Software Requirements Specification

HelpMate Mobile App

Sirisha Veeraganta

Table of Contents

1. Introduction 1
1.1 Purpose 1
1.2 Scope 1
1.3 Definitions, acronyms, and abbreviations 1
1.4 References 2
2. Overall description 3
2.1 Product perspective 3
2.2 Product functions 3
2.3 User characteristics 3
2.4 Constraints 3
2.5 Assumptions and dependencies 4
3. Specific requirements 4
3.1.1 User interfaces 4
3.1.2 Hardware interfaces 4
3.1.3 Software interfaces 4
3.1.4 Communications interfaces 4
3.2 Functional requirements 5
3.2.1 User Class - The User 5
3.3 Performance requirements 8
3.4 Design constraints 9
3.5 Software system attributes 10
4. Prioritization and Release Plan 10
4.1 Choice of prioritization method 10

1. Introduction

This section gives a scope description and overview of everything included in this SRS document. Also, the purpose for this document is described and a list of abbreviations and definitions is provided.

1.1 Purpose

The purpose of this document is to give a detailed description of the requirements for the "HelpMate Mobile software. It will illustrate the purpose and complete declaration for the development of system. It will also explain system constraints, interface and interactions with other external applications.

1.2 Scope

HelpMate is a virtual assistant for android devices. Your very own personal assistant awaits your commands – performing tasks, notifying you about important events, and making your daily routine easier (and, often, more fun along the way).Our goal is to make working with everyday technology easier, more effective, and more fun so that you are free to enjoy the important things in life.

This product is best suitable to everyone who uses smart phone extensively. Helpmate communicates with the other apps on the mobile and sends the user updates and alerts.
Some of the important features of HelpMate:

- Parking lot assist (Park Mate)
- Stay Connected with contact favorites (Stay Connected Mate)
- GPS Tracker (GPS Mate)

Basically the software needs both Internet and GPS to fetch and display results. All system information is maintained in a database. The software also interacts with the GPS-Navigator software which is required to be an already installed application on the user's mobile phone.

1.3 Definitions, acronyms, and abbreviations

Table 1 - Definitions

Term	**Definition**
User	Someone who interacts with the mobile phone application
Admin/Administrator	System administrator who is given specific permission for managing and controlling the system
Web-Portal	A web application which present special facilities for restaurant owner
	and admin
GPS	Global Positioning System

GPS-Navigator	An installed software on mobile phone which could provide GPS connection and data, show locations on map and find paths from current position to defined destination
Application Store	An installed application on mobile phone which helps user to find new compatible applications with mobile phone platform and download them from Internet
HM	HelpMate
PM	ParkMate
SCM	Stay Connected Mate
GM	GPS Mate

1.4 References

[1] IEEE Software Engineering Standards Committee, "IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications", October 20, 1998.

[2] Feldt R,"re_lecture5b_100914", unpublished.

[3] Davis M A, "Just Enough Requirements Management: Where Software Development Meets Marketing", New York, Dorset House Publishing, 2005.

[4] Karlsson J, "A Cost-Value Approach for Prioritizing Requirements", Norges TekniskNaturvitenskapelige Uni. 1997

1.5 Overview

The remainder of this document includes three chapters and appendixes. The second one provides an overview of the system functionality and system interaction with other systems. Further, the chapter also mentions the system constraints and assumptions about the product.

The third chapter provides the requirements specification in detailed terms and a description of the different system interfaces. Different specification techniques are used in order to specify the requirements more precisely for different audiences.

The fourth chapter deals with the prioritization of the requirements. It includes a motivation for the chosen prioritization methods and discusses why other alternatives were not chosen.

The Appendixes in the end of the document include the all results of the requirement prioritization and a release plan based on them.

2. Overall description

This section will give an overview of the whole system. The system will be explained in its context to show how the system interacts with other systems and introduce the basic functionality of it. It will also describe what type of stakeholders that will use the system and what functionality is available for each type. At last, the constraints and assumptions for the system will be presented.

2.1 Product perspective

This product is unique and independent by its features but there are many mobile apps in the market which work on the similar concept of interacting with user and handling the user's task more efficiently. Most of the other products are standalone independent software while HelpMate will try to integrate these applications into one product.

HelpMate needs to communicate with other apps on the phone in order to retrieve information and utilize that to alert or notify the user. HelpMate largely utilizes the GPS on the phone to communicate and keep the track record of its locations.

Helpmate also connects to the call history and updates its internal database with number of times calls made to each number. This information is used to classify the contacts list and favorites by prompting to user to help generate a favorite list.

2.2 Product functions

HelpMate largely utilizes the GPS on the phone to communicate and keep the track record of its locations. This information is fed to the database and is used for parking lot assist and GPS tracker.

