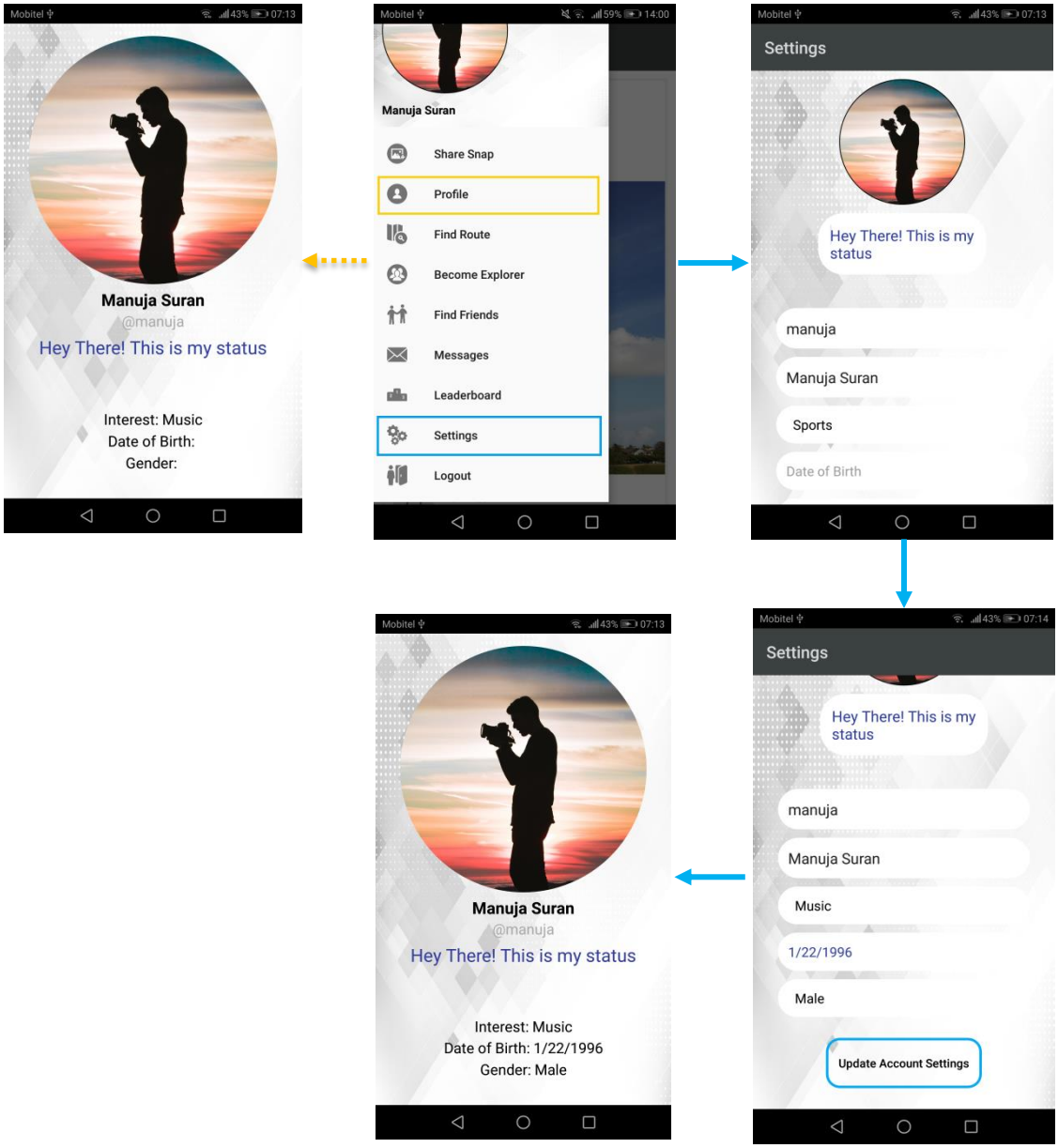
In relevance to Figure 1 users of Unified Explore face the Login page as the start-up screen of the system. From here the users have the option to either sign up and create a new account or log in to the system in case he or she has already an existing account. If the decision is made to create a new account a 2 step process is followed where the first step is by giving the credentials for the needed account and the second by giving additional information to personalize the new account. After this accomplishment, the user can benefit from all the features Unified Explore provides. If users want to log out they can simply do so and since now an account exists, registered users can log in back whenever intended.



