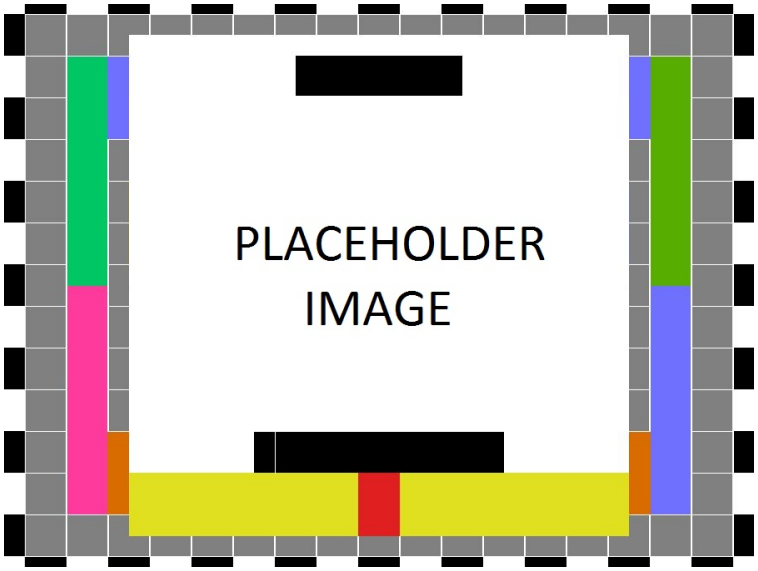
Department of Computer Science & Engineering
The University of Texas at Arlington

Detailed Design Specification
CSE 4316: Senior Design II
Summer 2019



EngiMavs
Swift Scan

Jacob Wilkins
Alex Mejia
Zohar Moreno
Sirjan Khanal
Umanga Shrestha

Revision History

Revision	Date	Author(s)	Description
0.1	10.01.2015	GH	document creation
0.2	10.05.2015	AT, GH	complete draft
0.3	10.12.2015	AT, GH	release candidate 1
1.0	10.20.2015	AT, GH, CB	official release
1.1	10.31.2015	AL	added design review requests

CONTENTS

List of Figures

List of Tables

1 INTRODUCTION

The project consist of a single mobile application that can be used to keep an inventory of products. The app scans a bar-code and proceeds to look for the number in the internal database, if the product has already been registered then the app will present the available information to the user but in case that the product is not found in the database the app will allow the user to add the product to the inventory.

The app consist of 3 main components, the user interface, the internal database and the ZXing library. The app uses the ZXing library (Zebra Crossing) to read the bar-code and then proceeds to look into the internal database, the database is created when the app is launched for the first time and can be modified both in the app and in the internal phone memory (not recommended).

2 System Overview

The product will have 3 layers: The mobile application layer, web application layer, and database layer. This section will describe what each of these layers do.

