Department of Computer Science

Project: NavUP

Client: Department of Computer Science,
University of Pretoria

Software Requirement Specification

Team Jelly

Author(s):	**Student number(s):**
Mpho Baloyi	14133670
Seonin David	15063021
Cian Steenkamp	15095682
Victor Twigge	10376802
Wanrick Willemse	29560617
Idrian van der Westhuizen	15078729

Contents

1 Introduction

This section provides an outline of this Software Requirements Specification(SRS). The purpose for this document and the scope it covers is described and a definition list is provided for abbreviations.

1.1 Purpose

The purpose of this Software Requirements Specification(SRS) is to describe the requirements and specifications of the NavUP system,a mobile navigation application for the University of Pretoria, Hatfield campus. It contains details about the functional features of the NavUP system,along with the interface details,design constraints and related considerations such as software quality attributes.
Further specific requirements are detailed within the document and makes mention of functional requirements and other constraints to give a finer detail of what is expected in the NavUP system. This SRS documents is intended for the clientèle that will oversee the development of the NavUP system.

1.2 Scope

The 'NavUP' system will be used to navigate around the University of Pretoria Hatfield campus. The software will include a map of the campus that distinguishes between food courts, lecture halls, administrative buildings and other locations of interests.
The system will allow a user to identify his current location on campus and search for a destination on campus. The system will then determine the ideal route to the destination based on restrictions specified by the user such as avoiding pedestrian traffic and provide directions. Heat maps will then be used to reveal congested areas and show where large number of students are moving in close proximity. The system will also provide users with information about activities and events that are taking place on campus as well information about points of interests. The system will also include game-like functionality that will award badges to users who have achieved certain distance milestones, and to those who travel to a new area for the first time.
The software will run on any Android or iOS smart phone or tablet. The system will mainly use WiFi connectivity to determine users' locations.

1.3 Definitions, Acronyms and Abbreviations

SRS: Software Specifications Requirement
UP: University of Pretoria
GPS: Global Positioning System
Admin: Administration

1.4 References

Kung, D. (2014). Object-oriented software engineering. 1st ed. McGraw-Hill, p.98.

1.5 Overview

The SRS firstly gives an overall description of the NavUP system and its various interfaces. Each interface is divided into a separate subsection where further detail is given about what it must do and how it interacts with other interfaces. Afterwards the SRS makes mention of the memory constraints and operations. The requirements to the site adaptation are specified further in the SRS documentation.
The SRS describes the average expected user for NavUP the system and what constraints the developers need to take into account when designing the overall system in more detail. A list of assumptions and dependencies of the user and the overall system is given that were used to design the basic functionality of the NavUP system detailed in the SRS document.
Specific requirements are given for the external interface and functional requirements. The functional requirements contains smaller logical modules and how they might work. Performance requirements that describes how the NavUP system should perform and what is expected of its performance along with design constraints are given for the NavUP system. The SRS then describes the various quality attributes the system should have in order to function reliably.

2 Overall Description

2.1 Product Perspective

2.1.1 System interfaces

The system is going to be designed in a modular fashion, where the separate functionalities are broken up to allow for multiple programmers and design-

ers to work on the NavUP at once. The modularity also allows for better maintenance and upgrading of future software and/or hardware by allowing the programmers to only change a smaller group of modules.
The System will need to be coded in such a way that multiple types of mobile devices would be able to use the NavUP system. It must also be able to communicate with an external database/server such as ClickUP where user information can be tracked and saved for further use in other applications and functionalities.

2.1.2 User interfaces

The user interface should be designed for a mobile device, in other words the screen should not be cluttered with icons and make use of touch screen technologies and its gestures. The map of the Hatfield campus along with various points of interest should be clearly visible on the screen. The user interface must be unambiguous since not only students and staff will make use of the NavUP system, but visitors as well. Since visitors will make use of the system, a way to locate and find various building by name would be beneficial, not only for visitors, but perhaps first year students as well.

2.1.3 Hardware interfaces

The NavUP system should be able to make use of the Wi-Fi routers scattered throughout the Hatfield campus. The application itself should be able to run on a mobile device and therefore make use of phone data alongside the Wi-fi routers and make use of the built-in GPS system on most mobile devices. The NavUP application should be able to support input from touch screen devices from the user's mobile device to communicate and request various functionalities of the system.
The system should also make use of an external database to track a user's progress for various achievement based activities. The system would also be able to use the database to direct specific help/information to the user.

