**MINISTRY OF EDUCATION AND TRAINING**

**FPT UNIVERSITY**

Capstone Project Document

**Design and Implement**

**The model of Handheld POS**

**(point of sale) system for retailers**

|  |  |
| --- | --- |
| **Group 2** | |
| **Group members** | Nguyễn Thanh Hải – HaiNTSE90195  Đặng Minh Hiếu – HieuDMSE60969  Lê Long Hồ - HoLLSE61291  Huỳnh Hữu Nghị - NghiHHSE61055 |
| **Supervisor** | Bùi Đại Trí - TriBD |
| **Ext. Supervisor** | N/A |
| **Capstone Project code** | HHPOS |

-Ho Chi Minh City, ***1st Sep 2016*** *-*

*This page is intentionally left blank*

# Table of Contents

[Table of Contents 4](#_Toc464199788)

[A. Report No. 1 Introduction 7](#_Toc464199789)

[1. Project Information 7](#_Toc464199790)

[2. Introduction 7](#_Toc464199791)

[3. Current Situation 7](#_Toc464199792)

[4. Problem Definition 9](#_Toc464199793)

[5. Proposed Solution 10](#_Toc464199794)

[5.1 Feature functions 10](#_Toc464199795)

[5.2 Advantages and disadvantages 10](#_Toc464199796)

[6. Functional Requirements 10](#_Toc464199797)

[7. Role and Responsibility 10](#_Toc464199798)

[B. Report No. 2 Project Management Plan 12](#_Toc464199799)

[1. Project Information 12](#_Toc464199800)

[1.1 Name of this Capstone Project 12](#_Toc464199801)

[1.2 Problem Abstract 12](#_Toc464199802)

[1.3 Project Overview 12](#_Toc464199803)

[1.3.1. Curent Situation 12](#_Toc464199811)

[1.3.2. The Proposed System 12](#_Toc464199812)

[1.3.3. Boundaries of the System 13](#_Toc464199813)

[1.3.4. Future Plans 15](#_Toc464199814)

[1.3.5. Development Enviroment 15](#_Toc464199815)

[2. Project Organization 15](#_Toc464199816)

[2.1 Software Process Model 15](#_Toc464199817)

[2.2 Role and responsibilities 16](#_Toc464199818)

[2.3 Tools and Techniques 17](#_Toc464199819)

[3. Project Management Plan 18](#_Toc464199820)

[3.1 Software development life cycle 18](#_Toc464199821)

[3.2 Phase Detail 18](#_Toc464199822)

[3.3 All Meeting Minutes 18](#_Toc464199823)

[4. Coding Convention 18](#_Toc464199824)

[C. Report No. 3 Software Requirement Specificaton 21](#_Toc464199825)

[1. User Requirement Specification 21](#_Toc464199826)

[1.1 Shop Assistant Requirement 21](#_Toc464199827)

[1.2 Manager Requirement 21](#_Toc464199828)

[2. System Requirement Specification 21](#_Toc464199829)

[2.1 External Interface Requirement 22](#_Toc464199830)

[2.1.1. User Interface 22](#_Toc464199834)

[2.1.2. Hardware Interface 25](#_Toc464199835)

[2.1.3. Software Interface 28](#_Toc464199836)

[2.2 System Overview Use Case 29](#_Toc464199837)

[2.3 List of Use Case 29](#_Toc464199838)

[2.3.1. <Shop Assistant, Manager> Login 29](#_Toc464199844)

[2.3.2. < Shop Assistant, Manager > Logout 32](#_Toc464199845)

[2.3.3. < Shop Assistant, Manager > Input Barcode 34](#_Toc464199846)

[2.3.4. < Shop Assistant, Manager > Scan Barcode 36](#_Toc464199847)

[D. System Implementation & Test 79](#_Toc464199848)

[4.1 Introduction 79](#_Toc464199849)

[4.1.1 Overview 79](#_Toc464199850)

[4.1.2 Test Approach 79](#_Toc464199851)

[4.2 Database Relationship Diagram 79](#_Toc464199852)

[4.2.1 Physiscal Diagram 79](#_Toc464199853)

**Definitions, Acronyms, and Abbreviations**

|  |  |
| --- | --- |
| **Name** | **Definition** |
| HHPOS | Handheld POS (point of sale) system for retailers |

# Report No. 1 Introduction

## Project Information

* Project name: **Design and Implement The model of Handheld POS (point of sale) system for retailers.**
* Project Code: **HHPOS**
* Product Type: **Device**
* Start Date: **01/09/2016**

## Introduction

Whether you are in a grocery store, at a craft fair, or at the mall, no one wants to deal with slow point of sale (POS) systems. During busy or hectic times, slow, lagging computers may actually deter sales and frustrate both employees and customers. For the benefit of not only your workforce, but also the customer service experience you provide, portable, handheld POS systems may be the solution to slower, less effective systems.

Business owners also benefit with these types of systems, as they lend themselves to multitasking and help to complete necessary tasks that are essential for the business to function optimally. With an effective handheld POS system, you can analyze sales data, manage inventory, and authorize sales on the fly.

This document describes our working process in 4 months includes our perspective in the system, component designs and detailed core workflows. We developed a good solution that provides a user convinient application for retailer.

## Current Situation

Handheld POS system is necessary for any business store so today there are many system being introduced to solve this problem such as:

+ **Handheld pos terminal EPOS-6780:** An handheld POS device with elegant and slim design, works well for a store, but they aren’t many function to manager product.





+ **Accupos - Android Handheld POS:** start developing POS systems from 2006. A comptele system include: an Android PDA with Magnetic Card Reader or an Android PDA with Barcode Scanner and an Android Wireless Desktop POS Terminal.



## Problem Definition

According Searching and reading documents of current situations, we give some disadvantage of those above systems if applied.

+ **Handheld pos terminal EPOS-6780:** working well at a grocery store or at craft fair, they can’t can analyze sales data, manage inventory, or authorize sales on the fly.

+ **Accupos - Android Handheld POS:** is a good solution, but they have high initial expenses.

## Proposed Solution

In Vietnam, they are prefer working with a large system like Accupos at a mall or chain stores, terminal POS for a grocery store or a craft fair. Our proposed solution is to design a product combined the convenient of terminal POS with the power of an POS system.

### **Feature functions**

* Full integration with accounting software.
* Rocket fuel for your company’s efficiency.
* Friendly for the user, and friendlier for the customer.

### Advantages and disadvantages

The advantages and disadvantages of the proposed solution:

Advantages:

- A compact with full integration.

- Expansion function can scan more than 1 bar code at a time.

- Friendly GUI for customer with two languages.

- The system has low initial cost.

Disadvantages:

- Depended on hardware and external lighting conditions.