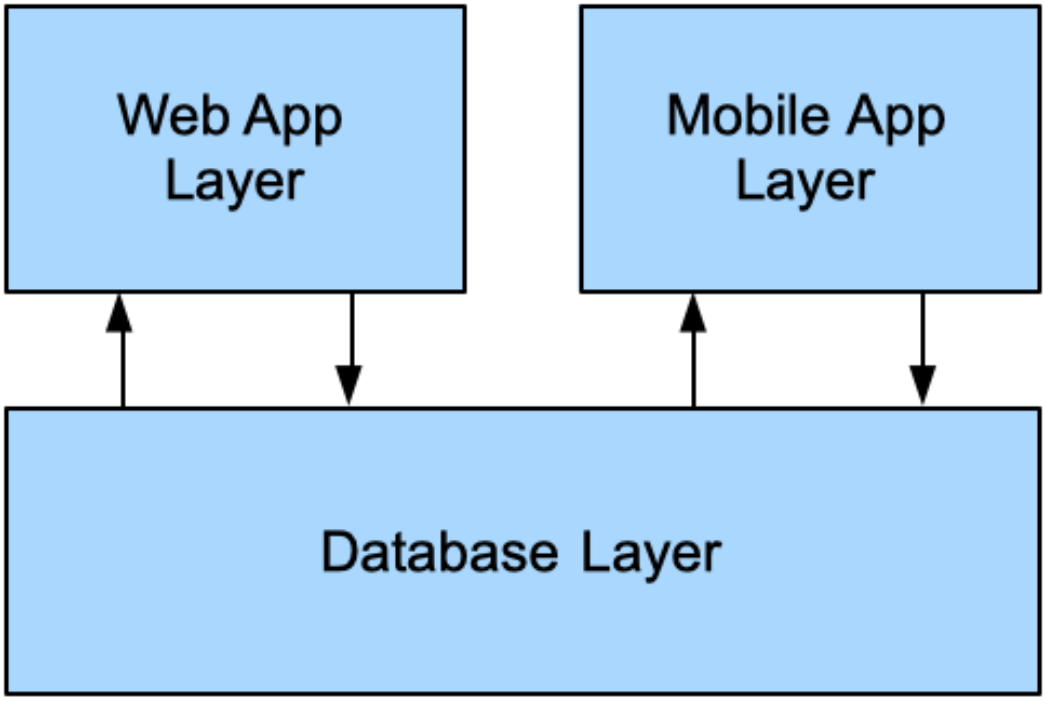


Figure 1: Swift Scan System Architecture

The mobile app layer will contain 3 tabs: profile, scan, and saved. The profile tab will ensure that all user settings are saved and can be synced with the web app. The scan tab will allow users to scan products using the camera and view information about them. If they choose to save the product information, it will be stored in the saved tab. The saved tab lists every product save by the user. These products can be sorted by type of alcohol or any other category. The user can also make collections with products of their choice.

The web app layer will include a profile tab and saved tab with the same features as the mobile app. The information will be synced from the mobile app via the database system so that the user can access their products anywhere. When the web page is refreshed, a request will be sent to the database for the up-to-date information on the user.

The database system is the driving force of the project. When a user scans a product, the database will be queried for the barcode number. Information such as name, brand, price, ABV, type, country, and category will be returned the user. In addition, the database will store the information of all users and will be monitored by the maintenance team. To login, the user will enter their username and password which will be queried in the database to confirm their identity and all of their setting setting will be returned.

3 X Layer Subsystems

3.1 Web Layer Hardware

Swift Scan app will no longer use a Web Layer to pull and store brewery information records. The Swift Scan app will pull and store information from an internal text file that'll be created once the user decides to manually add a brewery record or search for a specific brewery product.

3.2 Web Layer Operating System

Swift Scan will execute on Android OS smartphones.

3.3 Web Layer Software Dependencies

Swift Scan will use numerous libraries from the Android API and also several from the Java SDK. Due to the features that the GUI will provide, the Android API libraries will help accomplish this task and most of the back-end development will be accomplished with libraries from the Java SDK.

3.4 Web Layer Subsystem

The Web Layer was supposed to be used as a means for higher access to the data for the brewery products, but now the Web Layer is treated as an internal file that the app will create and store in the internal storage of an Android smartphone. There will be separate classes within the functionality of the app that'll open the file for adding and/or modifiying records and also to retrieve any information that pertains to a specific brewery. Therefore, an internal file stored in the Android OS will be more of the Web Layer.

3.4.1 Web Layer Subsystem Hardware

The Web Layer will not consist of any hardware components for functionality purposes.

3.4.2 Web Layer Subsystem Operating System

The main operating system (OS) for the overall app is Android OS.

3.4.3 Web Layer Subsystem Software Dependencies

Any of the class files for Swift Scan will use Android's API along with Java SDK libraries for development of the GUI and back-end development of the functionality of the app, respectively.

3.4.4 Web Layer Subsystem Programming Languages

The main programming language used for the Swift Scan app is Java and using Android's API for the GUI development.

3.4.5 Web Layer Subsystem Data Structures

Swift Scan will contain two classes to write data to the internal text file and the second class will be used to retrieve the data and display the data to the user. The second class where the data will be retrieved from the text file will have a method to sort the data as well so it can be displayed to the user. In addition, the second class will also contain a method for updating any records within the text file, depending on what the user decides to update.

3.4.6 Web Layer Subsystem Data Processing

In order for the brewery data to be recorded in the text file, the user will need to scan first the barcode of the product. The app will attempt to search for the brewery record in the internal text file and there will be a class that'll perform a linear search through the text file to see if the barcode string exists in the text file. In case the barcode string does not exist within the text file, the user will be prompted to enter the brewery information into the app and a Java class file will write the data to the text file. On