CHAPTER 4 FIGURE 2: UPLOAD POST

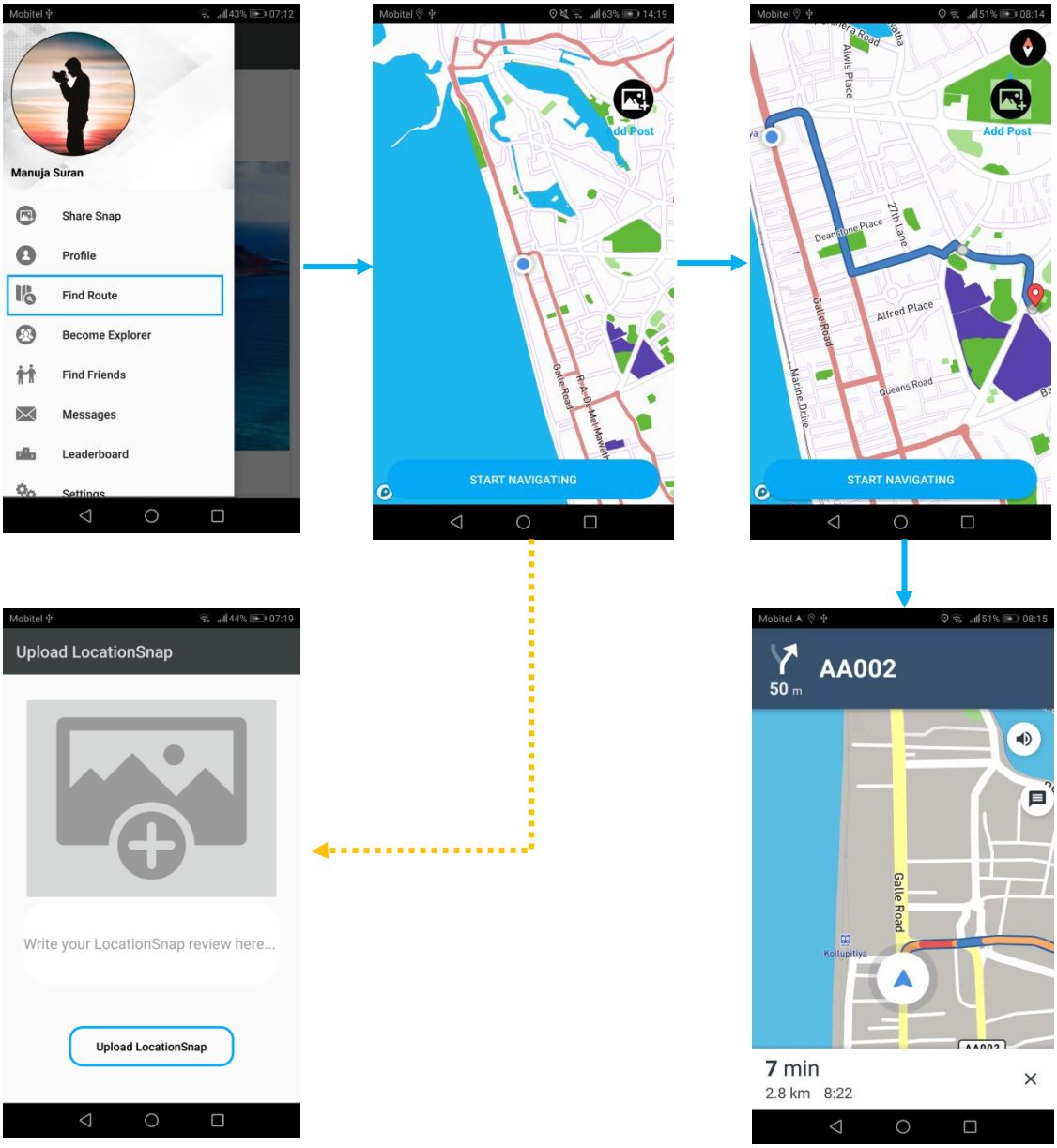
One great feature Unified Explore provides is the upload of new posts, so-called LocationSnaps which are basically users' posts of locations in form or reviews that can be liked with a star by another user. Likes give points and through the Leaderboard functionality, the best LocationSnaps are determined so that other user can see what location and users post are currently trending and stay up to date.

