Software Requirements Specification

for

Product Wizard

Version 1.0 approved

Prepared by K.O.D

K.O.D

December 13, 2018

Table of Contents

Revision History

Name	Date	Reason For Changes	Version
Product Wizard	DEC 14, 2018	Initial Release	1.0.0

1. Introduction

1.1 Purpose

Product Wizard is an application that allows the user to scan the barcode of a particular product and displays the product information of that item.

1.2 Document Conventions

This document follows MLA Format. Bold faced text has been used to emphasize section and subsection headings. Highlighting is to point out words in the glossary and italicized text is used to label and recognize diagrams.

1.3 Intended Audience and Reading Suggestions

This document is to be read by the development team, the project managers, testers and documentation writers. This application has been developed to be useful for people looking for product information above the age of 15.

1.4 Product Scope

Product wizard is a lightweight application that allows its users to scan or input the barcode of products. The app then returns the information of that specific Product. For example, scanning the barcode of a water bottle will display the name of the maker, origin, nutritional content and the size of the water bottle. This application is beneficial to users because it allows them to get product information about an item instantly anywhere they are at. Users can additionally generate a database for personal or enterprise use.

1.5 References

IEEE SRS Document Template

https://web.cs.dal.ca/~hawkey/3130/srs_template-ieee.doc

Firebase Setup Documentation for Web

https://firebase.google.com/docs/web/setup

2. Overall Description

2.1 Product Perspective

Product wizard is a new application which utilizes the camera to enable the users to scan the barcode of many different products; users are also allowed to use barcode numbers to search products, the result will be based on the number the user inputs. If the product isn't registered then users have the ability to enter the product information and add it to the users personalized product list.

2.2 Product Functions

Product wizard offers these functions:

1. Scans a barcode, image (containing barcode), recognizes if the item has been stored before; if it has been stored before the product information is displayed to the user, if not it the user is prompted to add the product information manually and is saved to the users homepage. An exemplary use of this functionality is scanning the barcode of an item online which will bring up the products' information.

2. Accepts typed barcode numbers in text, recognizes if the item has been stored before; if it has been stored before the product information is displayed to the user, if not it the user is prompted to add the product information manually and is saved to the user's homepage.

3. Allows the user to be able to edit and delete existing product on the user's homepage.

4. The user is able to export the products saved to the device's storage

2.3 User Classes and Characteristics

Possible User Classes are listed below:

1. Nutritionist
2. Inventory Manager
3. Customer

2.4 Operating Environment

The system will be written to operate on the android operating system with an IOS variant to be compiled at a later date. As a result of the application being a hybrid app it can run

on most versions of Android however the recommended minimum version of Android to run this application is Android 4.4 (KitKat).

2.5 Design and Implementation Constraints

1. For full working application requires Internet connection. Notify the user accordingly if not present

2. Considering the audience the application should be available in multiple languages, even if it a case in which the initial release is only available in English, other languages should be included gradually with future updates

2.6 User Documentation

Google's Firebase Documentation

https://firebase.google.com/docs/?authuser=0

This documentation is required for the implementation of the Firebase API

The following Cordova plugins are required for the necessary native functionalities:

- cordova-plugin-dialogs 2.0.1 "Notification"
- cordova-plugin-headercolor 1.0 "HeaderColor"
- cordova-plugin-statusbar 2.4.2 "StatusBar"
- cordova-plugin-whitelist 1.3.3 "Whitelist"
- cordova-plugin-x-toast 2.7.0 "Toast"
- nl.madebymark.share 0.1.1 "Share"
- phonegap-plugin-barcodescanner 8.0.0 "BarcodeScanner"

The documentation for each plugin can be found on GitHub.