- Sometimes this system does not work right.

## Functional Requirements

Function requirements of the system are listed as below:

* Input: imported products to database.
* Output: exporting product.
* Sale manager: Scan barcode, printing invoices.

## Role and Responsibility

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| No | Full Name | Role | Position | Contact |
| 1 | Bùi Đại Trí | Project Manager | Supervisor | Daitribk@gmail.com |
| 2 | Nguyễn Thanh Hải | Developer, Tester | Leader | HaiNTSE90195@fpt.edu.vn |
| 3 | Đặng Minh Hiếu | Developer, Tester | Member | HieuDMSE60969 |
| 4 | Lê Long Hồ | Developer, Tester | Member | HoLLSE61291@fpt.edu.vn |
| 5 | Huỳnh Hữu Nghị | Developer, Tester | Member | NghiHHSE61055@fpt.edu.vn |

Table 1: Roles and Responsibilities

# Report No. 2 Project Management Plan

## Project Information

### **Name of this Capstone Project**

* Official name: **Handheld POS** (point of sale) system for retailers.
* Vietnamese name: **Thiết bị bán hàng cầm tay cho siêu thị nhỏ.**
* Abbreviation: **HHPOS**.
* Product Type: **Device**.

### Problem Abstract

As current in Viet Nam customer often use Handheld pos terminal for a grocery store or at a craft fair, … Using a Handheld POS terminal can not manager the input, output so they don’t know about the inventory.

We provide a solution can do all that thing just in one device with low initial costs. With our device you can scan barcode, import, export product, save your data to database and using it to analyze sale data, manage inventory, and authorize sales on the fly. In addition, you can expand your store into chain stores.

### Project Overview



### Curent Situation

Below are the problems encountered in this project:

//

### The Proposed System

According to the technology researches, we found that **Raspberry pi 3, Zbar** and **Opencv** are very capable of resolve the current situations.

We create a database with MySQL to store and update the good information such as price, name... and transaction data.

Build the program to manage the store and good status. Image processing using **Opencv** libraries and Use **ZBar** libraries to decode barcodes.

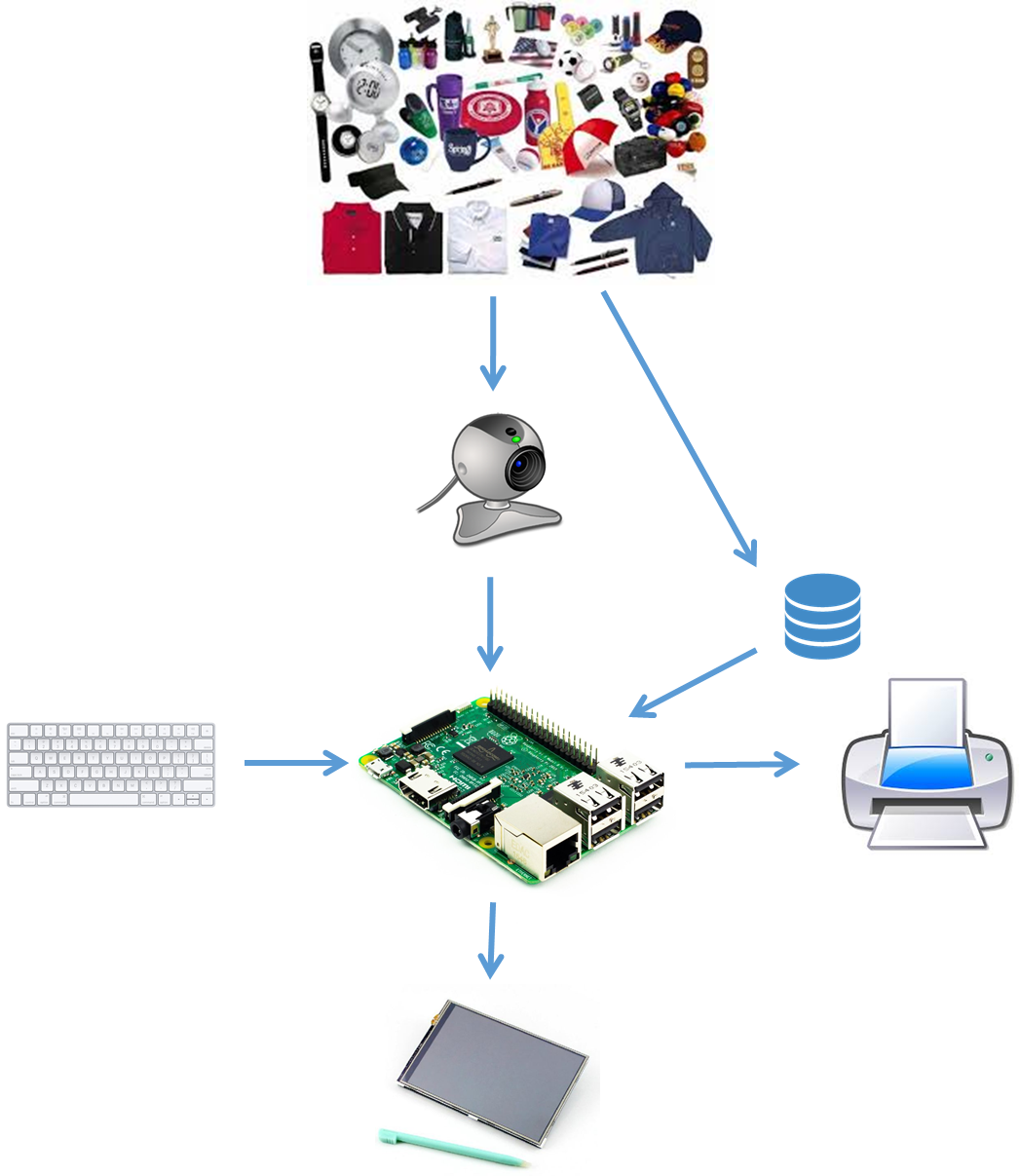
Create the device GUI (graphic user interface) to display the purchase information on screen with QT framework.

Build power supply circuits for the system. Send the receipt to printer via the Wi-Fi or Bluetooth.

Build main program to control all sub-function.

Our system includes two main subsystems: adevice to read barcode and transfer information to web-server, a website for manager goods and user.

### Boundaries of the System

****

**Figure 01. HPOS system’s component**

This system includes:

* Raspberry Pi 3 with an embedded program running in QT framework can: (1) scan barcode using Opencv and ZBar library, (2) manage the store and goods status, (3) send the receipt to printer via Wifi (or Bluetooth).
* A database with MySQL to store and update the goods information.

### Future Plans

Current system only can deploy to a small store. We will provide a web-server to save more data such as goods status, customer information and a program to analyse this data to develop this system can be deploying in a large chains store.