Furthermore, the uploading LocationSnap process allows the user to upload a picture as a new post by either capturing a picture using the device's camera or uploading a picture using the device's gallery.



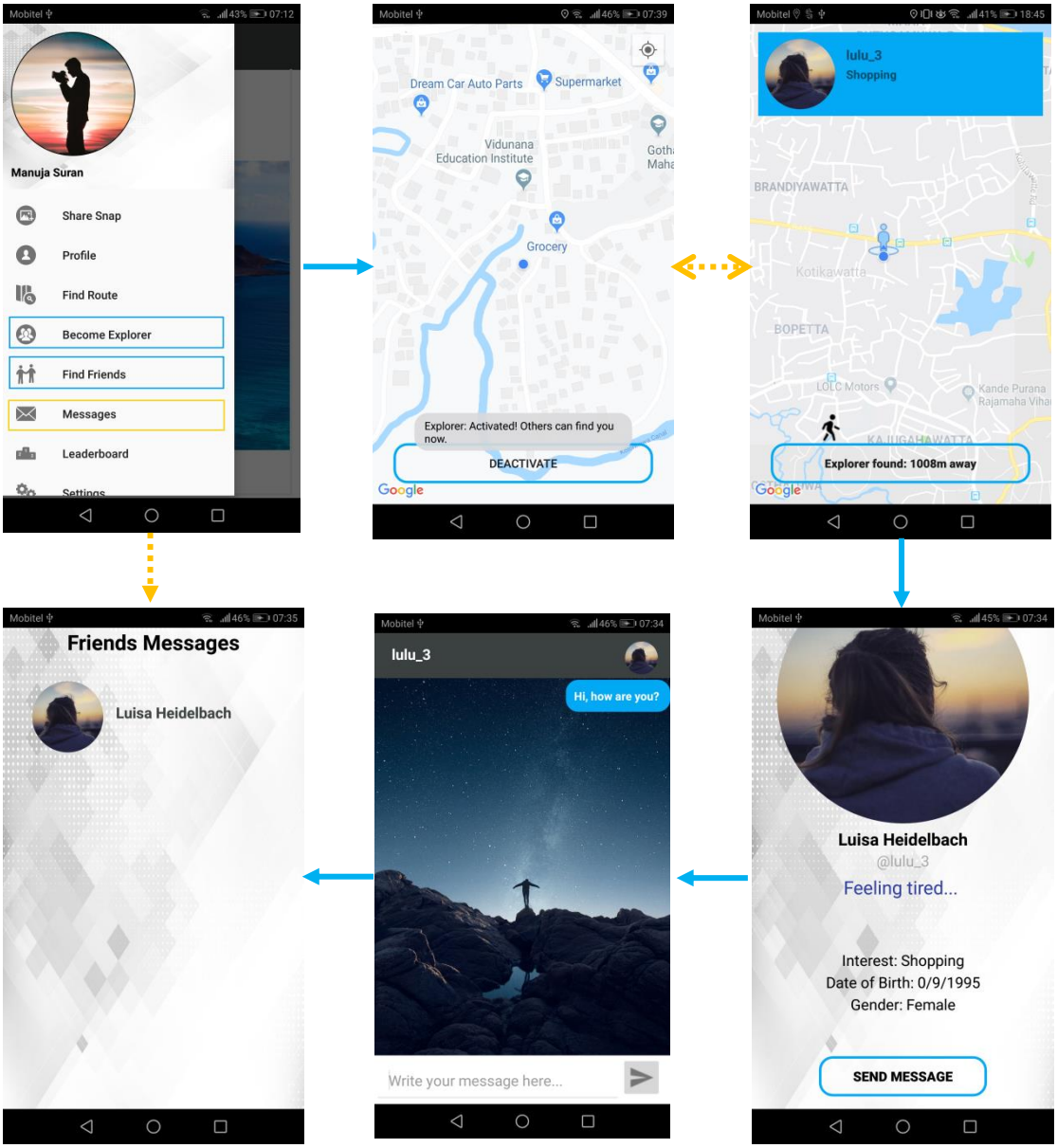
CHAPTER 4 FIGURE 3: USER PROFILE MANAGEMENT

Profiles can be managed in a very simple manner. As the Setting page allows users to change personal information which then is displayed on the Profiles page. Customizable options are the profile picture, status, user id, users name complete name, user's date of birth, field of interest and gender. Moreover, the status allows users to display even something they want to share if friend's visit once profiles page.