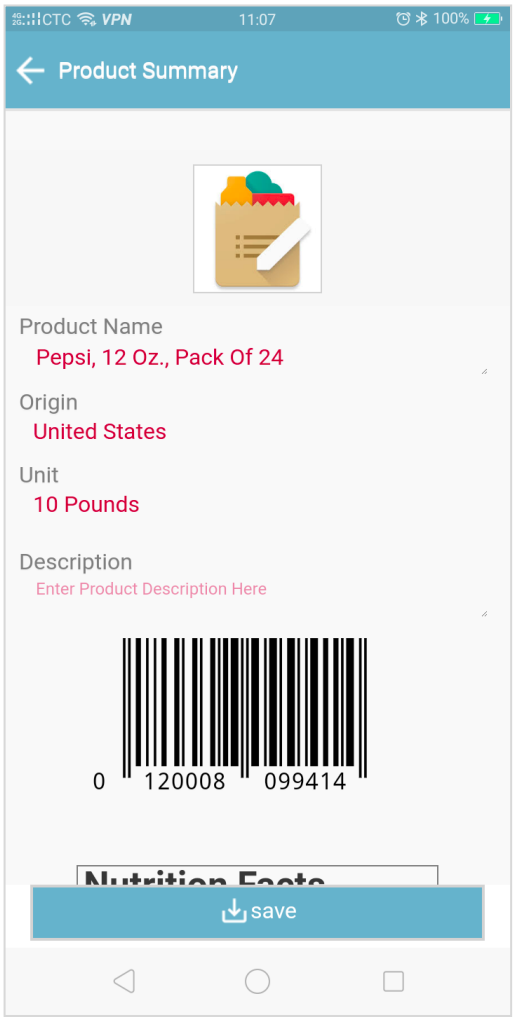
2.7 Assumptions and Dependencies

It should be assumed that the user is running an Android Device running at least KitKat as this is the minimum required version. Product wizard required access to Google Services. This is assumed because Google is accessible in most countries around the world. It is crucial that the android device has an external camera. Therefore an emulated android version will not be able to access the scanning functionality because it does not have a physical external camera.

3. External Interface Requirements

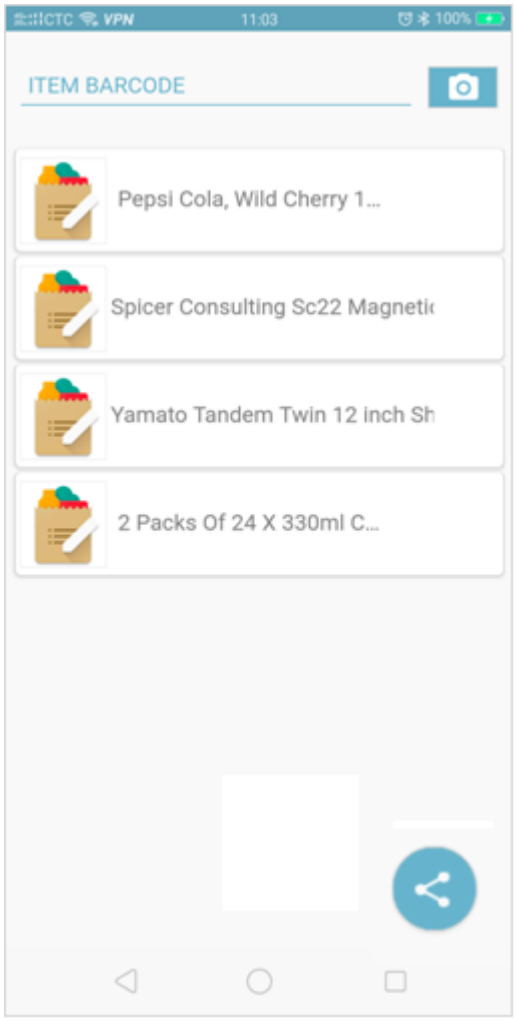
3.1 User Interfaces

Product Wizard's user interface has been designed with their users in mind. The home screen offers the user a choice between entering the numerical barcode or scanning the items barcode. The user can select one of the options, if the user clicks on the camera icon is taken to the Scanning Screen where all that is needed to do is to align the scanner with a barcode, after which the product information is loaded onto the Details Screen and the User redirected to said screen. The homepage displays all the items the user has scanned, when the user clicks on an item in the list they are taken to the items product information page. All fonts will be Arial. The UI also conforms to Google's Material design; utilizing flat colours and icons.



