Software Requirements Specification

Documentation By:

Aditya Mahajan - 323043

Viraj Pathak -323052

Vishwajeet Patil - 323055

Nikhil Karve -323030

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.

PURPOSE

The purpose of this document is to give a detailed description of the requirements for the "BusHop" mobile application. 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. This document is primarily intended to be proposed to a customer for its approval and a reference for developing the first version of the system for the development team.

The main purpose of this project is to track nearby bus-stops using Google Maps and Google Nearby Places API. Location tracking and Nearby Places tracking is one of the areas where app development has flourished.

The main intention was to help general public who find it difficult to understand the bus-system in Pune. The purpose of having location tracking feature in this project is to help a person who doesn't know about his current whereabouts and is a frequent bus traveller.

SCOPE

The scope of BusHop is restricted to following functionalities:-

Login-Signup	Authenticates the user and provides access to further functionalities. Signup allows new users to register.
Map	Map will provide user-location and shows nearby bus-stops..
Search	Allows a user to search required bus he/she wants to take.

DEFINITIONS, ACRONYMS AND ABBREVIATIONS

- Firebase

  Firebase provides a real-time database and backend as a service. The service provides application developers an API that allows application data to be synchronized across clients and stored on Firebase's cloud.

- Firebase Auth

  Firebase Auth is a service that can authenticate users using only client-side code. It supports social login providers Facebook, GitHub, Twitter and Google (and Google Play Games). Additionally, it includes a user management system whereby developers can enable user authentication with email and password login stored with Firebase

- API

  In computer programming, an **application programming interface** (**API**) is a set of subroutine definitions, communication protocols, and tools for building software. In general terms, it is a set of clearly defined methods of communication among various components. A good API makes it easier to develop a computer program by providing all the building blocks, which are then put together by the programmer.

- Google Map

  Google Maps is a web mapping service developed by Google. It offers satellite imagery, street maps, 360° panoramic views of streets, real-time traffic conditions, and route planning for travelling by foot, car, bicycle, or public transportation.

REFERENCES

- Google Map API Documentation:
  https://developers.google.com/maps/documentation/android-sdk/intro
- Google Places API Documentation:
  https://developers.google.com/places/web-service/intro
- Firebase Documentation:
  https://firebase.google.com/docs/android/setup
- Firestore Documentation:
  https://firebase.google.com/docs/firestore/

OVERVIEW OF DOCUMENT

The next chapter, the Overall Description section, of this document gives an overview of the functionality of the product.
It describes the informal requirements and is used to establish a context for the technical requirements specification in the next chapter.

The third chapter, Requirements Specification section, of this document is written primarily for the developers and describes in technical terms the details of the functionality of the product.
Both sections of the document describe the same software product in its entirety.

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.

SYSTEM ENVIRONMENT

The overall system is divided into two sub-systems the Map and the Database.
The user can interact with both of these systems once he has successfully authenticated himself/herself.
He/she can navigate between these sub-systems through a Navigation Drawer.
The Database Administrator has access over the entire database and keeps it updated in real-time if there are any changes in the bus-schedule.

HARDWARE INTERFACES

The application 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.

SOFTWARE INTERFACES

The application communicates with the GPS in order to get geographical information about where the user is located. It communicates with Google Places in order to get location of bus-stops nearby.

For authentication of user, it communicates with Firebase Auth.
The database which contains information of all the buses and their timings is hosted on Firestore and is linked with the app.

PRODUCT FUNCTIONS

BusHop will have three important sections:

- Login - Signup
- Map
- Search using Firestore database

Login-Signup:

Sign-up should allow the user to Register. During Registration the user should provide the email-id and valid (at least 6 characters long password). This registration will be linked up with Firebase Authentication.

Login should verify these details and allow access to further functionalities within BusHop.

Map:

The main functionality of the map is to show the user location and the nearby bus-stops around the user location. Markers are placed on nearby bus-stops.

The user-location should change dynamically.

Search:

The Search should allow the user to search Buses with the help of 3 criteria's viz. Route Number, Source and Destination. It should give the list of buses as the output.

USER CHARACTERISTICS

This application is mainly intended for general public. There are is no special skill set that is required to operate the application.