CHAPTER 4 FIGURE 4: NAVIGATION FUNCTIONALITY

Figure 4 shows the route navigation UI that is accessed by selecting the “Find Route” option in the Home Menu. This feature allows the user to set a destination for a route by simply clicking on the map. Intermediate stops are displayed as a grey dot, whereas the location of the end destination will display a red marker. Additionally, an “Add Post” icon is available on the right top corner that will lead the user to the Upload LocationSnap feature to share pictures of the users traveling experience. The map is customized for the user's ease and once a destination is set, a “Start Navigating” Button will appear. Once the button is pressed this will lead the user to a navigation view that will display directions until the destination is reached.



CHAPTER 4 FIGURE 5: FRIENDS MATCHING AND MESSAGES FUNCTIONALITY

Figure 5 shows the friend matching feature in which two different users with separate accounts (relationship indicated in orange) can find each other if they are located nearby to each other. Once located each other on the map. The user who wants to find explorers can simply view his or her profile and if wanted send that potential friend a simple text message. This feature can be accessed in 3 different ways:

Once a message is sent to someone else a shortcut for of the user with the text message will appear in the Messages UI so that he or she can access the previously

send text messages later again and keep chatting with that person whenever wanted.

As a summary, all the achievements are listed below that were obtained after time spend for each of the phases of the research and it’s the researches relevant development stages.

- Acquiring knowledge in how to implement systems with the usage of unknown technologies and concepts through its study
- The entire development of a functioning, practical and user-friendly Android application that provides multiple enhanced services to improve the user's communication skills and assist in the users day to day life
- Time and project management concepts
- Ability to identify research requirements and how to perform problems solving strategies in the occurrence of major casualties

4.2 System Function Accuracy and Reliability

Every segment is parted into a number of procedures that were tested using multiple executions to check the output in terms of relevance and consistency. Within parenthesis factors that specify whether each case is considered as accurate or as reliable. The table below shows an overall accuracy rate of 83.57% which is not perfect but certainly difficult to achieve, whereas the reliability rate is highly dependent on a secure connection to the internet to make use of the real-time database. Therefore the overall reliability rate concludes to 80.71%.

CHAPTER 4 TABLE 1: SYSTEM FUNCTION ACCURACY AND RELIABILITY

Case Description	Accuracy Rate	Reliability Rate
User Account Handling	95% (Proper updates performed)	90% (Successful attempts)
Post Functionality	90% (Accurate Signup and Login)	90% (Successfully completed)
User Profile Management	85% (Proper storage of data)	85% (Successful attempts)
Navigation Functionality	70% (Correct data retrieval)	70& (Successful attempts)
Friend Matching Functionality	65% (Correct data retrieval)	60% (Successful attempts)
Messaging Functionality	95% (Proper storage of data)	90% (Successful attempts)
Leaderboard Functionality	85% (Proper updates performed)	80% (Successful attempts)
Overall Rate	83.57%	80.71%

5. Conclusion

This is the final thesis report for the all-in-one communication system called Unified Explore. The document consists altogether of five main chapters in which the whole development procedure for the software artefact is explicitly discussed throughout the report.

In the following, the final chapter will not only show an overall idea about the significance of the system in relevance to the accomplished objectives, but also the limitations and challenges that were faced during the time of the project. Additionally, possible future work and possible recommendation that can improve the system is stated in this chapter.

5.1 Objectives achieved

After a careful study and analysis of the evaluation, the end result concludes that the overall project is successful up to a percentage of 80%.

