**Software Project Documentation**

MyLittleShop

Team:

1. Nguyen Hai Duc
2. Hoang
3. Le Hoang Quan
4. Nguyen The Viet (Team leader) – 9990

**Prepared by:**

Nguyen The Viet

**Table of contents**

1. **Introduction**
   1. **Overview**

MyLittleShop is a desktop application available for PC users. In our project, we assume that each shop uses a laptop and has internet connection to use the application and there are 3 available shops. The application provides access to the available web services (server side) via **Simple Object Access Protocol (SOAP).** This app will allow employee (client) to perform barcode scanning, get the data of the products in the database for the check-out process, and manager (client) to get the view of the database of all shops (including the view of each shop) in order to perform re-stock process and manage the financial lists.

This document provides information of the requirements of the project, the architecture of the application, the schedule (tasks that each member has performed). Additionally, project constraints (risks and changes) are also stated as the last part of the documentation.

**1.2. Scope**

The application provides a role-based system which allows user to log in and access to different functions:

1. As Client: User is able to use the scanner to scan the barcode and retrieve the data of the products
2. As Manager: User is able to send request to retrieve sale report and data of all the shops (view of database). He/she can also send request to re-stock items of a particular shop if possible.
3. **Requirements**
   1. **Functional requirements**
      1. **Use case 1:**

Description: **Login System**

Actor: client (employee & manager)

Basic Path:

1. Client opens the application on the laptop desktop.
2. Client inputs data (username and password) and chooses a role (Employee/Manager).
3. System sends request to the server side to validate authentication.
4. System prompts success, client can access to the allowed functions: **barcode scanning and generating** for Employee, **managing views, updating database (restock) and generating sale report** for Manager.
5. Client clicks the button “log out”.
6. System exits.

Alternate Path:

4. System prompts failure, client re-enters the username and password again  
 5. Back to step 3 of the basic path.

**2.2.2. Use case 2:**

Description: **Barcode Scanner**

Actor: Client (Employee)

Basic Path: (assume user has successfully logged in)

1. Employee clicks “Scan” to start the process.
2. Camera stream appears, the barcode is scanned through the stream (128-Barcode).
3. Once a barcode is scanned, the data will be stored and appear on the GUI, the camera stream will stop.
4. Employee can go back to step 1 and perform scan again, otherwise go to step 5.
5. Employee clicks “Submit”.
6. The server receives the request, returns the check-out list and updates database.
7. Employee clicks “Logout”.
8. System exits.

**2.2.3. Use case 3:**

Description: **Barcode Generator**

Actor: Client (Manager)

Basic Path: (assume user has successfully logged in)

1. Manager inputs code (string) to be generated
2. Manager clicks “GenerateBarcode”
3. An image of barcode (.jpg) is generated
4. Manager clicks “Logout”
5. System exits

**2.2.4. Use case 4:**

Description: **Manager View**

Actor: Client (Manager)

Basic Path:

1. User successfully logs in as Manager.
2. Server returns the view of the product database of all shops. .
3. Manager clicks “Logout”.
4. System exits.

**2.2.5. Use case 5:**

Description: **Restock Item(s)**

Actor: Client (Manager)

Basic Path:

1. User successfully logs in as Manager
2. Server returns the view of the product database of all shops.
3. Manager changes (restock) the quantity of a product in a particular shop by taking items from other shops if available.
4. Manager clicks “Submit”.
5. Server receives request, update the database and return message.
6. Manager clicks “Logout”
7. System exits.

**2.2.6. Use case 6:**

Description: **Sale Report Generating**

Actor: Client (Manager)

Basic Path:

1. User successfully logs in as Manager
2. Server returns the view of the product database of all shops
3. Manager clicks “Sale Report”
4. Server returns list of products that have been checked out (name, price and timestamp)
5. Manager clicks “Logout”
6. System exits.

**2.2 Non-functional requirements**

**2.2.1 Security**

All data of the shops is stored in the database, which can only be accessed via back-end of the server side. Users (client side) only know and are able to access information through available web services in the server side, not directly from the database.

Authentication: application provides login system (role-based) for users to access to the services. Additionally, data of users (username and password) is stored in the database as **hash value** (SHA256)

**2.2.2 Cost of server**

Demo of application will run on local server for cost-saving purpose. Additionally, hiring server and buying assisting services in the future should not cost no more than $5 each month.

**3. Architecture**

**Server**

**Addition**

**4. Schedule**

**5. Risks**

**Authentication:** It is impossible to recover forgotten password because data is stored as hash value inside database. For future implementation, a function to allow user to reset password can be built to fix this problem.

**6. Tasks**

**7. Changes**