Helpmate also connects to the call history and updates its internal database with number of times calls made to each number. This information is used to classify the contacts list and favorites by prompting to user to help generate a favorite list. At regular intervals of time, the user is suggested to get back in touch with that one such person, the user made calls more than a certain number of times. Thus HelpMate takes care of the social aspects of the user when the user is stuck in the daily chores of life.

2.3 User characteristics

HelpMate can be used by any person who carries a smart phone and who can download the app from app store.

2.4 Constraints

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.

The other constraint is that mobile is supposed to be a smart phone which can download the app and function. Regular mobiles will not have this provision.

HelpMate does require certain amount of memory space in the phone, not just for the installation/operational purpose and database but also for future upgrades of the software. When mobile is set to auto download, the mobile should have enough memory to accommodate new changes, hence more space.

2.5 Assumptions and dependencies

HelpMate requires a smartphone with Wifi or (4/3) G connectivity and basic working knowledge of a smartphone such as being able to install the app on the phone and operating an app. The smart phone should be GPS enabled. The smart phone should have tactile screen for interacting with the application. The phone should also have enough hardware requirements such as memory to run and store the data.

The user is required to download the app and install it in the phone. While downloading the user has to ensure the installation is complete without any errors. Partial download will not run the software.
The features are completely independent of each other although some features completely depend on the GPS software on the phone without which they cannot operate.

3. Specific requirements

This section contains all of the functional and quality requirements of the system. It gives a detailed description of the system and all its features.

3.1 External interface Requirements

This section provides a detailed description of all inputs into and outputs from the system. It also gives a description of the hardware, software and communication interfaces and provides basic prototypes of the user interface.

3.1.1 User interfaces

A first-time user of the mobile application should be able to download the app on to the mobile and run the app to install it. It also comes with manual for the first time user and the interface is very user –friendly and application is self-explanatory. It does not require any login username or Password. The first time user can open the app and maneuver the app to set the filters and criteria for the notifications

If the user is not a first-time user, he/she should be able to operate the software with ease. The user can open the app and maneuver the app to change the filters and criteria for the notifications if required.

3.1.2 Hardware interfaces

Since the mobile application have any designated hardware, it does not have any direct hardware interfaces. The physical GPS is managed by the GPS application in the mobile phone 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 GPS application in order to get geographical information about where the user is located. The communication between the database and the application consists of operation concerning both reading and modifying the data, while the communication between the database and the mobile application consists of only reading operations.

3.1.4 Communications interfaces

The communication between the different parts of the system is important since they depend on each other. The different app on the mobile are configures to the HelpMate and thus it retrieves the data and stores the required data to its database.

3.2 Functional requirements

This section includes the requirements that specify all the fundamental actions of the software system.

3.2.1 User Class - The User

3.2.1.1 Functional requirement 1.1

TITLE: Download mobile application
A user should be able to download the mobile application through either an application store or similar service on the mobile phone. The application should be free to download.

3.2.1.2 Functional requirement 1.2

ID: HM
Mobile application – Home Page
The home page has a very simple interface, easy to understand and maneuver. It contains 4 tabs which includes ParkMate, Stay Connected Mate, GPSMate.
When clicked on each item, it opens to respective page.

3.2.1.3 Functional requirement 1.3

ID: PM
Scenario: The ParkMate page looks very simple. It basically contains 3 buttons. A "Pin location" button, "Find the pin" button and "delete pin" button.
When the user arrives at a location and wants ParkMate to remember that location, he/she can simply go to the ParkMate page and click on the "Pin Location" button. What this basically does is, it connects to the GPS and fetches the Latitude, longitude details and sends that to its database to store it. When the user wants to go back to that pinned location, he/she can click on the "Find the Pin" button and the app goes to the database, retrieves the saved data and puts it on the GPS and gives the path to the location to the user. This app is used mainly for parking assistance. Often people forget where their car is parked and look around wasting time in order to spot it. This app can be best utilized then, it's very efficient, quick and time saving.

3.2.1.4 Functional requirement 1.4

ID: PM1
Pin Location button: There will be a button to pin the location. When user clicks on the button. It takes the latitude and longitude of the location and saves the details to the database.

3.2.1.5 Functional requirement 1.5

ID: PM2
Find the pin button: When the user clicks on the "find the pin" button, the app goes to the database, retrieves the saved data from the database and invokes Google maps app and gives the path to the location to the user. If a pinned location does not exist then the app should do nothing.

3.2.1.6 Functional requirement 1.6

ID: PM3

Delete pin button: This button is used to delete the pin that was created earlier. By clicking on this pin, it also deletes the stored data from the database if any exists.

3.2.1.7 Functional requirement 1.7

ID: PM4

Maps: When Google maps is invoked, it is supplied with current location and pinned location. Google maps will then show the directions to the user.