From the project initiation onwards there were six important objectives specified to aim for the success of the software system. Five of these objectives were achieved by the end of this project. One objective of identifying a profile matching algorithm that matches users with the same shared field of interest could not be completely achieved as expected. This will be explained in the last part of the Limitations section in detail for further clarification.

5.2 Limitations

Starting from the initiation of the project issues occurred that can be expected from any research. Regarding the development of Unified Explore the implementation was complicated up to a point that continuous progress was difficult to maintain due to several reasoning's. These obstacles were mentioned in the previous reflective report of this project as well but explained more thoroughly in this part of the project.

Even though the proposed system is completed up to a percentage of 80% as previously stated the process from the beginning of the project up until its

completion included several deviations and limitations that are mentioned in the following.

The first issue began with the planning phase in which a Gantt chart was created before starting the other phases of the project. The Gantt chart showed with the continuous progress of the project that few unrealistic time frames for particular tasks were stated, even though the milestone were correctly set. Moreover, delays in the progress happened over time that are clarified below and made the deadlines as a result difficult to follow. Furthermore, at the end of the projects duration the deadlines were extended because of casualties within the country. Owing to the national state of security, all universities in the country had been closed and progress was put to hold for that duration of time.

Secondly, using new technologies such as a real-time database Firebase and Android Studio IDE featured trouble. Due to a lack of resources that included having a low specs device that heated up if used, the development phase of the prototype system and the final system were challenging. Within the development stage the device the artefact was being implemented on started to malfunction frequently. A continuous Blue Screen of Death (a fatal system error that leads to an eminent system crash) occurred resulting in countless repairs and further delays. Unfortunately, this device which is equipped well enough to perform the requested tasks regarding the project had to be replaced with a second device which is a low specs rated ultrabook. The ultrabook features the following specs:

- CPU: 2.50GHz Intel Core i5 7200U
- Graphics: Intel HD 620
- RAM: 4GB

During the initial development, the processor of the laptop was overloaded while using the in-build virtual emulators of Android Studio. Various times due to the lack of a second Android phone the emulator had to be used for testing. Enormous delays when executing and running the application on the device problematized the issue of time. Due to the now forced slow working environment, as well as personal career-related work schedules, the time available allowed for development was

reduced immensely. Nevertheless, the implementation was slowly carried out in the best way possible in order to develop the proposed system.

Thirdly, the two concepts of gamification and crowdsourcing the proposed system utilizes were complete new approaches to the field of study. More time had to be given to study these new areas to ensure the success of the system. Hence I was rushed to complete the main functionalities of the prototype in order to continue with the projects further progress such as the testing phase and final product development.

Lastly, to be mentioned is the inability to complete sub-parts of the main functions which aims for the navigation service and friend matching service of the system. Thus, the evaluation of the project reveals that few requirements couldn't be achieved due to a lack of time. These include:

- Creation of an interface to retrieve the user's schedule as input to set the destinations on the route navigation map automatically instead of setting the destination manually.
- Let users match friends based only on the common field of interest. Currently, the system locates all available and potential friends in a nearby area, regardless what the field of interest of a particular user might be.

5.3 Recommendations and Future Work

Based on the user's feedback after the testing and during the evaluation phase, few recommendations were received to improve the system. Below a set of ideas as suggestions for the improvement of the system are listed that few users of this system have given.

CHAPTER 5 TABLE 1: USER RECOMMENDATIONS

No	Recommendations
1	A future improvement I would like to have is the possibility to call other users if their cellular number is provided.
2	My personal recommendation is for the system is to have more possibilities in the Profile section, such as viewing all uploaded pictures and LocationSnaps in a user's profile.
3	I would like to have a feature that lets users customize the app once logged in, for example, by changing the background of the profile or the message page. This would make the whole application even more satisfying to use.

The above recommendations given by users as listed in table 5 are upgrades that will follow up in the future with certainty. The retrieved recommendations are suitable to the software system and once implemented satisfy the user's experience as a result.

The most important future work to be done is the completion of the systems route functionality so that a user schedule as input can be finally taken and even better routes can be generated to help the user on his or her daily travels.