The mobile application users can use this application to find a bus-stations. This means that the user have to be able to search for buses, choose a bus from that search and then navigate to it using in-built map. In order for the users to get a relevant search result there are multiple criteria the users can specify.

CONSTRAINTS

The mobile application is constrained by the system interface to the GPS navigation system within the mobile phone. Since there are multiple system and multiple GPS manufacturers, the interface will most likely not be the same for every one of them.
Also, there may be a difference between what navigation features each of them provide.

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.

ASSUMPTIONS AND DEPENDENCIES

One assumption about the product is that it will always be used on mobile phones that have enough performance. If the phone does not have enough hardware resources available for the application, for example the users might have allocated them with other applications, there may be scenarios where the application does not work as intended or even at all. Also, the mobile phone should satisfy the minimum and maximum SDK requirements for 16 and 28 respectively.

Another assumption is that the GPS components in all phones work in the same way. If the phones have different interfaces to the GPS, the application need to be specifically adjusted to each interface and that would mean the integration with the GPS would have different requirements than what is stated in this specification.

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.

EXTERNAL INTERFACE REQUIREMENTS

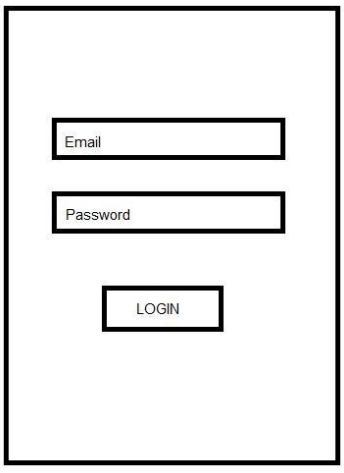
This section provides a detailed description of all inputs into and outputs from the system. It also provides basic prototypes of the user interface.

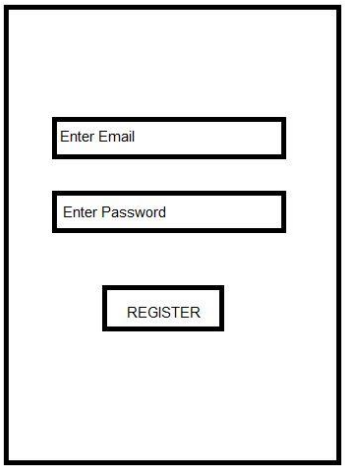
USER INTERFACES

A. On-Boarding Slides:

The on-boarding slides should give a brief overview of what the application is about when the app is initially installed. Skip and Next options should be provided.

B. Login/Signup Interface:

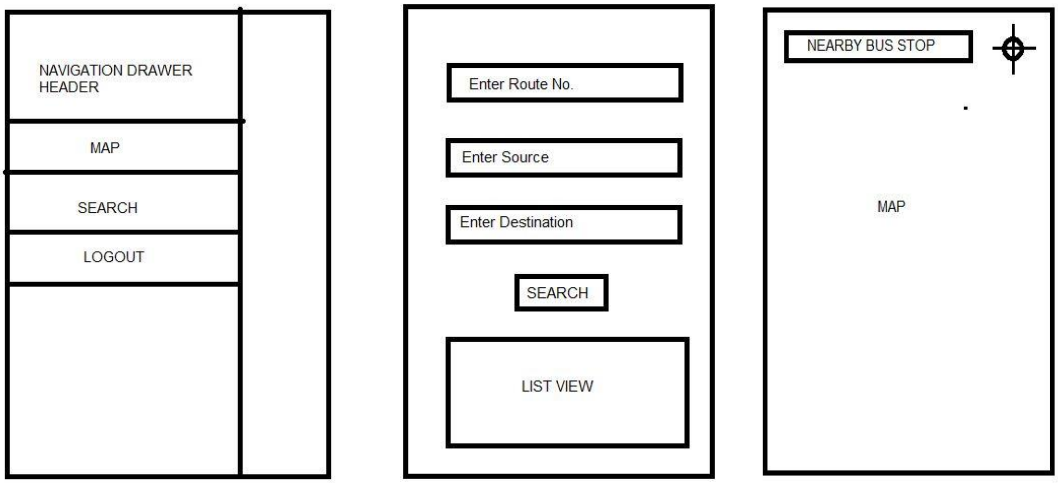




In Login interface the user should be provided with two fields to enter his email and password. A 'Login' button which authenticates the person and gives/denies access to further functionalities to him/her.