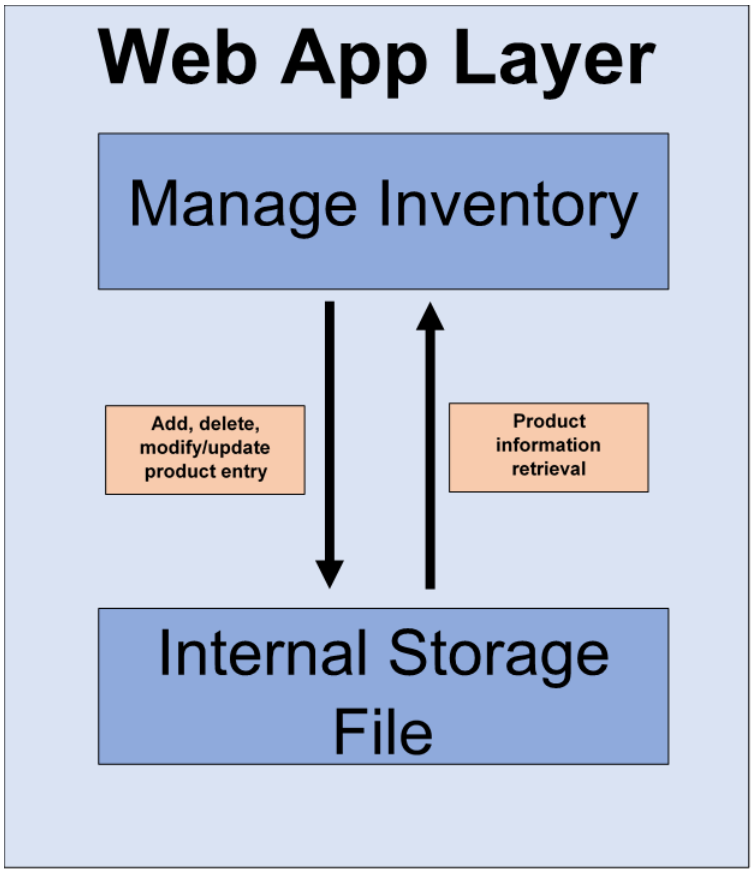


Figure 2: Example subsystem description diagram

the other hand, if the barcode string does exist in the text file, a Java class file will use the linear search algorithm to find the record within the text file and return the brewery product information to the user. All of the data processing for the Swift Scan app will be done with Java class files and the Android API will display the data to the user through the GUI.

4 Y Layer Subsystems

In this section, the layer is described in terms of the hardware and software design. Specific implementation details, such as hardware components, programming languages, software dependencies, operating systems, etc. should be discussed. Any unnecessary items can be ommitted (for example, a pure software module without any specific hardware should not include a hardware subsection). The organization, titles, and content of the sections below can be modified as necessary for the project. Layer Y is mobile application layer. This layer includes mobile app featuring barcode scanner, and internal and private database to keep track of inventory of product. The hardware component in this layer is android phone. We need android phone featuring camera on both sides to scan and read the beer. Generally, we are making our android app in android studio, and we are u. sing java as programming language. The app consists of user interface, internal database zxing library to read the barcode and procced to look into created when app is launched for the first time.

4.1 Layer Hardware

Our beer inventory app is mainly focused software rather than hardware. It consists of mobile app featuring barcode scanner connected with internal database to keep track of inventory product. In this layer hardware component is android phone. As to scan the beer and keep track of beer on the basis of their location, type, style and name. All of the codes and programs running in the app act as software.

4.2 Layer Operating System

The operating system required by layer is user interface and android os. In this layer, User interact with the mobile app features, its content and function. Similarly, in this user interact with app to scan the item with barcode placed on the basis of name, type, style and location.

4.3 Layer Software Dependencies

Our app is software based. We are using android studio for making beer app. The programming language we are using is java. Library used in mobile app is zxing library.

4.4 Subsystem 1

The subsystem of mobile app layer is barcode scanner. This subsystem scans the barcode number and get the product information like type, price, style and date from the database with respective private and public connectors. Barcode reader is connected with public database connector through barcode number. It performs to get the information of beer based on its types, origin, price, style and date. It extracts the information by its barcode number available in database.

4.4.1 Subsystem Hardware

There is no hardware used in this sub system. There is barcode reader that acts with brewing product to get brewing number and all the information.

4.4.2 Subsystem Operating System

The operating system act on this subsystem is user interface. User interact with app to scan the item with barcode placed on the basis of name, type, style and location.

4.4.3 Subsystem Software Dependencies

There are many libraries used while making the android app. Library used in mobile app is zxing library.

4.4.4 Subsystem Programming Languages

We are using android studio for making beer app. The programming language used in this subsystem is java.

4.4.5 Subsystem Data Structures

In this subsystem, when barcode reader scans the beer items; the information/data of beer is stored in internal database.

4.4.6 Subsystem Data Processing

In this subsystem, the algorithm used are insertion, counting sort and search algorithm. When barcode scan the beer, it helps to find the quantity , and search om the basis of type , style and name.