3.2.1.9 Functional requirement 1.9

ID: SCM

Scenario: When the HM app is downloaded, SCM feature communicates with the call history and screens the phone numbers of last 10 days and picks the number that was either dialed or received more than 3 times. This number is possible prospect of being added to the favorites of the SCM app. Before which, the app confirms with the user if the number could be added to favorites. Upon the approval of the user, the number is added to favorites. The app basically collects such numbers from the call history by screenings the call history on daily basis. The app makes a list of favorites and at regular intervals of time (The use can customize the interval), the app suggests the user to call or message the number added in favorites. That way the app ensures to alert the user to call or message all those numbers added to its favorites list. The user also has a choice to discard the suggestion by simply clicking cancel.

This app is best app of all because it is unique, it is a friendly app giving suggestions to the user to catch up with his/her old friends and stay in touch. In this busy world, such app comes very handy as it not only filters the call list and makes a favorites list but also ensures the user is notified to keep in touch with his /her close connections.

3.2.1. 10 Functional requirement 1.10

ID: SCM1

Favorites: This is a button when clicked opens to a new page which has the list that is generated based upon the number of time calls are made to a particular number. This list is not static, new numbers keep getting added. The favorites list is also customizable by the user, the user can delete the favorites at any given point of time.

3.2.1. 11 Functional requirement 1.11

ID: SCM2

Alert: Alert is a radio button option that is used by the user to customize the alert interval. The user can choose the alerts to be notified every 1, or 2, or 3, or 4 week. When user selects say “1” radio button, the App notifies the user about getting in touch with one of the favorites once every week.

3.2.1. 12 Functional requirement 1.12

ID: SCM3
App Detection: The app screens through the call history and selects the number that was called or received call by the user in the last 10 days. This number is added to the favorites list upon the approval of the user.

3.2.1. 13 Functional requirement 1.13

ID: SCM4
App checker: Once the number has been added to the favorites, the app monitors if any calls have been made by the user to that number in past few weeks (interval can be set by the user). The app suggests the user to contact the number to stay in touch.

3.2.1. 14 Functional requirement 1.14

ID: SCM5
Action alert: This alert box opens when the user is to be reminded to contact the favorite number, the user hasn't reached out in the past few weeks. The alert screen contains a "call", "message" and a "dismiss" button. These buttons are used to ask the user for an action. If the user clicks on the "call" button, the app dials the number. If the user chooses the "message" option, the app takes the user to the message create box. If the user chooses the "dismiss" button, the alert box closes.

3.2.1. 13 Functional requirement 1.13

ID: GPSM
GPSmate
Scenario: GPS mate is best used when the user frequently visits a place for a certain period of time and stops to visit the place, the GPSmate keeps a track record of this and reminds the user that it has been a while since the user has been there and he/her should could go there. For ideal example, if the user visits the gym regularly for a 15 days and stops to go after that, the app keeps a track record of this and suggests the user after a week that the user hasn't been working out so he/she should get some sweat out.
The GPSmate suggests the user more in a friendly manner and that's the unique aspect of it. It's a personal care taker who keeps a track of our outings and suggests based upon the location. For ex, if the user

3.2.1. 14 Functional requirement 1.14

ID: GPSM1
GPS detection: The app continuously detects the change of location and writes down the time at which the location changed to its database/memory. This data will be eventually used in giving suggestions

3.2.1. 15 Functional requirement 1.15

ID: GPSM2
Data collection: The app collects data for 10 days during which is tries to learn location pattern

3.2.1. 16 Functional requirement 1.16

ID: GPSM3
Alert: After the initial data collection the app notifies a user if a user hasn't spent any time in a particular location over a few days. The details of time spent and number of days before alerting are implementation details and hence not included here.

3.2.1. 16 Functional requirement 1.16

ID: GPSM4
Alert Design: The dialog box is very simple. It has no buttons, just a message notification to the user suggesting/reminding him/her.

3.2.1. 17 Functional requirement 1.17

ID: GPSM4
Notification Interval: The user is notified whenever the GPS tracker shows a break in the frequent visits to a particular location. The break interval for notification is 7 days after 15 days of initial learning period for keeping a track record of user's GPS locations.

3.2.1. 18 Functional requirement 1.18

ID: GPSM5
Notification Condition: The user is notified once in every 7 days if the GPS tracker doesn't see the user make a visit to that location after a certain break period.

3.2.1. 19 Functional requirement 1.19

ID: GPSM6
Deletion: The user can go to the app page and delete the track location if he/she does not want any reminders from the app.
The user can visit the main page for the GPSM where all the recurring location visits are listed. Across each listing, there is a delete button. The user can click the button to delete that particular listing if he/she doesn't want any reminders for that location.

3.3 Performance requirements

The requirements in this section provide a detailed specification of the user interaction with the software and measurements placed on the system performance.