### Development Enviroment

* + - 1. **Hardware requirements**

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

**Table 01. Hardware requirements**

* + - 1. **Software requirements**

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

**Table 02. Software requirements**

## Project Organization

### **Software Process Model**

This project is developed under waterfall model. We apply customized waterfall model to capable with current situation in our team. We choose this model because the following reasons:

* //



**Figure 02. Waterfall model**

*Reference: Page 30, chapter 2, Software process model, SOFTWARE ENGINEERING 9th Edition, by Ian Sommerville.*

### Role and responsibilities

|  |  |  |  |
| --- | --- | --- | --- |
| No | Fullname | Role in group | Responsibilities |
| 1 | Bùi Đại Trí | - Supervisor  - Project manager | - Defining business  - Supporting in technique issues  - Controlling the development process |
| 2 | Nguyễn Thanh Hải | - Team leader  - BA  - Developer  - Tester | - Managing process  - Clarifying requirements  - Preparing documents and reports  - Creating task plan  - Reviewing documents and reports  - Committing all individual works.  - Design and implement hardware.  - Re”Search”ing components, document for implementing.  - Design and implement mobile application.  - Supporting each other.  - Test hardware system. |
| 3 | Đặng Minh Hiếu | - BA  - Developer  - Tester | - Review documents and reports  - Committing all individual works  - ReSearching components, document for implementing.  - Implement chart.  - Test hardware component.  - Test software system. |
| 4 | Lê Long Hồ | - BA  - Developer  - Tester | - Implement document and reports  - Review documents and reports  - Committing all individual works  - ReSearching components, document for implementing.  - Implement chart.  - Test hardware component.  - Test software system. |
| 5 | Huỳnh Hữu Nghị | - BA  - Developer  - Tester | - Implement document and reports  - Review documents and reports  - Committing all individual works  - ReSearching components, document for implementing.  - Implement chart.  - Test hardware component.  - Test software system. |

**Table 03. Role and responsibilities**

### Tools and Techniques

|  |  |  |
| --- | --- | --- |
| No | Tools/Techniques | Name/Version |
| 1 | For Embedded Software |  |
| 2 | For Hardware Design | OrCAD – PCB Editor16.6 |
| 3 | For Webserver |  |
| 4 | For Application |  |
| 5 | For Managing Database | MySQL Workbench 6.2 |
| 6 | For Managing documents, reports, models | Microsoft Office 2013  Microsoft Visio 2013  StarUML 2.7.0  Software Idea Modeler |

**Table 04. Tools and Techniques**

## Project Management Plan

### **Software development life cycle**

//left blank

### **Phase Detail**

//left blank

### **All Meeting Minutes**

Place at folder “Meeting Minutes”.

## Coding Convention

- C/C++ and Python: Using to develop program on hardware.

- Java: Using to develop Webserver.

- Android: Using to develop mobile application. The coding convention is base on Java.

Summary:

• Naming Conventions:

- Variable name should short yet meaningful. If the name is more than one word, it must be in mixed case, starting word with a lowercase.

- Constants name should be uppercase with all words separated by underscores.

- Methods name should be verbs, in mixed case with the first word lowercase, the first letter of each internal word capitalized.

- Class name should be nouns, in mixed case with the first letter of each internal word capitalized.

• Package and import statements:

- Package statements are the first non-comment line.

- Import statement is after package statement.

• Constants

- Numerical constants should not be coded directly.

• Variable Assignments:

- Avoid assigning several variable to the same value in a single statement.

• Comments:

- Using /\* \*/ and """ """ for block comments.

- Using // and # for line comments.

• Return statements:

- A return statement with a value should not use parentheses.

Using C/C++ coding convention from

https://google.github.io/styleguide/cppguide.html

Using Python coding convention from

https://google.github.io/styleguide/pyguide.html

Using Java coding convention from

http://www.oracle.com/technetwork/java/codeconvtoc-136057.html

Using Android coding convention form

http://source.andoird.com/source/code-style.html

References: Code Conventions for the Java TM Programming Language

# Report No. 3 Software Requirement Specificaton

## User Requirement Specification

User is required to wear the HHPOS and a wireless printer connect to HHPOS systems. They use HHPOS by scanning the barcode of the items in a customer’s cart, providing information about the products to customers and printing a transaction receipt. After check out for customer, the transaction receipt is printed, they can move to a new customer, and begin a new transaction.

### **Shop Assistant Requirement**

Shop assistant can use some functions in the system. To use all functions, shop assistant must login. These are some functions shop assistant can use:

* Login/ Logout.
* View Information.
* Check Out.
* Search.
* Manage Orders.
* Manage Carts.
* Print Receipt.

### **Manager Requirement**

Manager can use all functions in the system. To use all functions, manager must login. These are all functions manager can use:

* Login/ Logout.
* View Information.
* Check Out.
* Search.
* Manage Orders.
* Manage Carts.
* Receipt printing.
* Manage employee.
* Manage Inventory

## System Requirement Specification

### **External Interface Requirement**



### User Interface

- General requirement for ggraphics user interface is the GUI should be simple, clear, intuitive, and easy to use.

- The user application is designed base on QT design and display best on 320x480 pixels – screen size interface use English language in Raspbian application.  
- The interface is divided by role, which will allow users to switch easily between different parts of application.

- Meet all the main functions and easily to identify each of functions.

- Use obvious icon to avoiding misunderstanding.

### Hardware Interface

**2.1.2.1 Raspberry Pi 3 Model B**



Figure: Raspberry Pi B3 Kit

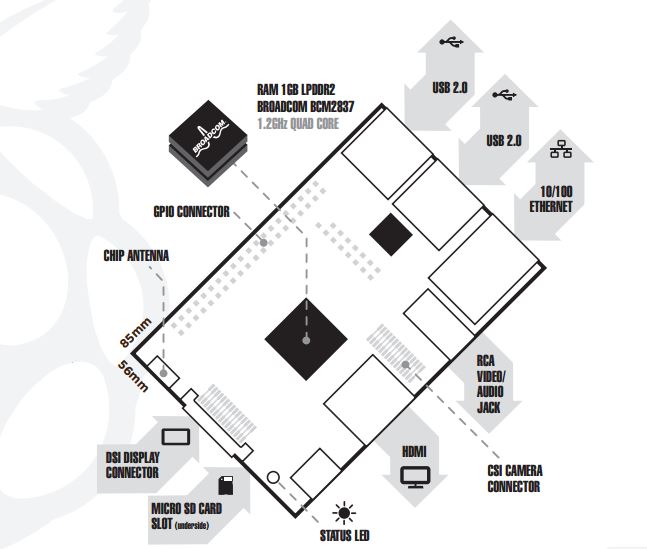


Figure: Raspberry Pi 3 Model B

Overview: The Raspberry Pi 3 Model B is the third generation Raspberry Pi. This powerful credit-card sized single board computer can be used for many applications and supersedes the original Raspberry Pi Model B+ and Raspberry Pi 2 Model B. Whilst maintaining the popular board format the Raspberry Pi 3 Model B brings you a more powerful processer, 10x faster than the first generation Raspberry Pi. Additionally, it adds wireless LAN & Bluetooth connectivity making it the ideal solution for powerful connected designs.