Additional enhancements in the future include the implementation of a method to send a friend request and cancel friend request before being able to message other users or view their personal information. This allows better privacy optionality giving individual users the possibility to manage who can see one's personal content. Moreover, data encryption will be added to ensure improved security and data privacy of text messages that are sent via the system.

References

Alzantot, M. and Youssef, M. (2012). CrowdInside. *Proceedings of the 20th International Conference on Advances in Geographic Information Systems - SIGSPATIAL '12*, pp99-108.

Android Developers. (2019). *Android Developers*. [online] Available at: https://developer.android.com/ [Accessed 24 May 2019].

Barnett, E. and Casper, M. (2001). A definition of "social environment." *American Journal of Public Health*, 91(3), pp.465a-465.

Bista, S., Nepal, S., Colineau, N. and Paris, C. (2012). Using Gamification in an Online Community. pp.611-618.

Brym, R. and Lie, J. (2013). *Sociology*. Australia: Wadsworth Cengage Learning.

Ceikute, V. and Jensen, C. (2013). Routing Service Quality -- Local Driver Behavior Versus Routing Services. *2013 IEEE 14th International Conference on Mobile Data Management*.

Chen, S., Li, M., Ren, K. and Qiao, C. (2015). Crowd Map: Accurate Reconstruction of Indoor Floor Plans from Crowdsourced Sensor-Rich Videos. *2015 IEEE 35th International Conference on Distributed Computing Systems*, pp.1-10.

Dicheva, D., Dichev, C., Agre, G. and Angelova, G. (2015). Gamification in Education: A Systematic Mapping Study. *Journal of Educational Technology & Society*, 18(3), pp.75-88.

Fan, X., Liu, J., Wang, Z., Jiang, Y. and Liu, X. (2017). Crowdsourced Road Navigation: Concept, Design, and Implementation. *IEEE Communications Magazine*, 55(6), pp.126-128.

Firebase. (2019). *Firebase*. [online] Available at: https://firebase.google.com/ [Accessed 24 May 2019].

Hannon, J., Bennett, M. and Smyth, B. (2010). Recommending twitter users to follow using content and collaborative filtering approaches. *Proceedings of the fourth ACM conference on Recommender systems - RecSys '10*.

Hersh, M. and Johnson, M. (2008). *Assistive technology for visually impaired and blind people*. Berlin: Springer.

Huang, S., Zhang, J., Wang, L. and Hua, X. (2016). Social Friend Recommendation Based on Multiple Network Correlation. *IEEE Transactions on Multimedia, 18(2), pp.287-299*.

Lee, J. and Hammer, J. (2011). Gamification in Education: What, How, Why Bother? Academic Exchange Quarterly, 15(2), 146.

Li, N. and Chen, G. (2009). Multi-layered friendship modeling for location-based Mobile Social Networks. *Proceedings of the 6th Annual International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*.

MakeUseOf. (2019). *The 20 Most Popular Android Apps in the Google Play Store*. [online] Available at: https://www.makeuseof.com/tag/most-popular-android-apps/ [Accessed 24 May 2019].

Olmschenk, G., Yang, C., Zhu, Z., Tong, H. and Seiple, W. (2015). Mobile Crowd Assisted Navigation for the Visually Impaired. *2015 IEEE 12th Intl Conf on Ubiquitous Intelligence and Computing and 2015 IEEE 12th Intl Conf on Autonomic and Trusted Computing and 2015 IEEE 15th Intl Conf on Scalable Computing and Communications and Its Associated Workshops (UIC-ATC-ScalCom)*.

Rummel, R. (1991). *The conflict helix*. New Brunswick, U.S.A.: Transaction Publishers, p.Principles and Practices of Interpersonal, Social, and International Conflict and Cooperation.

