Software Requirements Specification

for

Quick-shoppe

Version 1.0 approved

Prepared by Viren, Rehan, Gaurang, Jasdeep, Shailendra

IIT Ropar

26-09-2017

Table of Contents

Table of Contents ii

Revision History ii

1. Introduction 1

1.1 Purpose 1

1.2 Document Conventions 1

1.3 Intended Audience and Reading Suggestions 1

1.4 Product Scope 1

1.5 References 1

2. Overall Description 2

2.1 Product Perspective 2

2.2 Product Functions 2

2.3 User Classes and Characteristics 2

2.4 Operating Environment 2

2.5 Design and Implementation Constraints 2

2.6 User Documentation 3

2.7 Assumptions and Dependencies 3

3. External Interface Requirements 3

3.1 User Interfaces 3

3.2 Hardware Interfaces 3

3.3 Software Interfaces 4

3.4 Communications Interfaces 4

4. System Features 4

4.1 System Feature 1 4

4.2 System Feature 2 (and so on) 5

5. Other Nonfunctional Requirements 5

5.1 Performance Requirements 5

5.2 Safety Requirements 5

5.3 Security Requirements 5

5.4 Software Quality Attributes 5

5.5 Business Rules 5

6. Other Requirements 6

Appendix A: Glossary 6

Appendix B: Analysis Models 6

Appendix C: To Be Determined List 6

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| QS 1.0 | 26-09 | First draft | 1.0 |
|  |  |  |  |

# Introduction

## Purpose

In this document, we describe the software requirements for a mobile application, currently named as Quick Shoppe. As part of this application we will develop a utility software that allows the users to shop on the go without having to carry the products while purchasing that they have chosen from a specific store.

## Document Conventions

The document layout closely follows the IEEE template for Software Requirements Specification (SRS).

The important features or points have been specified in **bold**.

API – Application Programming Interface

## Intended Audience and Reading Suggestions

This document is intended for testers as well as other developers. This document contains the requirements and specifications of the application to a minute detail.

## Product Scope

Quick-shoppe is a utility mobile application that allows the users to create a dynamic shopping list on their mobile devices while browsing through a store, without the need to carry the desired products in a shopping cart. The application scans the barcodes to identify products and creates a list of the products selected, which can be changed later at any time.

After shopping, the user, once after presenting the list to the shopkeeper has the choice to either wait for the him to collect the items or get it sent to your home via home delivery, provided the option is available at the shop.

