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. 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 the software requirements specification (SRS) document is to highlight the requirements and functionality of the NavUP system. The SRS describes the overall design of the system and its interfaces from the various interfaces and how they are linked with one another.
The documents contains details of what the product must be able to do and describes the various users that will make use of the NavUP system so that appropriate use interfaces can be designed. The details about the products assumptions and its dependencies are also contained within the document along with the constraints of the system, this is to allow for a better understanding of what must be done and what can be done within the system, along with what must be tried to avoid.
Further specific requirements are detailed within the document and makes mention of functional requirements and other design constraints to give a finer detail of what is expected in the NavUP system. The SRS documents is intended for the clientèle that will oversee the creation 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. The user will then be able to pick a destination on campus and the system will determine an ideal route and provide directions. Heatmaps will reveal congested areas and show where large numbers of students are moving in close proximity. This will allow a user to use a route that has less pedestrian traffic. Furthermore, a user will be allowed to save locations and search for destinations.
The system will 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 smartphone 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

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 designers 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

2.4 Constraints

2.5 Assumption and Dependencies

3 Specific Requirements

3.1 External Interface Requirements

3.2 Functional Requirements

3.2.1 1-Admin system

3.2.2 2-Account management

3.2.3 3-Core-Navigation system

3.2.4 4-Activity management

3.2.5 5-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.
    - System must be designed in a way that allows the users to learn as they are utilizing it.

- Interoperability:
    -

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