2.1.4 Software interfaces

The various classes and modules programmed on the software of the system should be capable of receiving data from the hardware and communications functions of the system. The software should be able to calculate and update values on the internal system as well as the external database. Various classes and modules should be able to send and receive values from one another and these updated/received values should be able to communicate with the mobiles devices interface in order to update the map. The software should also

be capable of updating the data of the external database.

2.1.5 Communication Interfaces

The mobile device used by the user should be able to communicate with the Wi-Fi routers throughout the Hatfield campus in order to update values such as coordinates on the system. The user's mobile device should also be able to send and receive data from an external database, this data can also be used to block/allow access to certain features for instance a student must be able to participate in game like activities, but a visitor does not have to. Various mobile devices should be able to communicate with their navigation systems and other mobile devices (directly or indirectly) in order to calculate and create heat maps of high user traffic in an area on the map.

2.1.6 Memory

Because this application will be mainly mobile based, it should use as little as possible primary memory. It must in no way overload the mobile device's functional capacity. The installation size must be small, to not clutter up user space on the user's device. Application download should also preferably take place over the UP WiFi network to alleviate user data costs.
The entire system will also make use of an external database in order to save and track user progress and various other events, this way the user need only retrieve the data from the external database rather than waste the memory space of the mobile device.

2.1.7 Operations

2.1.8 Site Adaptation Requirements

2.2 Product Functions

2.3 User Characteristics

The following is a list of the four types of users for the NavUP system and the different ways in which they interact with the system.

- Students
  - This group of users will use the front-end of the system, which consists of a mobile application which will run on various mobile devices.

  - Students will be provided with ability to view their current location, search and locate venues and navigate to destinations without the need to log in.
  - To enjoy the benefits of saving locations, receive personalized information such as activities and events based on points of interest, students need to be logged in.

- University Of Pretoria Staff
  - This group of users will use both the front-end and back-end of the system,they will use the back-end to add view and change user details.
  - The front-end may be used the same way it is used by students.

- Guests
  - Guest users will be able to view their location, search venues, view events and navigate to destinations.

- Administrators of the NavUP system
  - This group of users will use the back-end part of the system.
  - They will be able to add, remove and manage the following: users, activities and event notifications.
  - They are responsible for the running of the system.

2.4 Constraints

The mobile application is constrained by the system interface to the WiFi hardware within the mobile phone. Since there are multiple WiFi manufacturers, the interface will most likely not be the same for every one of them. Also, there may be a difference in the performance and specifications of the hardware.
The Internet connection is also a constraint for the application. Since the application fetches data from the database over the Internet, it is crucial that there is an Internet connection for the application to function. Both the back-end and the mobile application will be constrained by the capacity of the database.

2.5 Assumption and Dependencies

An assumption about the product is that it will always be used on mobile phones that have built-in WiFi hardware and have enough performance to run the application in a working and consistent manner. Another assumption is that the WiFi components in all mobile phones work in the same way. If the phones have different interfaces to the WiFi, the application need to be specifically adjusted to each interface and that would mean the integration with the WiFi would have different requirements than what is stated in this specification. For the application to launch and run it will depend on a compatible operating system and version of the mobile phone.

3 Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interface

The mobile application will interface with the supported input and output features of the host's operating system. Inputs include text that the user will enter for login or searching a venue. Outputs include the type of fonts to display text or graphics to show images or draw the map.

3.1.2 Hardware Interfaces

Since neither the mobile application nor the web portal have any designated hardware, it does not have any direct hardware interfaces. The WiFi software in the mobile phone manages the built-in WiFi and the hardware connection to the database server is managed by the underlying operating system on the mobile phone and the web server.

3.1.3 Software Interfaces

The mobile application communicates with the WiFi software in order to get signal strength information from multiple WiFi access points to determine (using triangulation) where the user is located. The communication software between the database and mobile application consists of operation concerning creating, reading, removing and modifying the data.

3.1.4 Communication Interfaces

The communication between the different parts of the system is important since they depend on each other. However, in what way the communication is achieved is not important for the system and is therefore handled by the underlying operating systems for both the mobile application and the back-end of the system.