Details Screen

The Details Screen contains various fields that show the product information, some of which are customizable and may be saved to the device or updated on the database



Home Screen

The Home Screen contains a list of items saved to the device if any has been saved. There is also a single input field and a button allowing the user to either enter the product's EAN-13 Barcode manually or scanning it using the device's camera

```
const saveProduct = function(){
    new Promise(function(resolve, reject){
        var savedProducts = JSON.parse(localStorage["ProductWizard"]);
        savedProducts[currentBarcode] = createProductObject();
        localStorage["ProductWizard"] = JSON.stringify(savedProducts);
        toastMessageBottomShort("Product information saved to this device");
        loadSavedProducts();
    });
}

const removeProduct = function(){
    confirmUser("Are you sure you want to remove this item?", function(choice){
        if(choice == 2){
            //CANCEL
        }
        else{
            //REMOVE CONFIRMATION
            var savedProducts = JSON.parse(localStorage["ProductWizard"]);
            delete savedProducts[currentBarcode];
            localStorage["ProductWizard"] = JSON.stringify(savedProducts);
            toastMessageBottomShort("Removed Successfully");
            loadSavedProducts();
            setTimeout(function(){
                window.history.back();
            }, 500);
        }
    }, "Remove", ["OK", "CANCEL"]);
};
```

Code snippet showing how the saving and removal of products is to be implemented

Scanning Screen

The Scanning Screen provides a simple interface for scanning barcodes which is detected automatically. Once the barcode has been detected the App will search for its product information and display the results on the Details Screen if the information has been found. Users have the option of toggling the devices torch. By default the torch is enabled as it may be required in most scanning cases

```
scanBarcode = function(){
    return new Promise(function(resolve, reject){
        cordova.plugins.barcodeScanner.scan(
            function (result) {
                if(result.text[0] == 0)
                    result.text = result['text'].substring(1);
                resolve(result.text);
            },
            function (error) {
                alertUser("Scanning failed: " + error);
            },
            {
                preferFrontCamera : false, // iOS and Android
                showFlipCameraButton : false, // iOS and Android
                showTorchButton : true, // iOS and Android
                torchOn: true, // Android, launch with the torch switched on (if available)
                saveHistory: true, // Android, save scan history (default false)
                prompt : "Place a barcode inside the scan area", // Android
                resultDisplayDuration: 200, // Android, display scanned text for X ms. 0 suppresses it entirely,
                formats : "EAN_13", // default: all but PDF_417 and RSS_EXPANDED
                orientation : "portrait", // Android only (portrait|landscape), default unset so it rotates with
                disableAnimations : true, // iOS
                disableSuccessBeep: false // iOS and Android
            }
        );
    });
}
```

Code snippet showing how scanner functionality is to be implemented

3.2 Hardware Interfaces

A Mobile Hybrid Application, such as Product Wizard, runs on a series of Stack Layers. As such all software interfaces of the application has to pass through these layers before the hardware interfaces can be accessed. Below is a graphical representation:

Hybrid HTML5

In terms of any specific hardware components of the device used, the Product Wizard application only uses an inbuilt file storage interface called “localStorage” to store the user’s session information and other configurations.

3.3 Software Interfaces

Product Wizard makes use of a few interfaces to achieve functionality. They are listed below:

Google’s Firebase API
Product Wizard utilized Google’s Firebase for data storage and as such it also utilizes Google’s Firebase API to read and write information from said database.

3.4 Communications Interfaces

Google's Firebase database uses HTTPS handle inbound and outbound data transfers. As such the Product Wizard application securely connects to the server for each transaction.

4. System Features

4.1 Scan Barcode (U1)

Objective -User scans barcode of an item and the information of that product is shown.
Priority – High
Source –
Actors - User
Flow of Events
5.1. Basic Flow
5.1.1. User selects the scan barcode option
5.1.2. User's camera is opened.
5.1.3. User aligns the camera with the barcode
5.1.4. System tries to recognize the item's barcode
5.1.5. If recognized the products information will be displayed to the user.
5.1.6. The product's information is saved.

5.2. Alternative Flow - at 5.1.5 the system does not recognize the barcode the user will be asked to input the products information.

4.2 Enter EAN Number (U2)

Objective -User enters the EAN-13 barcode number of an item and the information of that product is shown.
Priority – High
Source –
Actors - User
Flow of Events
5.1. Basic Flow
5.1.1. User Enters the EAN-13 barcode number of a product.
5.1.2. System tries to recognize the item's EAN-13 barcode number.
5.1.3. If recognized the products information will be displayed to the user.
5.2. Alternative Flow - at 5.1.3 the system does not recognize the barcode the user will be asked to input the products information.