3.3.1 Parkmate

ID: PM
TITLE: Location activity
The Parkmate feature doesn't require user to do anything besides press "pin Location" when arrived at a destination or "Find the pin" button to find and "delete the pin" to delete the used pinned location.
These are the only 3 buttons the user is required to operate, it's the only interaction user has with this feature.

3.3.2 StayConnectedMate

ID: SCM1
TITLE: Favorites Approval
The user has to approve the number selected by the app to add to favorites. It's the user's choice whether or not to add that number to favorites. The pop-up window has a message "Would you like to add the caller to your favorites?" The buttons "Yes" or "No" is offered on the window box for the user to pick. This is the simple operational activity to be carried out by the user.

3.3.3 StayConnectedMate

ID: SCM2
TITLE: Contact the Favorite number
The pop-up window opens to suggest the user to either call the favorite number or message. The user can also choose to dismiss the message. The suggestion the dialog box would be "would you like to get back in touch with this caller?" The buttons "call" for calling, "Message" to leave a message and "Dismiss" to discard the suggestion is offered to the user.

3.3.3 GPSMate

ID: GPSM
TITLE: Reminder
This app is basically a suggesting app to the user. The user receives a notification with a suggestion - "It's been a while, would you like to visit xyz place?" The notification goes in the notification center, the user can refer to it whenever he/she would like to.

3.4 Design constraints

This section includes the design constraints on the software caused by the hardware.

3.4.1 Application memory usage

ID: HM
TAG: ApplicationMemoryUsage
GIST: The amount of Operate System memory occupied by the application.
SCALE: MB.
METER: Observations done from the performance log during testing
MUST: No more than 20 MB.
PLAN: No more than 16 MB
WISH: No more than 10 MB
Operate System: DEFINED: The mobile Operate System which the application is running on.
MB: DEFINED: Megabyte.

3.5 Software system attributes

The requirements in this section specify the required reliability, availability, security and maintainability of the software system.

3.5.1 Reliability

ID: HM

The application is very reliable. The results generated are tested and deliver complete accuracy and reliability. The features PM, SCM, GPSM also offer absolute reliability in the data produced and suggestions rendered to the user.

3.5.2 Availability

ID: HM

The software is available every time of the year at every hour of the day. The application runs throughout collecting data and generating lists for the user. The system is available always.
The internet connection should be connected to the internet in order for the application to communicate with the database.
The GPS device should be connected in order for the application to get the users location, the map and to calculate the distances etc.

3.5.3 Security

ID: HM

The communication security between the system and the other apps like GPS and call history.

3.5.4 Maintainability

ID: HM

The system is very low maintenance. The data generated is being cleared instantaneously after every use. The software can be further modified and many features can be added to it in future but for the current scenario, there will no updates required and it's a bug free software.

4. Prioritization and Release Plan

In order to get a view of how to divide the requirements into different releases and what requirements should be included in which release, a prioritization of the requirements is needed. This section discusses the choice of prioritization methods and gives a suggestion of how the release plan for these requirements could look like.

4.1 Choice of prioritization method

When prioritizing the requirements the ten most important ones were picked out first. This was done with a simple "1 to 10" ranking method, with one being "not important" and ten "very important". Based on the elicitation meetings, and the perceived ideas of what was important to the different stakeholders, a number was set for each requirement. The numbers were then summed up for each requirement and the ten with the highest score were chosen to be prioritized with the cost value approach.

The remaining requirements were prioritized according to the "Five-Way Priority Scheme". This method was chosen since it gives the different stakeholders the same importance and has an enough wide range for determining which requirement is more important than the other. However, in this prioritization process, the development team was not included.

Other methods for prioritization, such as the hundred-dollar test and the yes-no vote, were also considered. The hundred-dollar test is quite similar to the five-way priority scheme, since it also gives a wide range for ranking the requirements. However, it is more easily misused since someone could save all their money and put them on a requirement that they think is very important. Others might not agree that this requirement is important but it might still get the most votes since one person cared about it.

The yes-no vote method might be fairly simple to carry out, however the range is too narrow. For instance, if two requirements are not very important it would be hard to determine which of those requirements that is more important than the other.

In conclusion, weighing the disadvantages and advantages of these methods against each other lead us to choose the five-way priority scheme.

4.2 Release Plan

The requirements were divided into three releases based on the prioritization and their dependencies. The three different releases were assembled so that each would work as a fully functional application.

In the first release the requirements that build up the foundation of the application were included, together with the most highly prioritized requirements and their dependencies.
The second release also includes important requirements. However, these requirements are not vital for a functional application. They are more suited to act as additional features that can contribute to making the software product more attractive.

The third release includes the requirements that can be afforded to discard if the project gets delayed or overruns the budget.