3.2 Functional Requirements

3.2.1 1-Admin system

1-1 The admin system will allow a designated administrator to add, remove and manage user accounts.
Preconditions:

- Must have admin privileges
- Must be registered and logged in

Postconditions:

- Changes made to user accounts

1-2 The admin system will allow a designated administrator to add, remove and manage system, activity and event notifications.
Preconditions:

- Must have admin privileges
- Must be registered and logged in

Postconditions:

- Changes made to event notifications

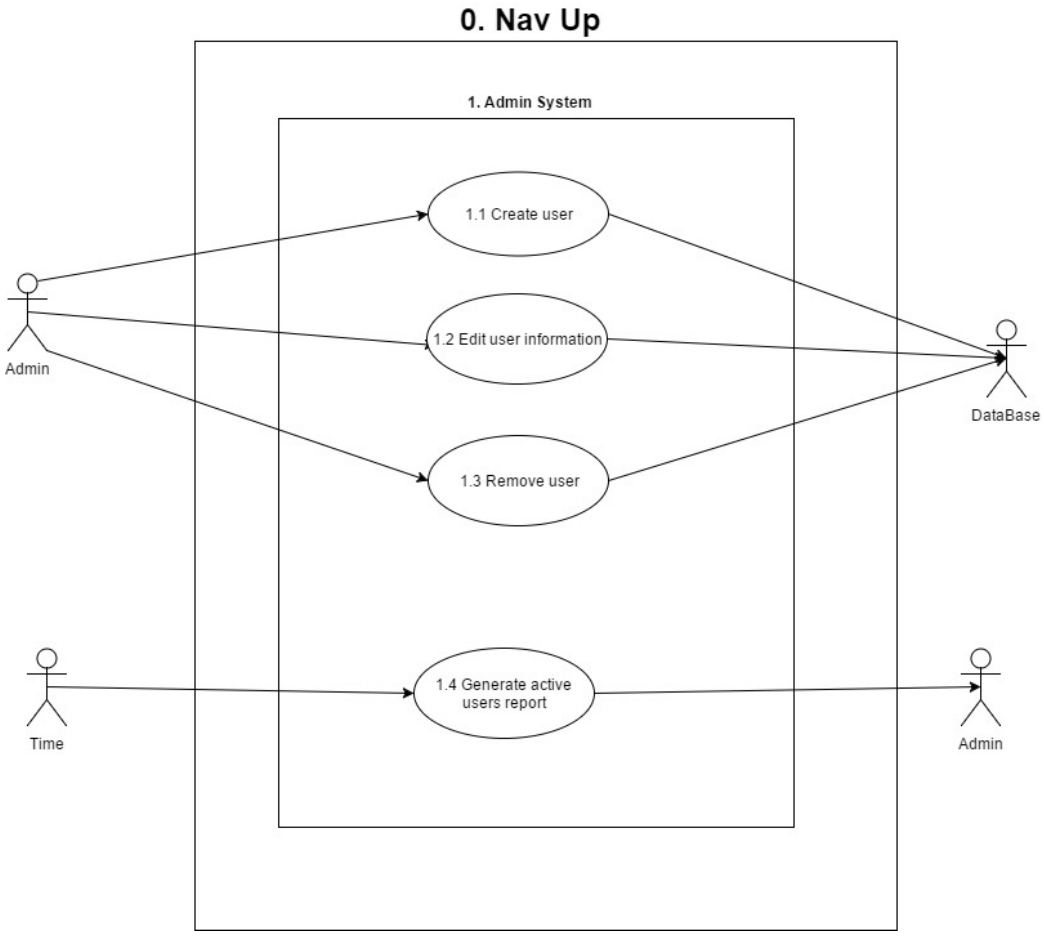

Figure 1: Admin System use case diagram

UC1: Admin System

Actor: Admin	System: Admin System
	0. The system displays the input boxes for the admin to a) Add a new user b) Remove a user c) Edit user details
1. The admin a) Adds a new user b) Removes a user c) Edits user details if it was originally incorrect.	2. The system a) Will add a new user to the database b) Remove a user that is no longer part of the university c) Edit user details and update the database
3. The admin will update the database if need and then log out of the system.	4. The system will generate a report every week that displays the number of active users for the admin to check.