4.3 Select Product (U3)

Objective -User selects a specific product to see its information.
Priority – High
Source –
Actors - User
Flow of Events
5.1. Basic Flow
5.1.1. User selects a product from the list of products on the user's homepage.
5.1.2. The product's information is displayed to the user.
Preconditions – User has products saved to their homepage

4.4 Delete Product (U4)

Objective -User selects a specific product and deletes its information .
Priority – medium
Source –
Actors - User
Flow of Events
5.1. Basic Flow
5.1.1. Includes Select Product Use Case
5.1.2. User clicks on the trash bin icon
5.1.3. User is asked to confirm that they want the item deleted
5.1.4. The item is deleted
5.2. Alternative Flow 1 – At step 5.1.4 the product is not deleted
6 Includes
6.1. U3 – Select Product
Preconditions – User has products saved to their homepage

4.5 Edit Product Information (U5)

Objective -User selects a specific product and edits its information
Priority – medium
Source –
Actors - User
Flow of Events
5.1. Basic Flow
5.1.1. Includes Select Product Use Case
5.1.2. User clicks on the details of a specific item. For example product calories
5.1.3. User enters the new product detail
5.1.4. Changes are saved
6 Includes
6.1. U3 – Select Product
Preconditions – User has products saved to their homepage

4.6 Share Product List

Objective –App attempts to share a textual representation of the items saved to their device to clipboard
Priority – medium
Source –
Actors - User
Flow of Events
5.1. Basic Flow
5.1.1. User taps “Share” floating action button. Compatible share application are then shown
5.1.2. User selects the desired application
Preconditions – User has products saved to their homepage

5. Other Nonfunctional Requirements

5.1 Performance Requirements

It is crucial that the Product Wizard app performs database lookups quickly. Each barcode lookup consists of sending the server a request containing the product’s barcode as its payload. This request is send asynchronously as to not halt the application’s I/O operations until it receives a response.

5.2 Safety Requirements

There are no safety requirements associated with the Product Wizard Application as no user information is collected

5.3 Security Requirements

In order to ensure that any responses are not modified, the Product Wizard application should use HTTPS over regular HTTP to ensure that all data in secure during transit.

5.4 Software Quality Attributes

Adaptability
The Double Six App contains some level of adaptability as it is not affected by the user environment, specifically geographic location. Information acquired from the users is also of a fixed format and would affect the system regardless of what is inputted by the user. Strings are converted into UTF-8 before being handled by the server.

The Product Wizard application is unable to adapt to regions where Google Services are blocked, for instance, In China Google services are prohibited and as such the Product Wizard application would be unable to retrieve product information from the database without the use of a VPN.

Scalability
The Product Wizard Backend Service is very scalable because it utilizes Google's Firebase Database which is automatically adjusts its allocated resources based on the traffic it receives.

Availability
Google's Firebase database as with other Google software services is almost always available to users worldwide. As such users can enjoy the services that Product Wizard has to offer at any time of the day.

Correctness
Since key-value conformity is used to store information, where the keys in this case would be a barcode, the correct information for each product would be returned since the barcode for each item is unique.

Flexibility
The Product Wizard App possesses some level of flexibility in the instance where the user loses internet connection they are still able to view the information of the products the application thought the offline product that have been saved to the device

Interoperability
In order for the Double Six application to achieve all the necessary functionalities it needs to interface with many other servers through API's. As such it contains some interoperability.

Maintainability
The Product Wizard App would have to be maintainable, as new features an updates will have to be included as the app scales.

Portability
As a mobile App that runs on the Android OS, the Product Wizard app is portable by default.

Robustness
JavaScript by nature has great error handling capabilities. If an asynchronous function throws an exception it will only affect the Promise that encapsulates it, in addition to logging the exception. As such the App will continue to run

5.5 Business Rules

Google's Firebase service offer three plans, however it is free to use as long as a certain data transfer or storage threshold is not surpassed per month. During the embryotic phases the Product Wizard application not much firebase resources will be used. However as the application scales another plan might be induced. As such measures should be in place to cover monthly fees if any.