4.5 Subsystem 2

Another subsystem of mobile app is private database connectors. The app scans a bar-code and proceeds to look for the number in the internal database, if the product has already been registered then the app will present the available information to the user but in case that the product is not found in the database the app will allow the user to add the product to the inventory. In this subsystem, the manually added information is stored in the app database internally. All information is stored in private database as we are adding all the information manually

4.5.1 Subsystem Hardware

Our app mainly focused on software. There is no hardware used in this subsystem. However, there is internal database storage when the items are added manually in the beer app.

4.5.2 Subsystem Operating System

The operating system act on this layer is memory management or storage

4.5.3 Subsystem Software Dependencies

There are many libraries used while making the android app. Library used in mobile app is zxing library.

4.5.4 Subsystem Programming Languages

We are using android studio for making beer app. The programming language used in this subsystem is java.We are using local database(sQlite) to add beer manually and to store the information.

4.5.5 Subsystem Data Structures

In this subsystem, all the information and data are stored in local database. They are added manually in the app and after adding it stored internally in app database.

4.5.6 Subsystem Data Processing

In this subsystem, the algorithm used are search algorithm. It helps to search the information from the database. It also used insertion algorithm ; adding information manually.

5 Z Layer Subsystems

5.1 Layer Hardware

Swift Scan app will no longer use a Web Layer to pull and store brewery information records. The Swift Scan app will pull and store information from an internal text file that'll be created once the user decides to manually add a brewery record or search for a specific brewery product.

5.2 Layer Operating System

Swift Scan will execute on Android OS smartphones.

5.3 Layer Software Dependencies

Swift Scan will use numerous libraries from the Android API and also several from the Java SDK. Due to the features that the GUI will provide, the Android API libraries will help accomplish this task and most of the back-end development will be accomplished with libraries from the Java SDK.

5.4 Subsystem 1

The Smart scan app will access the database via database connector and will be able to request the required information or make changes. The databse will send the requested output as required or will allow the user to input new data for the brewery product.

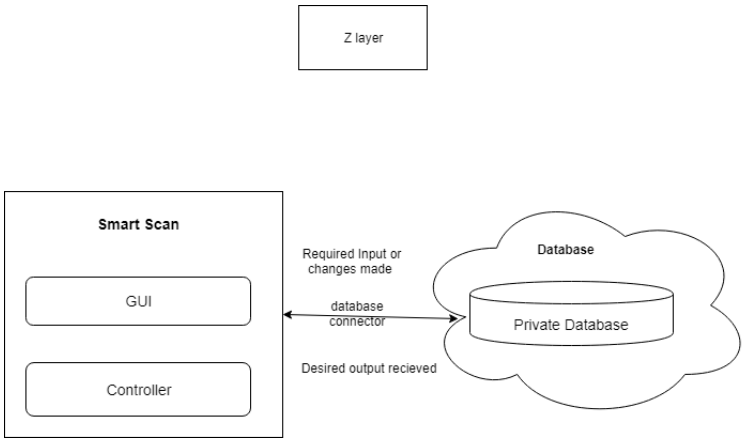


Figure 3: Example subsystem description diagram

5.4.1 Subsystem Hardware

Smart scan will use the devise storage.

5.4.2 Subsystem Operating System

A description of any operating systems required by the subsystem.

5.4.3 Subsystem Software Dependencies

Any of the class files for Swift Scan will use Android's API along with Java SDK libraries for development of the GUI and back-end development of the functionality of the app, respectively.

5.4.4 Subsystem Programming Languages

The main programming language used for the Swift Scan app is Java and using Android's API for the GUI development.

5.4.5 Subsystem Data Structures

ASwift Scan will contain two classes to write data to the internal text file and the second class will be used to retrieve the data and display the data to the user. The second class where the data will be retrieved from the text file will have a method to sort the data as well so it can be displayed to the user. In addition, the second class will also contain a method for updating any records within the text file, depending on what the user decides to update.

5.4.6 Subsystem Data Processing

rder for the brewery data to be recorded in the text file, the user will need to scan first the barcode of the product. The app will attempt to search for the brewery record in the internal text file and there will be a class that'll perform a linear search through the text file to see if the barcode string exists in the text file. In case the barcode string does not exist within the text file, the user will be prompted to enter the brewery information into the app and a Java class file will write the data to the text file. On the other hand, if the barcode string does exist in the text file, a Java class file will use the linear search algorithm to find the record within the text file and return the brewery product information to the user. All of the data processing for the Swift Scan app will be done with Java class files and the Android API will display the data to the user through the GUI.

6 APPENDIX A

Include any additional documents (CAD design, circuit schematics, etc) as an appendix as necessary.

References