Figure 2: Admin System Actor Diagram

3.2.2 2-Account management

2-1 The account management system will allow a registered user (staff member and student) to login to the system.
Preconditions:

- Must be registered

Postconditions:

- User logged in

2-2 The account management system will allow a guest user to login to the system using a guest account.
Preconditions:

- Must be a visitor

Postconditions:

- User logged in

2-3 The account management system will allow a registered user (staff member and student) to view and change user details.
Preconditions:

- Must be registered and logged in

Postconditions:

- Changes made to user details

2-4 The account management system will allow an administrator to make a change to the account system, ie what details about users are stored.
Preconditions:

- Must have admin privileges
- Must be registered

Postconditions:

- Changes made to account management system

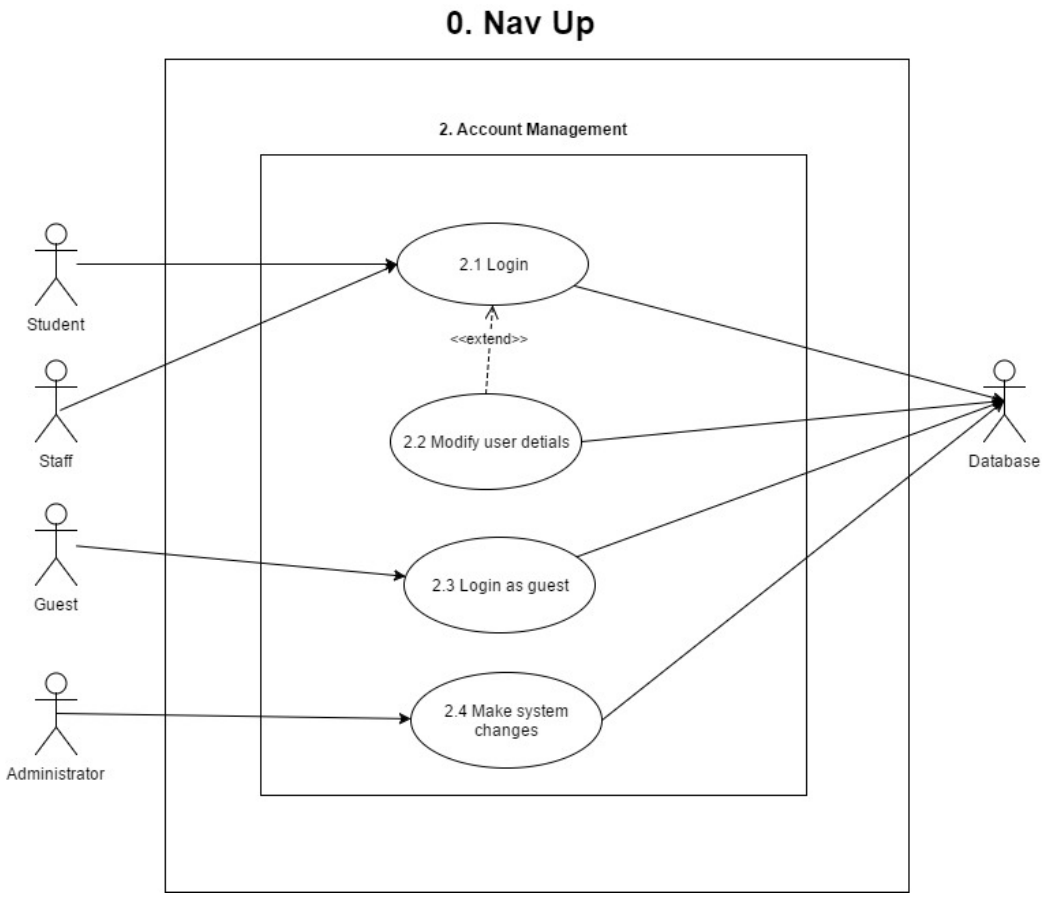

Figure 3: Account Management use case diagram

UC2: Account Management

Actor: Student, Staff	System: Account Management System
	0. The system displays the input boxes for the user to enter his/her details.
1. The user a) enters his/her details and clicks the login button b) chooses not to login and closes the app	2. The system displays the following depending on the details entered a) Logs the user in b) Displays incorrect login
3. The user a) is able to modify his/her details b) the user is not on the system and has to register as a guest	

Actor: Admin	System: Account Management System
	0. The system displays the input boxes for the admin to login
1. Admin enters details	2. System verifies that the admins details are correct and logs the admin in
3. The admin can a) UC1.1: Add user b) UC1.2: Remove users c) UC4.4: Edit activity information	

Figure 4: Account Management Actor Diagram

3.2.3 3-Core-Navigation system

3-1 The core navigation system will allow a user to view his/her current location.
Preconditions:

- Must be registered and logged in
- Must be on campus and connected to WiFi network