# Overall Description

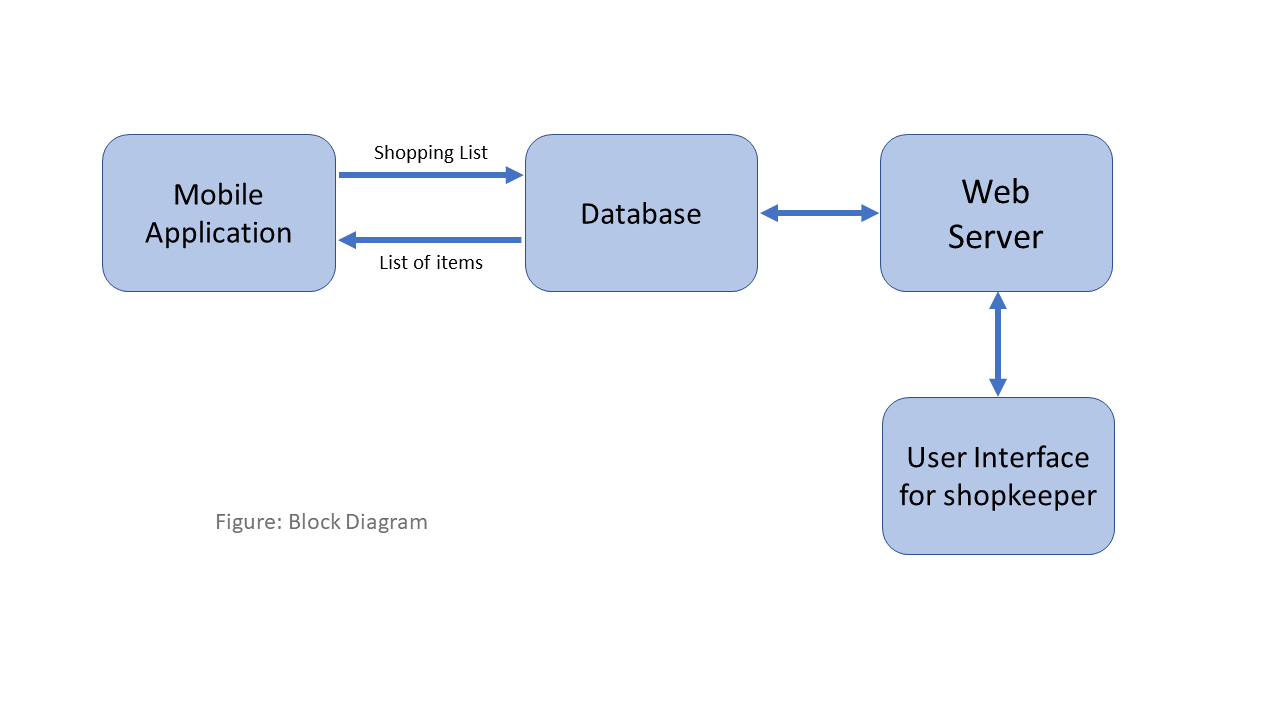
## Product Perspective

Quick-shoppe is an android based application that is intended for single user, non-profit purpose.

It is a self-contained product and is not a part of a bigger product/system.

It implements a database in the back, Google Firebase, containing the shops and their data of the barcode and the products corresponding to them.

Using this data, a shopper can quickly scan and identify the product in front of him/her. This is the part of the front end, which is the user interface. The user can add these products to the shopping cart in their app and increase/decrease the quantity as well. These products are then submitted to the counter via a simple checkout option and then it’s the work of the shopkeeper to collect these products and present it to the shopper while the shopper waits or send it to his/her address.



## Product Functions

Select the shop you are entering

Then browse the product on its counter

Scan the product’s barcode and the app recognizes it

Change the quantity to the desired value and move on to the next product

Click the checkout button when the list is complete

The list gets pushed to the server and a reference ID is generated, using which the user can confirm his/her order with the shopkeeper

The shopper can then move on to payment while the shopkeeper arranges the items

## User Classes and Characteristics

There are three types of users that interact with the system: users of the mobile application, shop owners and administrators. Each of these three types of users has different use of the system so each of them has their own requirements.

This is a utility application for personal use. The mobile application users are the common people who go to nearby shops for their daily needs. The user class requirements are android phone with internet connectivity. The frequency of use might be around once per week, when a person goes to shop to the local store.

The shop owners will not use the mobile application but the web portal instead. There they will manage the information about the order lists that they receive, with their reference IDs that will enable them to confirm orders and move onto payments.

The administrators also only interact with the web portal. They are managing the overall system so there is no incorrect information within it. The administrator can manage the information for each shop as well as the options for both the mobile application users and the shop owners.

## Operating Environment

Android is the working environment preferably version 5.1 and above.

The minimum requirements for the mobile phone on which the app would work are:

Camera, for reading the bar codes.

Internet connectivity which shall enable the app to read the database for the list of items and also push the list to the web portal of the particular shop.

## Design and Implementation Constraints

Android permissions like storage and camera permissions are required to run the app

We require the database of the products of the shop in which to implement this functionality

This database is implemented using Google Firebase and we would need to follow the protocols for the same while implementing the backend.

The application would read the database when you enter the shop, thus a good internet connectivity is required to sync the database and the values in it on the user’s mobile phone.

Both the web portal and the mobile application will be constrained by the capacity of the database.

The mobile application would require around 30MB of disk space on the device and a certain amount of memory (RAM) while running

The web portal would require to save cookies and other data to ensure login and other details are saved

Tools used: Google vision, an image processing API to quickly and efficiently read barcodes from the products.