The Sign-up is a registration activity where the user has to again enter the same details as mentioned above with a 'Signup' button which adds him/her to our authentication database.

C. Home Interface:



Home interface should contain 2 fragments viz. Map and Search. It has a navigation drawer which allows user to switch between the two fragments.

Map fragment shows the map through the API and the search fragment provides three fields viz. route no., source and destination for searching.
A table containing search results should pop-up.

Lastly, the home interface's drawer should contain a Log-out button which logs the user out.

FUNCTIONAL REQUIREMENTS

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

Functional requirement 1.1

ID: FR1
TITLE: Download mobile application
DESC: 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.
RAT: In order for a user to download the mobile application.
DEP: None

Functional requirement 1.2

ID: FR2
TITLE: Download and notify users of new releases

DESC: When a new/updated version or release of the software is released, the user should check for these manually. The download of the new release should be done through the mobile phone in the same way as downloading the mobile application.
RAT: In order for a user to download a new/updated release.
DEP: FR1

Functional requirement 1.3

ID: FR3
TITLE: User registration
DESC: Given that a user has downloaded the mobile application, then the user should be able to register through the mobile application. The user must provide e-mail address and password.
RAT: In order for a user to register on the mobile application.
DEP: FR1

Functional requirement 1.4

ID: FR4
TITLE: User log-in
DESC: Given that a user has registered, then the user should be able to log in to the mobile application. The log-in information will be stored on cloud (Firebase Auth) and in the future the user should be logged in automatically.
RAT: In order for a user to register on the mobile application.
DEP: FR1, FR3

Functional requirement 1.5

ID: FR5
TITLE: Map
DESC: Given that a user has logged in successfully, he can view his location as well as nearby bus-stops around him/her. The user location should dynamically change. If the user disturbs the map, then a button should be provided in order to refocus the map again on user location.
The radius for bus-stop results should be 1km. Markers should be placed on nearby bus-stops. If the user selects one of the bus-stops then functionality should be given for redirecting to Google maps for navigating to the actual place.
This entire functionality should be disabled if the user location is not enabled.
RAT: In order for a user to know nearby bus stations.
DEP: FR4

Functional requirement 1.6

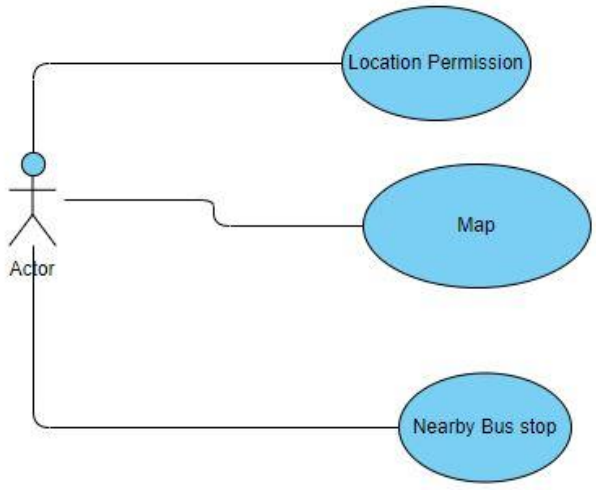
ID: FR6
TITLE: Search
DESC: Given that a user has logged in successfully, he can search the buses according to his requirement. There will be three search criteria:- Route No., Source and Destination.
The result of the search will be in form of a list/table. The search result should also contain timing of bus arrival at the source bus-stop.
RAT: In order for a user to search required buses.
DEP: FR4

FUNCTIONAL REQUIREMENT SPECIFICATION:

For this specific module we have divided user into two different roles for convenience of explanation.

→Navigator: The user who wants to find bus-stops through map.
→Searcher: The person who wants to search buses through database.