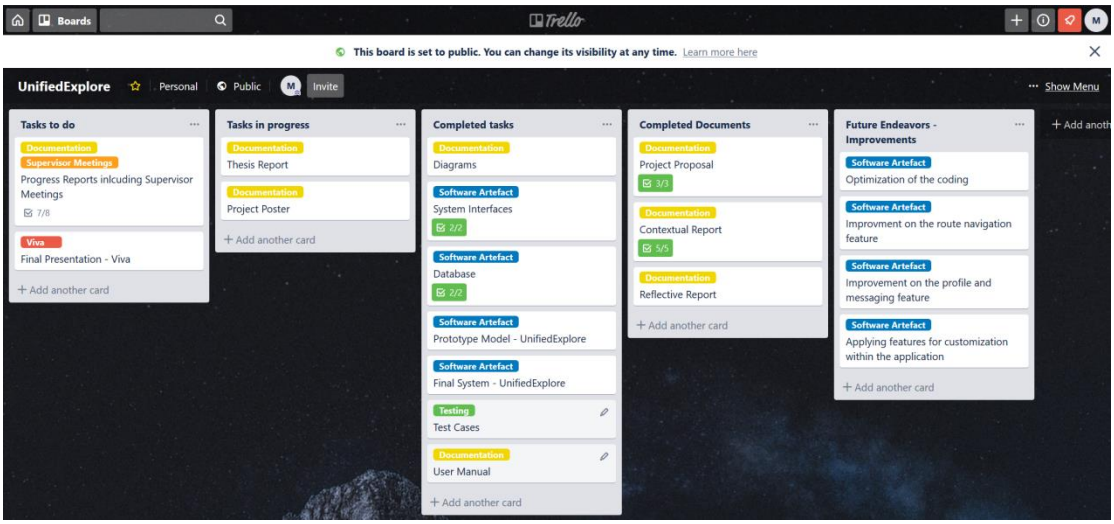
Su, H., Zheng, K., Huang, J., Jeung, H., Chen, L. and Zhou, X. (2014). CrowdPlanner: A crowd-based route recommendation system. *2014 IEEE 30th International Conference on Data Engineering*.

W3schools.com. (2019). *Introduction to Java*. [online] Available at: https://www.w3schools.com/java/java_intro.asp [Accessed 24 May 2019].

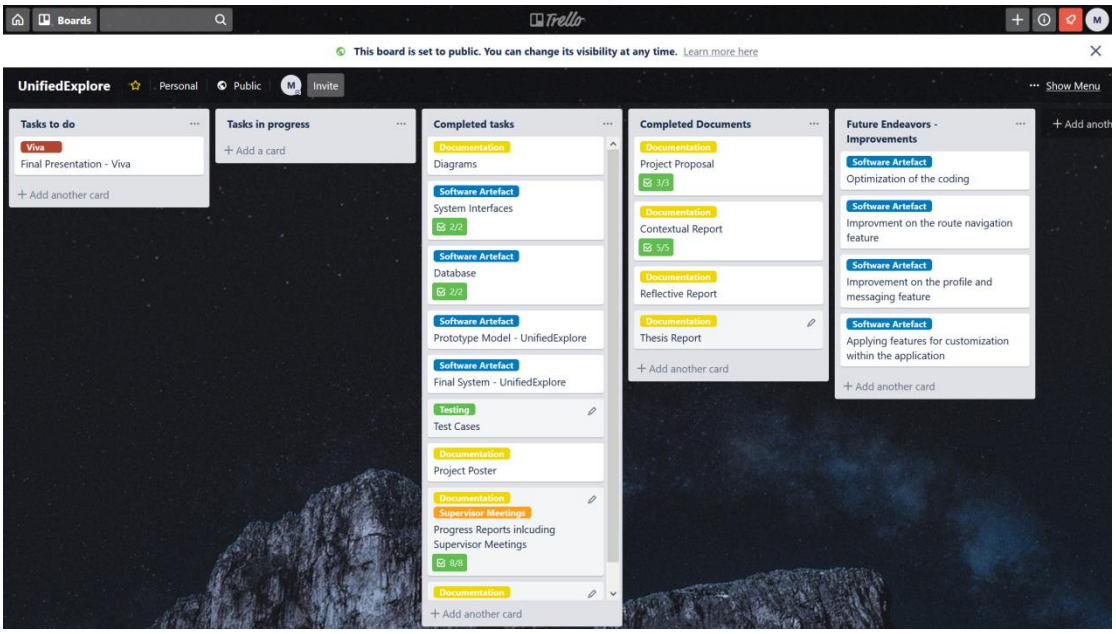
Weber, M. (1978) “The Nature of Social Action,” in Runciman, W. G. (ed.), Matthews, E. (tran.) *Max Weber: Selections in Translation*. Cambridge: Cambridge University Press, pp. 7–32.