## User Documentation

Proposed: A simple tutorial that takes first time user through the functions of different aspects of the application.

## Assumptions and Dependencies

One assumption about the product is that it will always be used on mobile phones that have enough performance. The application requires continuous use of camera while shopping and constant opening and closing of the camera. 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, leading to crashes.

# External Interface Requirements

## User Interfaces

Home screen contains a persistent search bar at the top to search for the shops. Below it on the page are the list of shops recently visited. For a first-time user, this screen is blank.

Clicking on the search retrieves items from the main database of shops and you can select the shop you are visiting.

Next screen shows the name of shop selected on the top and below shows that your cart is currently empty. Then you can click on start shopping and the next screen opens your camera and you can start scanning the products.

The top right corner of the screen shows the current value of your shopping list on which you can click and view the current list.

When a barcode is scanned, the product is shown and the app asks the user for the quantity of the concerned product, and the same is updated in the shopping list.

Then it goes back to the scanning page and the user can continue shopping

To end, the user must click on the shopping cart icon that will open the cart and select checkout button at the bottom of the screen. It is then that the list goes to the server and appears on the shopkeeper’s web portal

The shop owners and administrators interact with the system through a web-portal **(to do)**. A shop owner should be able to register on the web-portal to log in and manage the information. An administrator should also be able to log in to the web-portal where he/she can administer the system by for instance editing shop information.

## Hardware Interfaces

The application makes use of mobile camera in order to scan and identify the products 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 software is built on Android platform, so the application is created pertaining to the coding paradigms of Android.

The tools and libraries used like vision API and Google firebase would require the application to follow its rules and regulations.

The communication between the database and the web portal 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.

## Communications Interfaces

The underlying operating systems can take care of the background communication processes that take place in the mobile application and the web portal, since no special type of communication protocols is being used in either of them.

# System Features

<This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.>

## Getting Permissions

### Description and priority

The application requires permissions to use camera and location.

Priority - Medium

### Stimulus/Response Sequences

The user needs to accept all permissions by allowing them. Then only the application would work further. Once the system has received these permissions, the user further proceeds.

### Functional Requirements

REQ-1: Camera permissions:

Camera is needed for scanning the barcode.

REQ-2: Location permission:

This allows the application to know your current location to suggest stores near you.

## Suggesting stores nearby user’s current location

### Description and priority:

The application suggests the nearby stores to the current location of a user (10 km displacement from current location). The user can then select the required store and proceed further.

Priority: High

### Stimulus/Response Sequences:

The user has to select a particular shop from the list of shops registered on our portal. Once the shop is selected, there are two options – Get the route or enter into the shop.

### Functional Requirements

REQ-1: Get the route:

It will give route on the map from user’s current location to the selected shop.

REQ-2: Get the route to a different shop in the map:

If the user wants to get route to a different shop in the map without going back to the search shop option again, the user can just select the starting and destination positions by clicking on the map. It will give the route between those 2 positions.

REQ-3: Navigation:

If the user wants to navigate to the shop, just click on the destination shop(shown by the red bubble). It will pop an option of navigation, clicking on that option will take the user to google maps navigation.

## Create User Id:

### Description and priority:

Each user must be assigned a unique user id once he enters the shop. This id will help him to place his order uniquely.

Priority: High

### Stimulus/Response Sequences:

Once the user selects a shop, an id is assigned to him.

### Functional Requirements

REQ-1: Unique Id

No two persons present in the shop must not have same customer id.

REQ-2: Short ID:

Id must not be too long. These are simple integers. The database is cleared at the end of the day so that fresh id’s can be generated the next day.

## Scan the barcode of products

### Description and priority:

This feature uses Google vision API to scan the barcode of a product kept in the store. This will retrieve the barcode of the product.

Priority: High

### Stimulus/Response Sequences:

Place the camera above the barcode of the product until the barcode is read. Once it is read, then the user proceeds further to add items

### Functional Requirements

REQ-1: Read the barcode

This will retrieve the barcode from the product and store it in the firebase database as a purchase of item.