**Detail Specifications:** Here are functions that used in system.

|  |  |
| --- | --- |
| **Specifications** | |
| Processor | + Broadcom BCM2387 chipset.  + 1.2GHz Quad-Core ARM Cortex-A53.  + 802.11 b/g/n Wireless LAN and Bluetooth 4.1 (Bluetooth Classic and LE). |
| GPU | + Dual Core VideoCore IV® Multimedia  Co-Processor. Provides Open GL ES 2.0,  hardware-accelerated OpenVG, and 1080p30  H.264 high-profile decode.  + Capable of 1Gpixel/s, 1.5Gtexel/s or 24GFLOPs   with texture filtering and DMA infrastructure. |
| Memory | + 1GB LPDDR2 |
| Operating System | + Boots from Micro SD card, running a version of the Linux operating system or Windows 10 IoT. |
| Dimensions | + 85 x 56 x 17mm |
| Power | + Micro USB Socket 5.1V, 2.5A |
| Connectors | |
| Ethernet | + 10/100 BaseT Ethernet socket |
| Video Output | + HDMI (rev 1.3 & 1.4)  + Composite RCA (PAL and NTSC) |
| Audio Output | + Audio Output 3.5mm jack, HDMI  + USB 4 x USB 2.0 Connector. |
| GPIO Connector | + 40-pins 2.54 mm (100 mil) expansion header:   2x20 strip.  + Providing 27 GPIO pins as well as +3.3 V, +5 V and GND supply lines. |
| Camera Connector | + 15-pins MIPI Camera Serial Interface (CSI-2) |
| Display Connector | + Display Serial Interface (DSI) 15 ways flat flex cable connector with two data lanes and a clock lane. |
| Memory Card Slot | + Push/pull Micro SDIO |

Table: Raspberry Pi B3 specification

**2.1.2.2 Camera**



The camera specification:

* USB 5.0 Mega Pixel Webcam Camera High Quality.
* Shows the high definition and true color images in network conference and video chat.
* Up to 5 Mega Pixels window capture resolution supported.
* Portable mini size and stylish design.

**2.1.2.3 LCD Touch Screen 3.5 inch**

****

Figure: LCD

The LCD specification:

* + - Power input: 3.3V DC
    - Designed for Raspberry Pi, an ideal alternative solution for HDMI monitor
    - Supports any revision of Raspberry Pi (directly-pluggable)
    - Drivers provided (works with your own Raspbian/Ubuntu directly)
    - 320×480 resolution, better display
    - Convenient Men-Machine interface for Raspberry Pi, combined with the portable power, DIY anywhere anytime
    - Supports Raspbian system, enables your system to:
    - Take photos by touching (up to 17 camera modes)
    - Support software keyboard (system interaction without keyboard/mouse)
    - High quality immersion gold surface plating

**2.1.2.4 Power Supply**

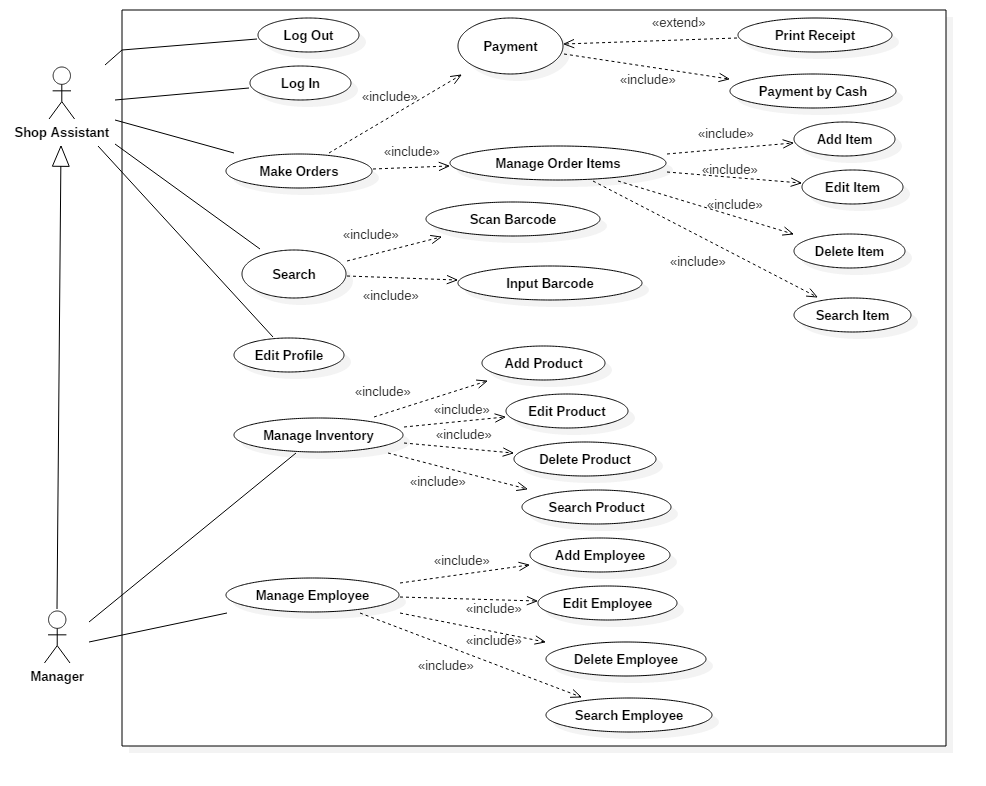
### Software Interface

- Mobile Application: Android OS (v4.0 or above)