Xie, X. (2010). Potential Friend Recommendation in Online Social Network. *2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing*.

Appendix-A (Trello)



APPENDIX-A FIGURE 1: TRELLO - BEFORE FINAL CHANGES

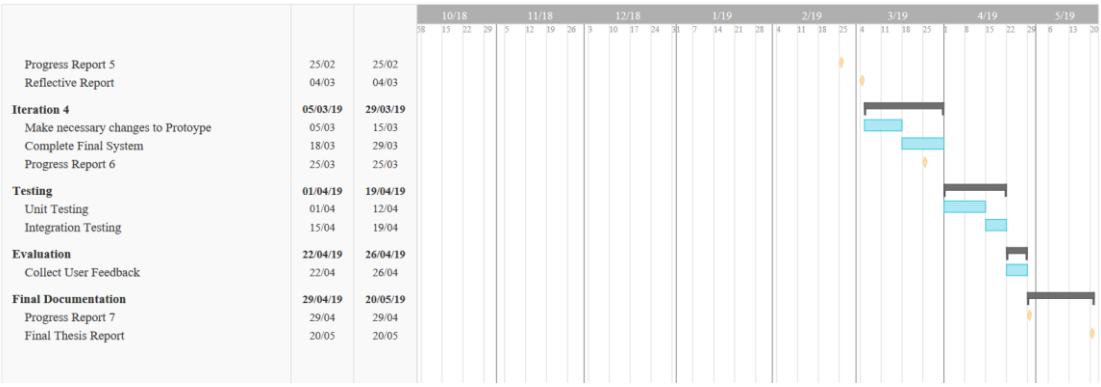


APPENDIX-A FIGURE 2: TRELLO - AT POINT OF PROJECT FINALIZATION

Appendix-B (Gantt chart)

Unified Explore	start	end
Planning	**07/10/18**	**05/11/18**
Determine suitable research topic	07/10	14/10
Feasibility study	15/10	25/10
Progress Report 1	29/10	29/10
Project Proposal & Ethics Form	05/11	05/11
Requirement Gathering	**06/11/18**	**20/01/19**
Finalize Software Requirements	06/11	08/11
Primary Data	**09/11/18**	**16/11/18**
Questionnaire	09/11	16/11
Secondary Data	**19/11/18**	**20/01/19**
Literature Review	19/11	31/12
Research Papers	01/01	10/01
Reliable Websites	11/01	20/01
Requirement Analysis	**09/11/18**	**20/11/18**
Define & establish Software Requirements	09/11	20/11
Design	**21/11/18**	**03/12/18**
Prepare Gantt Chart	21/11	23/11
Progress Report 2	26/11	26/11
Prepare WBS	27/11	03/12
Iteration 1	**04/12/18**	**31/12/18**
Design Use Case Diagram	04/12	14/12
Design ER Diagram	17/12	28/12
Progress Report 3	31/12	31/12
Iteration 2	**21/01/19**	**28/01/19**
Contextual Report	21/01	21/01
Progress Report 4	28/01	28/01
Iteration 3	**29/01/19**	**04/03/19**
Implement Database	29/01	07/02
Implement Prototype	08/02	19/02
Review Prototype	20/02	22/02

APPENDIX-B FIGURE 1: GANTT CHART - 1



Progress Report 5	25/02	25/02
Reflective Report	04/03	04/03
Iteration 4	**05/03/19**	**29/03/19**
Make necessary changes to Protoype	05/03	15/03
Complete Final System	18/03	29/03
Progress Report 6	25/03	25/03
Testing	**01/04/19**	**19/04/19**
Unit Testing	01/04	12/04
Integration Testing	15/04	19/04
Evaluation	**22/04/19**	**26/04/19**
Collect User Feedback	22/04	26/04
Final Documentation	**29/04/19**	**20/05/19**
Progress Report 7	29/04	29/04
Final Thesis Report	20/05	20/05

APPENDIX-B FIGURE 2: GANTT CHART - 2

Appendix-C (Project Poster)

APPENDIX-C FIGURE 1: PROJECT POSTER