Postconditions:

- User sees his/her location

3-2 The core navigation system will allow a user to save his/her current location details for later retrieval.
Preconditions:

- Must be registered and logged in
- Must be on campus and connected to WiFi network

- Must view current location

Postconditions:

- User saves his/her location

3-3 The core navigation system will allow a user to enter a destination and get directions. This will utilize the heat map to find and indicate an optimal route based on pedestrian congestion.
Preconditions:

- Must be registered and logged in
- Must be on campus and connected to WiFi network

Postconditions:

- User sees optimal route and heat map

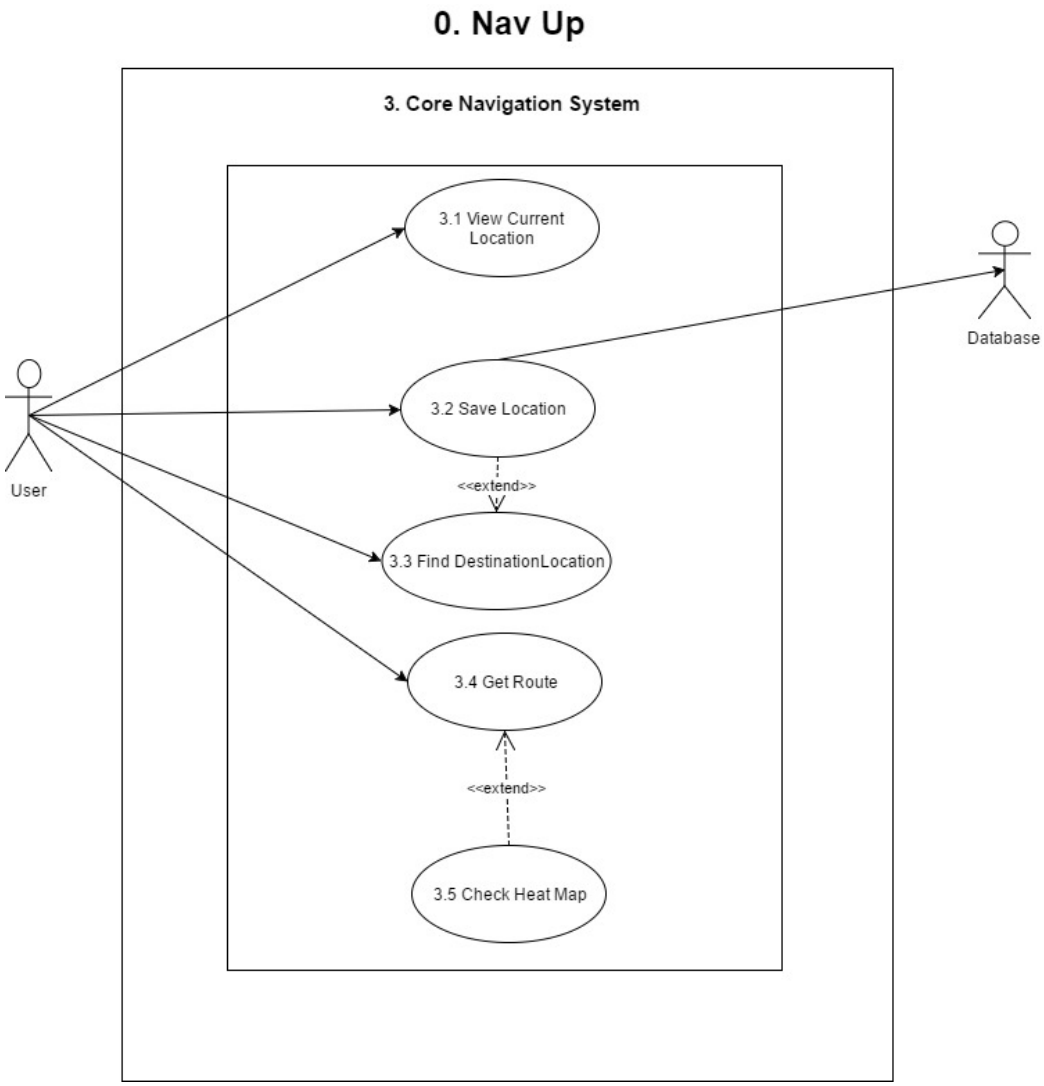

Figure 5: Core Navigation use case diagram

UC3: Core Navigation System

Actor: User	System: Core Navigation System
	0. System displays the main menu
1. User can a) view his/her current location b) enter the location that he/she is going to	2. The system will a) check the heat map to get the route with the least populated people b) generate the fastest route to get to the destination
3. The user can choose to save this location for future use	4. The system will store the users preferences on the database