- Java Web server: Springs & Hibernate

- Database: MySQL

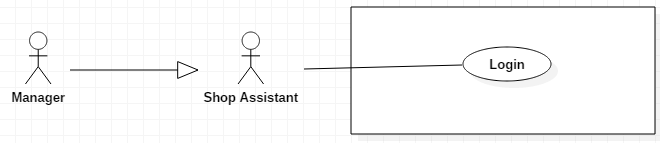
### **System Overview Use Case**



### **List of Use Case**

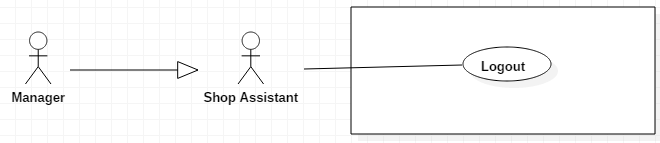


### <Shop Assistant, Manager> Login



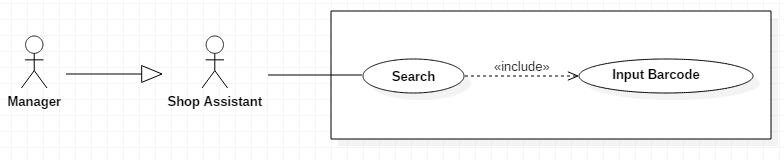
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC01** | | | |
| **Use Case No.** | **UC01** | **Use Case Version** | 1.0 |
| **Use Case Name** | Login | | |
| **Author** | Le Long Ho | | |
| **Date** | 27/9/2016 | **Priority** | Normal |
| **Actor:**  - Shop Assistant  - Manager  **Summary:**  - This use case allows user to log in the system.  **Goal:**  - Users can log in the system.  **Triggers:**  - User sends the login command.  **Pre-conditions:**  - User must have provided account from system.  - Loading is successsful.  **Post-conditions:**  - Success: User login the system. System goes to Menu screen.  - Fail: Show error message.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - User opened Raspberry Pi. | - Loading system display “Loading screen”. - System display a “Login” screen and require identity information from User:  Password: free text input | | 2 | - User input number password. |  | | 3 | - Press “Login” button. | - Check validation of password. - Check login data in database. - User will login system with their specific role.  [Alternative 1] [Alternative 2] [Exception 1]  - LCD will display “Menu” screen |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User enter wrong identity password. | Display error message: “Wrong Password” below Password textbox. | | 2 | User enter invalid password. | Display error message: “Invalid password” below Password textbox. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 |  | System show message. “System is busy” when the system is maintain. |   **Relationships:**  - N/A.  **Business Rules:**  - After login to system, user will be redirected to specific view based on their role on system: manager or customer.   * If role is “Manager”, the system will display to “Manager Menu” screen. * If role is “Shop Assistant”, the system will display to “Shop Assistant Menu” screen. | | | |

### < Shop Assistant, Manager > Logout



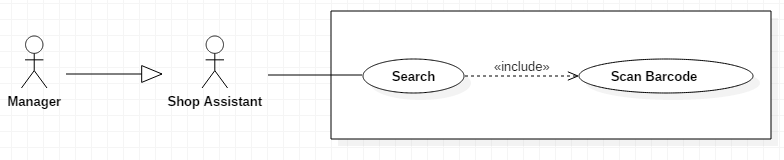
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC02** | | | |
| **Use Case No.** | **UC02** | **Use Case Version** | 1.0 |
| **Use Case Name** | Logout | | |
| **Author** | Le Long Ho | | |
| **Date** | 27/9/2016 | **Priority** | High |
| **Actor:**  - Shop Assistant  - Manager  **Summary:**  - This use case allows user to log out the system. **Goal:** - Users can log out the system.. **Triggers:** - User sends the logout command. **Pre-conditions:** - User have been provided account from system. - Login is successsful.  **Post-conditions:**  - Success: LCD will display “Login” screen. - Fail: Show message error.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - Press “Logout” button on “Menu” screen. | [Exception 1]  - Redirect to “Login” screen. |   **Alternative Scenario:**  - N/A  **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 |  | - If logout failed, system show message:”System is busy”. |   **Relationships:**  - N/A.  **Business Rules:**  - N/A. | | | |

### < Shop Assistant, Manager > Input Barcode



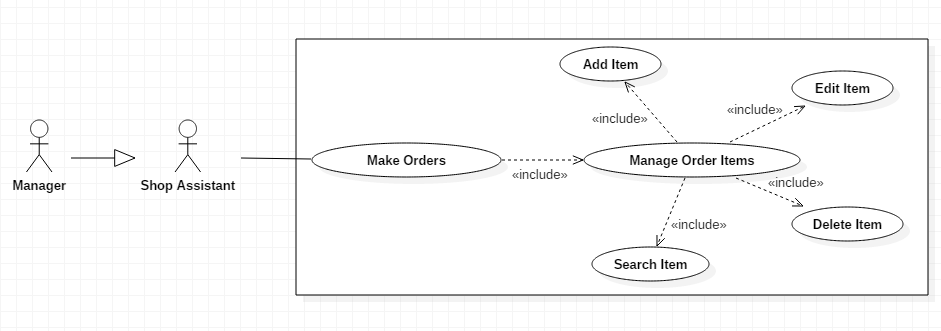
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC03** | | | |
| **Use Case No.** | **UC03** | **Use Case Version** | 1.0 |
| **Use Case Name** | Input Barcode | | |
| **Author** | Le Long Ho | | |
| **Date** | 30/9/2016 | **Priority** | Normal |
| **Actor:**  - Manager  - Shop Assistant  **Summary:**  - This use case allows User to search product by input barcode.  **Goal:**  - User can search by barcode inputting to wiew information or add order.  **Triggers:**  - Input image or number of barcode.  **Preconditions:**  - User goes to “Menu” screen.  **Post Conditions:**  - Success: System will display product information on “Search” screen.  - Fail: Show message error.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - Press “Search” button on “Search” screen. | - LCD will display “Search” screen with text field to input number barcode. | | 2 | - Input number of barcode into text field.  - Press “Search” button.  [Alternative 1] [Alternative 2] [Alternative 3] [Alternative 4] | - Display product information on “Result Search” screen. [Exception1] [Exception2] |   **Alternative:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User press “Add Cart” button. | Add item to Cart.  LCD will display “Search” screen. | | 2 | User press “Menu” button. | LCD will display “Menu” screen. | | 3 | User press “View Cart” button. | LCD will display “View Cart” screen. | | 4 | User press “Back” button. | LCD will display “Search” screen. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 |  | - System show message.”Product is not found” when search barcode is not exist in database. | | 2 | User enter invalid barcode | - System show message. “Invalid barcode”. |   **Relationships:**  - Have <<include>> relationship with “Search” use case.  **Business Rules:**  - Input barcode request will be sent to the system with inputted information.  - A notification will be sent to user after the process is completed.  - After searching user can add product to cart. - In case user can not search by scan barcode, user will use on screen keypad or keyboard to input number of barcode.  . | | | |