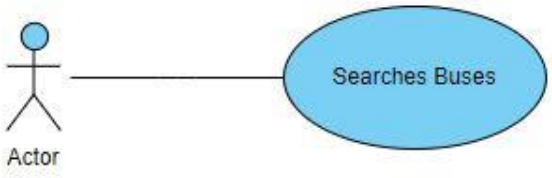
Use Case Diagram: Navigator



Description:

i. The Navigator initially gives permission for accessing location.
ii. Then he/she has to turn on their device location through settings.
iii. The map fragment will now be accessible to them.
iv. Once he/she clicks the 'Nearby Bus-stop' button, markers should be placed over the nearby bus stations surrounding him/her.

Use Case Diagram: Searcher



Description:

i. Searcher can search his/her required bus through three keys: Route No., Source and Destination
ii. A table should pop up if the data that he/she has entered is valid.
iii. Appropriate error warnings should be displayed if no data is inserted.

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.

ID: QR1
TITLE: Prominent search feature
DESC: The search feature should be prominent and easy to find for the user.
RAT: In order to for a user to find the search feature easily.
DEP: none

ID: QR2
TITLE: Usage of the search feature
DESC: The different search options should be evident, simple and easy to understand.
RAT: In order to for a user to perform a search easily.
DEP: none

ID: QR3
TITLE: Search Results in list view
DESC: The results displayed in the list view should be user friendly and easy to understand.
RAT: In order to for a user to view search results in efficient manner.
DEP: none
ID: QR4
TITLE: Usage of the result in the map view

DESC: The results displayed in the map view should be user friendly and easy to understand. Selecting a pin on the map should only take one click.
RAT: In order to for a user to use the map view easily.
DEP: none

ID: QR5
TAG: ResponseTime
GIST: The fastness of the search
SCALE: The response time of a search
MUST: No more than 3 seconds 100% of the time.
WISH: No more than 1 second 100% of the time.

ID: QR6
TAG: ResponseTimeMap
GIST: The fastness of nearby bus-stop markers
SCALE: The response time of map
MUST: No more than 3 seconds 100% of the time.
WISH: No more than 2 second 100% of the time.

DESIGN CONSTRAINTS

Hard drive space:

ID: QR7
TAG: HardDriveSpace
GIST: Hard drive space.
SCALE: The application's need of hard drive space.
METER: MB.
MUST: No more than 50 MB.
PLAN: No more than 35 MB.
WISH: No more than 20 MB.
MB: DEFINED: Megabyte

Application Usage:

ID: QR8
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 70 MB.
PLAN: No more than 66 MB
WISH: No more than 30 MB
Operate System: DEFINED: Android

MB: DEFINED: Megabyte.

ID: QR9
TITLE: Internet Connection
DESC: The application should be connected to the Internet.
RAT: In order for the application to communicate with the database.
DEP: none

ID: QR10
TITLE: GPS Connection
DESC: The application should be connected to the GPS device.
RAT: In order for the application to get the users location, the map and to calculate the distance.
DEP: none

Maintainability:

ID: QR11
TITLE: Application extendibility
DESC: The application should be easy to extend. The code should be written in a way that it favours implementation of new functions.
RAT: In order for future functions to be implemented easily to the application.
DEP: none

Portability:

ID: QR12
TITLE: Application portability
DESC: The application should be portable with Android >= 16.
RAT: The adaptable platform for the application to run on.
DEP: none

LOGICAL DATABASE REQUIREMENTS

Database should be established on Firestore which is a noSQL database that keeps the data in sync across client apps through real-time listeners.

The fields that are to be maintained within the database are as follows:-

ROUTE NUMBER	PARTIAL KEY
SOURCE	PARTIAL KEY
DESTINATION	PARTIAL KEY
TIME	

APPENDIX

Keywords

Firestore:- Cloud Firestore is a cloud-hosted, NoSQL database that your iOS, Android, and web apps can access directly via native SDKs. Cloud Firestore is also available in native Node.js, Java, Python, and Go SDKs, in addition to REST and RPC APIs.

Navigation Drawer:- The navigation drawer is a UI panel that shows your app's main navigation menu. It is hidden when not in use, but appears when the user swipes a finger from the left edge of the screen or, when at the top level of the app, the user touches the drawer icon in the app bar.

Fragment:- A Fragment represents a behaviour or a portion of user interface in a Fragment Activity. You can combine multiple fragments in a single activity to build a multi-pane UI and reuse a fragment in multiple activities.