Figure 6: Core Navigation Actor Diagram

3.2.4 4-Activity management

4-1 The activity management system will allow a user to manage current activities.
Preconditions:

- Must be registered and logged in
- Must be on campus and connected to WiFi network

Postconditions:

- Changes made to activity list

4-2 The activity management system will allow a user to add an activity to an activity list.
Preconditions:

- Must be registered and logged in
- Must be on campus and connected to WiFi network

Postconditions:

- Activity added to list

4-3 The activity management system will allow a user to remove activities from an activity list.
Preconditions:

- Must be registered and logged in
- Must be on campus and connected to WiFi network

Postconditions:

- Activity removed from list

4-4 The activity management system will allow a user to view the heat map of current pedestrian traffic on campus.
Preconditions:

- Must be registered and logged in
- Must be on campus and connected to WiFi network

Postconditions:

- User sees campus pedestrian traffic on heat map

4-5 The activity management system will update a user on any relevant activity information.
Preconditions:

- Must be registered and logged in
- Must be on campus and connected to WiFi network

Postconditions:

- User receives notification

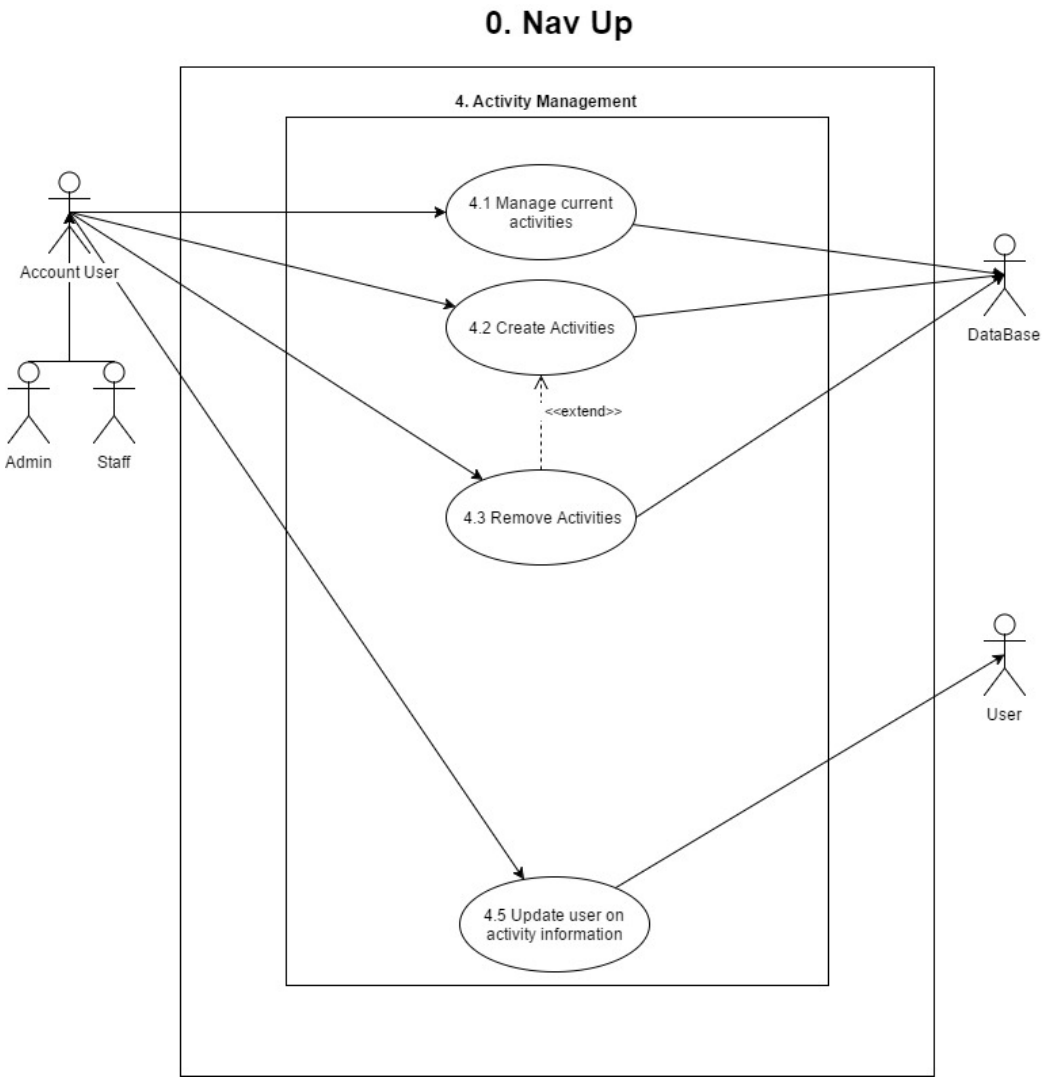

Figure 7: Activity management use case diagram

UC4: Activity Management

Actor: Staff, Admin	System: Activity Management System
	0. The system displays the input boxes for the user to enter new activities.
1. The staff or admin a) Create a new activity b) Edit a current activity	2. The system a) Will add the new activity to the database and update the users accordingly b) Change the activity information and update the users accordingly
3. The user will remove a created activity if it had passed or is finishing	4. The system will remove the activity from the database and update the user depending on the heat map on whether or not that area has less traffic

Figure 8: Activity Management System Actor Diagram

3.2.5 5-Notification

5-1 The notification system will allow an administrator to create a notification.
Preconditions:

- Must be registered and logged in
- Must have admin privileges

Postconditions:

- Notification created

5-2 The notification system will allow an administrator to remove a notification.
Preconditions:

- Must be registered and logged in
- Must have admin privileges

Postconditions:

- Notification removed

5-3 The notification system will allow an administrator to push notification to relevant users.
Preconditions:

- Must be registered and logged in
- Must have admin privileges

Postconditions:

- Relevant users received push notification

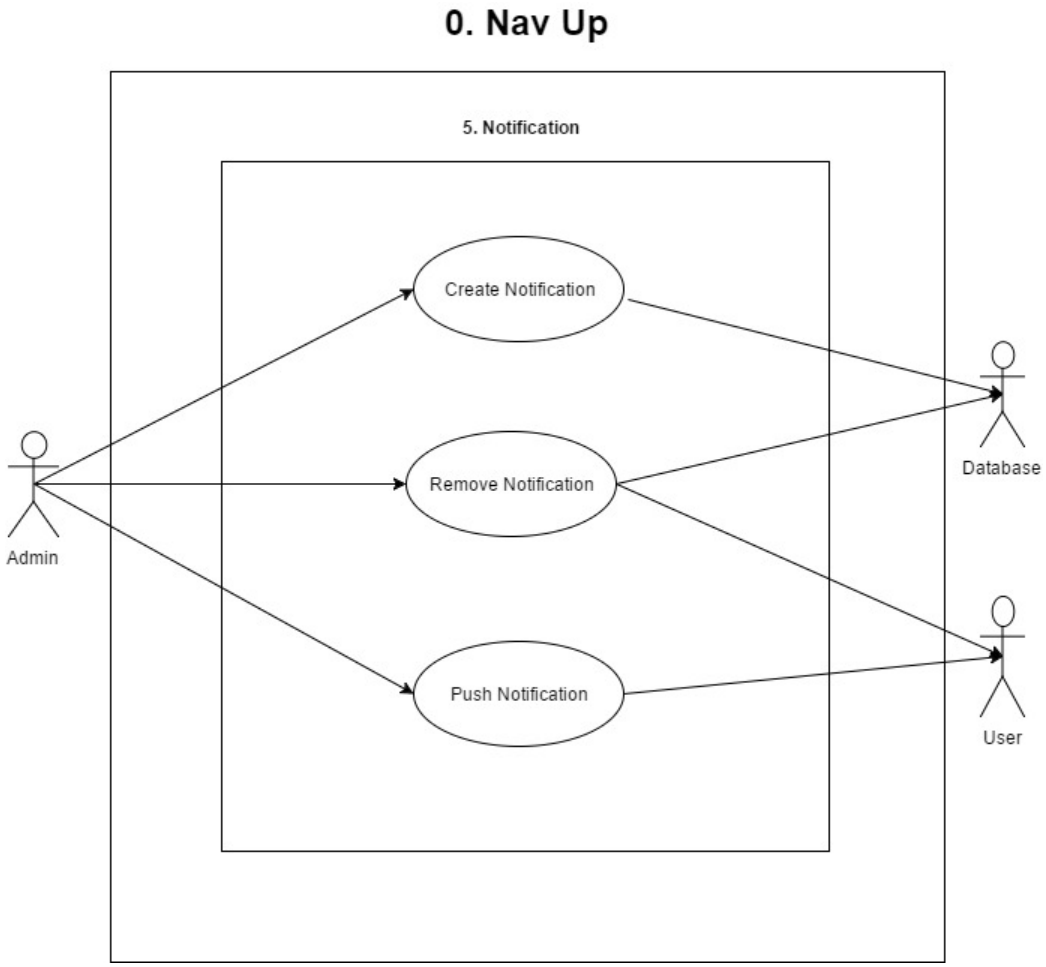

Figure 9: Notification use case diagram

UC5: Notification System

Actor: Admin	System: Notification System
	0. The system displays the input box for the admin to create a new notification or an option to push a notification.
1. The admin a) Create a new notification b) Push a notification c) Remove a notification	2. The system a) Will add the newly created notification to the database b) Push the notification to the user c) Remove the notification and update the user and database system accordingly

Figure 10: Notification Actor Diagram

3.2.6 Traceability Matrix

Requirement	Priority	1.1	1.2	1.3	1.4	2.1	2.2	2.3	2.4	3.1	3.2	3.3	3.4	3.5	4.1	4.2	4.3	4.4	5.1	5.2	5.3
Core Navigation System	1									x	x	x	x	x							
Activity Management	2														x	x	x	x			
Push Notification	3																	x	x	x	x
Admin System	4	x	x	x	x	x			x						x	x	x	x	x	x	x
Account Management	5					x	x	x	x												
UC Priority		8	9	9	9	10	11	10	11	1	2	2	3	3	5	4	5	5	6	7	7

Figure 11: Traceability Matrix

Use cases

- 1.1 Create User
- 1.2 Edit User Information
- 1.3 Remove User
- 1.4 Generate Active Users Report
- 2.1 Login
- 2.2 Modify user details

- 2.3 Login as guest
- 2.4 Make system changes
- 3.1 View Current Location
- 3.2 Save location
- 3.3 Find Destination
- 3.4 Get Route
- 3.5 Check heat map (Determine Pedestrian Traffic)
- 4.1 Manage Current Activities
- 4.2 Create Activities
- 4.3 Remove Activities
- 4.4 Update User Information
- 5.1 Create Notification
- 5.2 Remove Notification
- 5.3 Push Notification

3.3 Performance Requirements

3.4 Design Constraints

3.5 Software System Attributes

- Reliability:
    - Any information that is stored on the database must remain correct when being transferred to the user interface
    - The services offered by the system should be available to users except for when the system is undergoing maintenance
    - The system should reply to user requests in the shortest time interval possible
    - The system must be fault tolerant,it needs to maintain a certain level of performance and offer other services that are not affected by this fault to the users