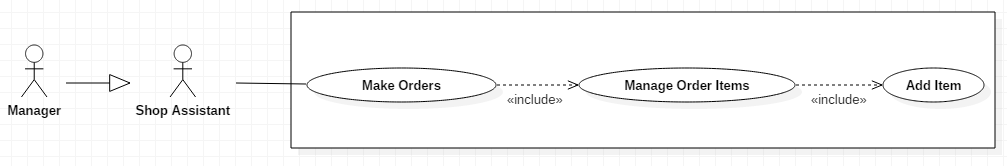
### < Shop Assistant, Manager > Scan Barcode



|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC04** | | | |
| **Use Case No.** | **UC04** | **Use Case Version** | 1.0 |
| **Use Case Name** | Scan Barcode | | |
| **Author** | Le Long Ho | | |
| **Date** | 1/10/2016 | **Priority** | Normal |
| **Actor:**  - Manager  - Shop Assistant  **Summary:**  - This use case allows User to search information by barcode scanning  **Goal:**  - User can use barcode scanning to search information.  **Triggers:**  - Camera read a barcode.  **Preconditions:**  - User goes to “Menu” screen.  **Post Conditions:**  - Success: System will display product information on “Search” screen.  - Fail: Show error message  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - User goes to Menu screen and click “Search” button. | - LCD will display “Search” screen. | | 2 | - Scanning the barcode.  [Alternative 1] [Alternative 2] [Alternative 3] [Alternative 4] | - Search barcode in database.  - Get information of product.  - Display product information on “Result Search” screen. [Exception 1] |   **Alternative:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User press “Add Cart” button. | Add item to Cart.  LCD will display “Search” screen. | | 2 | User press “Menu” button. | LCD will display “Menu” screen. | | 3 | User press “View Cart” button. | LCD will display “View Cart” screen | | 4 | User press “Back” button. | LCD will display “Search” screen. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - If barcode scanning is failed. | - System show message.”Product is not found”. |   **Relationships:**  - Have <<include>> relationship with “Search” use case.  **Business Rules:**  - System update status of the “Search” screen from “Scanning” to “number barcode”. - A notification will be sent to user after the process is completed. - After searching, user can add product to cart. - In case user can not “Search” by scan barcode, user will use on screen keypad or keyboard to input number of barcode. | | | |

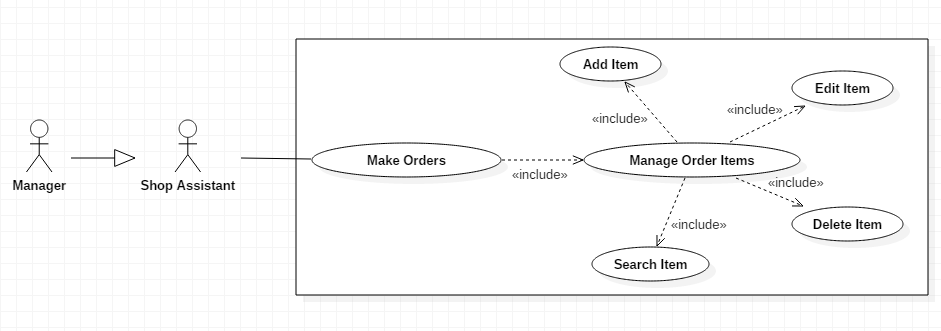
#### < Shop Assistant, Manager> Add Item





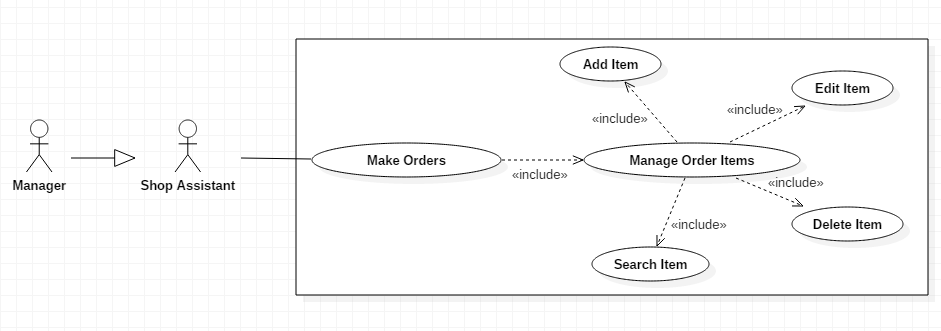
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC05** | | | |
| **Use Case No.** | **UC05** | **Use Case Version** | 1.0 |
| **Use Case Name** | Add Item | | |
| **Author** | Le Long Ho | | |
| **Date** | 3/10/2016 | **Priority** | Normal |
| **Actor:**  - Manager  - Shop Assistant  **Summary:**  - This use case allows User to add new item to cart.  **Goal:** - Add new item after view information.  **Triggers:** - User scan barcode. - User input number barcode.  **Preconditions:** - Have product on “Result Search” screen.  **Post Conditions:** - Success: Add new item to cart and user can view on “View Cart” screen.- Fail: Show error message  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - Input quantity (default quantity =1) |  | | 2 | - User press “Add Order” button on “Result Search” screen.    [Alternative 1] [Alternative 2] [Alternative 3] [Alternative 4] | - Added item to Cart.  - Calculate total and tax.  - LCD will display “Search” screen with currently item’s information. [Exception 1] [Exception 2] |   **Alternative:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User press “Add Cart” button. | Add item to Cart.  LCD will display “Search” screen. | | 2 | User press “Menu” button. | LCD will display “Menu” screen. | | 3 | User press “View Cart” button. | LCD will display “View Cart” screen | | 4 | User press “Back” button. | LCD will display “Search” screen. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - If add item failed | System show message to notify user that add item failed and renew request has been aborted. | | 2 | Available quantity less than or equal to zero after edit item. | Display error message: “Not enough product”. |   **Relationships:**  - Have << include>> relationship with “Manage Order Item” use case.  **Business Rules:**  - Total would be calculated. | | | |