6. Other Requirements

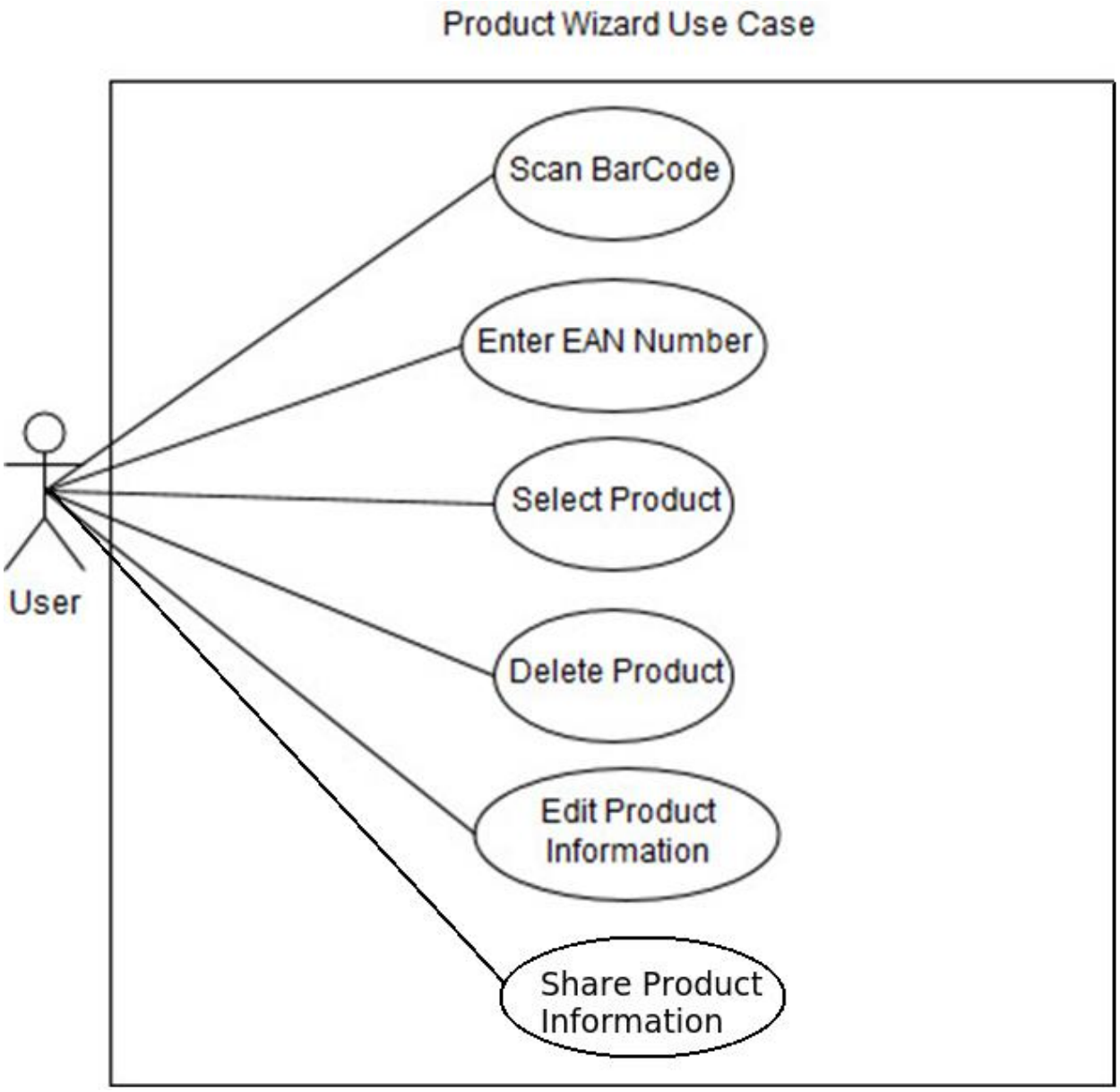
There are no other requirements to be discussed

Appendix A: Glossary

Firebase – An online database service offered by Google that uses a No-SQL like structuring for storing data.

HTTP – Hypertext Transfer Protocol

Appendix B: Analysis Models



Appendix C: To Be Determined List

- Ability to import exported JSON

Percentage Work Contribution

Below is a table showing the percentage effort contributed by each member of the group:

Name	ID	Effort (%)
Ottor Mills	180917	30
David Thomas	180912	23.33
Nicoy Smith	180902	23.33
Kenneth Anglin	180907	23.33

Course Instructor Thomas Canhao Xu

Course SWEN3006

Date October 25, 2018