    - In the event of a fault the system must be able to recover within the shortest time period possible and recover any data that may have been lost.
    - The system should be able to respond appropriately if it receives bad input data from the user.

- Scalability:

    - The system must be able to cater for increases in the work load, for example large number of users or activities at any given time,without impacting on the performance of the system.
    - If the system does not cater for increases in work load it should at least provide the ability to be readily enlarged/

- Maintainability:

    - The system must be designed in a modular fashion that provides high cohesion and loose coupling, the will allow parts of the system to be easily maintained without affecting the rest of the system.
    - Maintenance should be able to be carried out by different maintenance teams, therefore the system must be easy to learn and understand.

- Integrability:

    - Since we are following a modular design, components of the system that are separately developed should work correctly together.
    - Follow coding standards specified by the client to allow for easy integration and employ continuous integration in our design process

- Usability:

    - The system must be easy to learn.
    - System must cater for user mistakes,by providing the user with the undo or roll back option.
    - The user interface must be easy to use and intuitive.
    - System should have options in a logical manner.
    - Use widgets and icons that the target users may be familiar with.
    - The user manual should have a detailed description of the system.

    - A help option must be provided to the users.

- Interoperability:

    - The system must be able to communicate with the University of Pretoria WiFi system, because the wifi access points will be used for the navigation.

3.6 Other Requirements

- Low Resource Consumption

As mentioned above the NavUP will be running on mobile devices due to the small amount of resources some of these devices may have, unnecessary use of resources on the front-end of the system should be avoided.

4 Appendix