#### < Shop Assistant, Manager> Edit Item



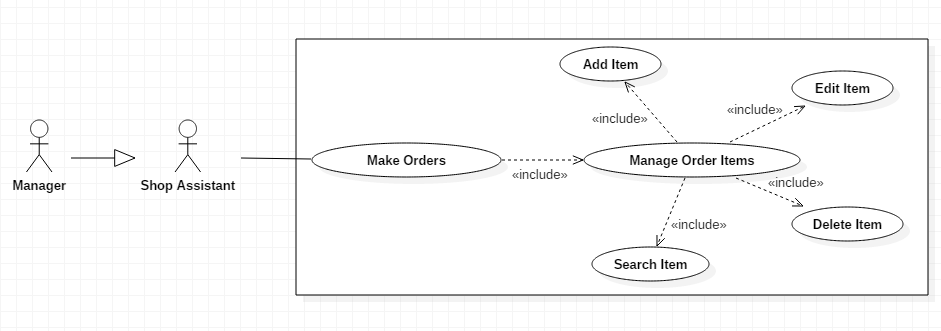
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC06** | | | |
| **Use Case No.** | **UC06** | **Use Case Version** | 1.0 |
| **Use Case Name** | Edit Item | | |
| **Author** | Le Long Ho | | |
| **Date** | 3/10/2016 | **Priority** | Normal |
| **Actor:**  - Manager  - Shop Assistant  **Summary:**  - This use case allows User to edit quantity of item in order list.  **Goal:**  - User can edit quantity of item after add item in order list.  **Triggers:**  - N/A.  **Preconditions:**  - User have available item in order list.  **Post Conditions:**  - Success: Edit quantity of item in order list and user can view on “View Cart” screen.  - Fail: Show error message  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - User click item in list order on “View Cart” screen. | - LCD will display “Input By Virtual Keyboard” screen. | | 2 | - Input new quantity.  - Press “Enter” button.      [Alternative 1] [Alternative 2] [Alternative 3] [Alternative 4] | - Checking inventory quantity.  - Calculate total of order.  - LCD will display “View Cart” screen. [Exception 1] |   **Alternative:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User press “Add Cart” button. | Add item to Cart.  LCD will display “Search” screen. | | 2 | User press “Menu” button. | LCD will display “Menu” screen. | | 3 | User press “View Cart” button. | LCD will display “View Cart” screen. | | 4 | User press “Back” button. | LCD will display “Search” screen. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Available quantity less than or equal to zero after edit item. | Display error message: “Not enough product to add”. |   **Relationships:**  - Have << include >> relationship with “Manage Order Item” use case.  **Business Rules:**  - Total would be calculated. | | | |

#### < Shop Assistant, Manager> Delete Item



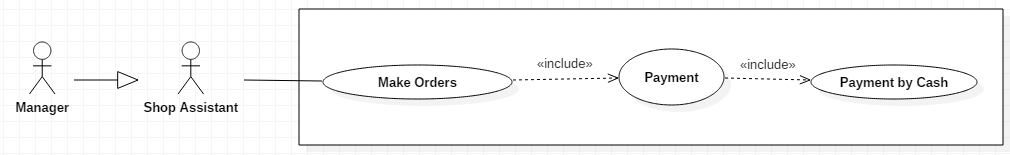
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC07** | | | |
| **Use Case No.** | **UC07** | **Use Case Version** | 1.0 |
| **Use Case Name** | Delete Item | | |
| **Author** | Le Long Ho | | |
| **Date** | 3/10/2016 | **Priority** | Normal |
| **Actor:**  - Manager  - Shop Assistant  **Summary:**  - This use case allows User to delete item in order list.  **Goal:**  - User can delete item in order list.  **Triggers:**  - User press “Delete” button.  **Preconditions:**  - User have available cart.  **Post Conditions:**  - Success: Delete item in order list. User can view on “View Cart” screen after delete item.  - Fail: Show error message  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - User press “Delete Item” button on “View Cart” screen.  [Alternative 1] [Alternative 2] [Alternative 3] [Alternative 4] | - Delete item in order list.  - Calculate total of order.  - LCD will display “View Cart” screen. [Exception 1] |   **Alternative:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User press “Add Cart” button. | Add item to Cart.  LCD will display “Search” screen. | | 2 | User press “Menu” button. | LCD will display “Menu” screen. | | 3 | User press “View Cart” button. | LCD will display “View Cart” screen. | | 4 | User press “Back” button. | LCD will display “Search” screen. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 |  | System show message to notify user that delete item failed. |   **Relationships:**  - Have <<include>> relationship with “Manage Order Item” use case.  **Business Rules:**  - Total and tax would be calculated. | | | |

#### < Shop Assistant, Manager> Search Item



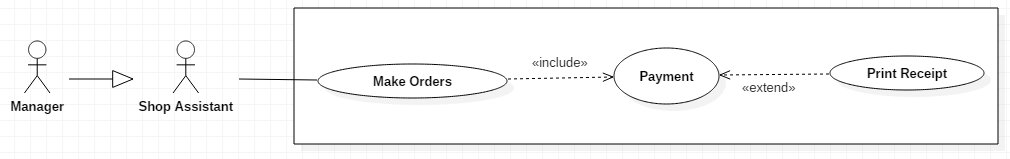
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC08** | | | |
| **Use Case No.** | **UC08** | **Use Case Version** | 1.0 |
| **Use Case Name** | Search Item | | |
| **Author** | Le Long Ho | | |
| **Date** | 3/10/2016 | **Priority** | Normal |
| **Actor:**  - Manager  - Shop Assistant  **Summary:**  - This use case allows User to search item in order list.  **Goal:**  - User can search item in order list.  **Triggers:**  - User press “Search” button.  **Preconditions:**  - User have available product in order list.  **Post Conditions:**  - Success: User can view on “View Cart” screen after search item.  - Fail: Show error message  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - User press “Search Item” button on “View Cart” screen.  [Alternative 1] [Alternative 2] [Alternative 3] [Alternative 4] | - LCD will display “View Cart” screen. [Exception 1] |   **Alternative:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User press “Add Cart” button. | Add item to Cart.  LCD will display “Search” screen. | | 2 | User press “Menu” button. | LCD will display “Menu” screen. | | 3 | User press “View Cart” button. | LCD will display “View Cart” screen. | | 4 | User press “Back” button. | LCD will display “Search” screen. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 |  | - System show message.”Product is not found” when failed search. |   **Relationships:**  - Have <<include>> relationship with “Manage Order Item” use case.  **Business Rules:**  - System update status of the “View Cart” screen from “Searching” to view item. | | | |

* + 1. **<Shop Assistant, Manager > Payment By Cash**



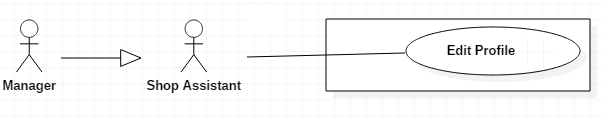
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC03** | | | |
| **Use Case No.** | **UC03** | **Use Case Version** | 1.0 |
| **Use Case Name** | Payment By Cash | | |
| **Author** | Le Long Ho | | |
| **Date** |  | **Priority** | Normal |
| **Actor:**  - Shop Assistant  - Manager  **Summary:**  - This use case allows User to use “Payment By Cash” function.  **Goal:**  - User can input cash and return change to customer.  **Triggers:**  - N/A.  **Preconditions:**  - User goes to “View Cart” screen.  **Post Conditions:**  - Success: System will display total and change on “Payment” screen.  - Fail: Show error message.  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - Press “Payment” button on “View Cart” screen. | - Caculate total and tax.  - Push total to “Payment” screen. | | 2 | - User input number of cash in text field on “Payment” screen.    -Press “Submit” button.  [Alternative 1] [Alternative 2] [Alternative 3] | - Calculate change and return change.  - Push change to “Payment” screen.  - LCD will display “Payment” screen.  -LCD will display “Reciept” screen. [Exception 1] |   **Alternative:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User press “Print” button. | Print receipt by printer. | | 2 | User press “Menu” button. | LCD will display “Menu” screen. | | 3 | User press “Back” button. | LCD will display “View Cart” screen. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | If payment failed | Show message to notify user that payment failed and the renew request has been aborted |   **Relationships:**  - Have <<include>> relationship with “Payment” use case.  **Business Rules:**  - System must ensure has no duplicate customer or vehicle.  - Start date musy not be earlier than the current date. | | | |