## Add and delete items

### Description and priority:

The items must be added and deleted from the cart as per user requirements. The quantity should be modified by the user.

Priority: High

### Stimulus/Response Sequences:

The user clicks add button to add the item into cart and the UI gives him the option to change the quantity of item. Then he presses add button to add the item or cancel to go back.

### Functional Requirements

REQ-1: Add products and cancel option:

The system must allow adding items to the cart and should give the option to cancel adding them.

REQ-2 : Delete items :

The user must be able to delete the items added in the cart.

## Change shop on the run

### Description and priority:

The user must be able to switch to a new shop once he decides to discard shopping in that particular store.

Priority: low

### Stimulus/Response Sequences:

The user presses the change shop button and is given a warning that this will delete his entire cart from that particular shop.

### Functional Requirements

REQ-1: Clear data in the present shop:

Once the user changes his/her shop, his/her data will be deleted from that shop’s database.

REQ-2 : Switch shop

The user must be redirected to the home page where he can select a new shop to enter.

## Shopkeeper should be able to manage orders

### Description and priority:

The shopkeeper should be able to manage the orders of the users who approach him for conveying their respective customer id’s.

Priority :medium

### Stimulus/Response Sequences:

The shopkeeper clicks on place order button which places the customer’s order with respect to his ID. He can also update his store and see the list of exhausting items in his store.

### Functional Requirements

REQ-1: Update items present in his store in the Quick-shoppe database

He can add items, edit items, delete items from the database corresponding to his store.

REQ-2 : Check for exhausting items

The shopkeeper can view the list of items which have a very low quantity in his store just as a warning for him.

# Other Nonfunctional Requirements

## Performance Requirements

The barcode must be scanned very quickly and efficiently so that is a performance benchmark that we would like to work on. However, since we are using Google vision API that would scan the barcode, it would not be controlled much.

4.1.1: Persistent search bar in home page

The search bar should be easily accessible when the application is opened  
 This enables the user to search for the shop easily and quickly

4.1.2: Usage of the result in list view

The results of shop names should be user friendly and easy to understand and selecting the shop name takes only one click

4.1.3: Fastness of the search

The search results of the shop names should be quickly shown according to what input the user has provided.

The search needs to be dynamic that would save the user from typing the whole name of the shop for selection  
 Preferred performance measure: 0.1 seconds for a dynamic search

4.1.4 Usage of the list of recent shops

This list enables a user to quickly select the shop one’s visiting to skip the search and continue with selection of the shop in a single click from the home page

4.1.5 System Dependability

This is the measure of how the app will respond if the device loses internet connectivity and tries to reconnect to the internet  
 The user will be prompted to check the device’s internet connection

The application shall save its current state and wait for the internet connection

## Safety Requirements

This is a mobile application, specifically in the utility section. So, there are no safety requirements for this application.

## Security Requirements

The application communicates with the database of the products of that shop and reads the values (quantity) of the products. However, these values should not be accessible from outside and immutable via the application, since the application performs only the Read operation on the database.

The web portal implements login system for administrator as well as the shopkeepers. The privileges and the control authority with the administrators should be limited to them only and not accessible by others.

The data of one shop is not visible and accessible to other shops, preventing them from making any changes as well.

## Software Quality Attributes

4.4.1 Reliability

The system gives a right result on search, both the shop and the product

The quantity and the product the user selects gets correctly updated in the list

The checkout function correctly sends the list to the shop owner, without any loss of information

4.4.2 Availability

The system is available always, since this is an online application and cannot work without communicating with the backend

Internet connection is required always for the mobile application and the web portal

4.4.3 Security

The login systems are robust enough to prevent login with wrong credentials

The information of one shop owner is not visible to other

4.4.4 Maintainability

The web portal can be extended later to work directly in sync with the shop’s own database, updating the quantities of the products in real time

The test environments built for the application allows testing of the different components of the application separately

The application can be easily extended as the code is written in a way that favors easy implementation of new functions

4.4.5 Portability

The application, currently developed in Android can be later ported to other platforms like iOS