* + 1. **<Shop Assistant, Manager> Print Receipt**



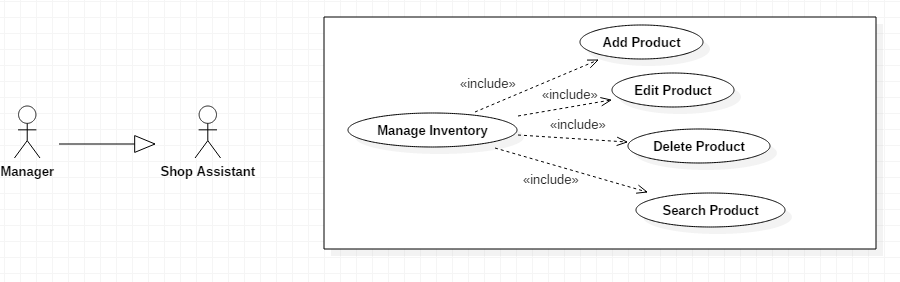
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC03** | | | |
| **Use Case No.** | **UC03** | **Use Case Version** | 1.0 |
| **Use Case Name** | Print Receipt | | |
| **Author** | Le Long Ho | | |
| **Date** |  | **Priority** | Normal |
| **Actor:**  - Shop Assistant  - Manager  **Summary:**  - This use case allows User to use “Print Receipt” function.  **Goal:**  - User can print receipt after payment.  **Triggers:**  - N/A.  **Preconditions:**  - User goes to “Payment” screen.  **Post Conditions:**  - Success: System will display total and change on “Payment” screen.  - Fail: Show error message  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - Press “Print” button on “Payment” screen.    [Alternative 1] [Alternative 2] [Alternative 3] | - Connect Printer.  - Get information on database.  - Print receipt. [Exception 1] |   **Alternative:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User press “Submit” button. | LCD will display “Receipt” screen. | | 2 | User press “Menu” button. | LCD will display “Menu” screen. | | 3 | User press “Back” button. | LCD will display “View Cart” screen. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User press “Print” button. | When the printer is not connect, show message to notify user that receipt printting failed |   **Relationships:**  - Have <<extend>> relationship with “Payment” use case.  **Business Rules:**  - N/A. | | | |

**2.3.11. <Shop Assistant, Manager> Edit Profile**



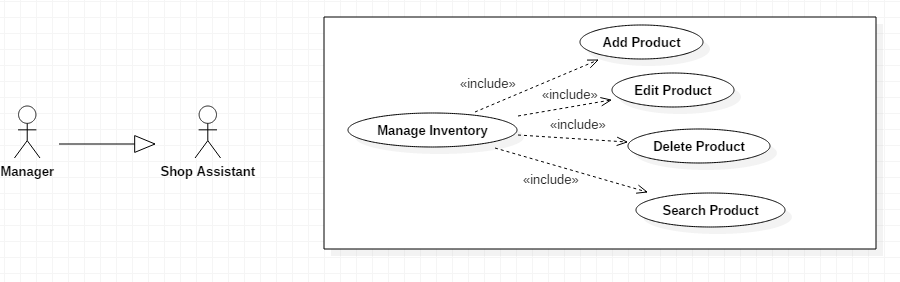
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC03** | | | |
| **Use Case No.** | **UC03** | **Use Case Version** | 1.0 |
| **Use Case Name** | Edit Profile | | |
| **Author** | Le Long Ho | | |
| **Date** |  | **Priority** | Normal |
| **Actor:**  - Shop Assistant  - Manager  **Summary:**  - This use case allows User to use “Edit Profile” function.  **Goal:**  - User can edit profile.  **Triggers:**  - Click to user’s avatar.  **Preconditions:**  - User goes to “Menu” screen.  **Post Conditions:**  - Success: System will display user’s profile information on “Profile” screen.  - Fail: Show error message  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - Click user’s avatar on “Menu” screen. | - LCD will display “Profile” screen. | | 2 | - Input new information. - Press “Save” button.  [Alternative 1] [Alternative 2] [Alternative 3] | - Added new user’s profile to database.  - LCD will display “Profile” screen with information was updated. [Exception 1] |   **Alternative:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User press “Menu” button. | LCD will display “Menu” screen. | | 2 | User press “Back” button. | LCD will display “Menu” screen. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 |  | Show message to notify user that editting profile is failed |   **Relationships:**  - N/A  **Business Rules:**  - N/A. | | | |

#### <Manager> Add Product



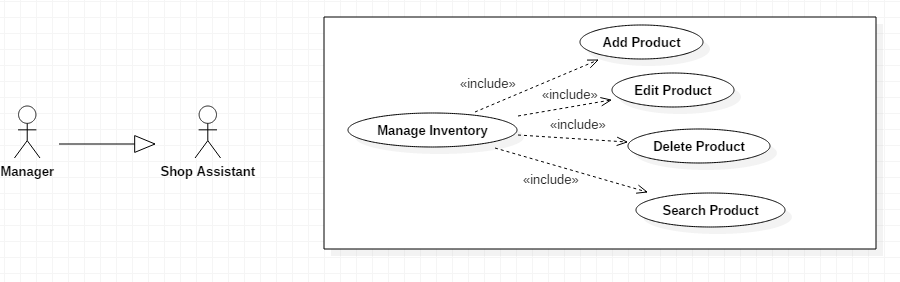
|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC12** | | | |
| **Use Case No.** | **UC12** | **Use Case Version** | 1.0 |
| **Use Case Name** | Add Product | | |
| **Author** | Le Long Ho | | |
| **Date** | 3/10/2016 | **Priority** | Normal |
| **Actor:**  - Manager  **Summary:**  - This use case allows User to add new product to inventory.  **Goal:**  - Add new product. - In case user can not add product by scan barcode, user will use on screen keypad or keyboard to input number of barcode to add product.  **Triggers:**  - User is manager. - User scan barcode. - User input number barcode.  **Preconditions:**  - User goes to “Manage Inventory” screen.  **Post Conditions:**  - Success: Add new product to database.  - Fail: Show error message  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - Scan barcode. |  | | 2 | - Input quantity (default quantity =1) |  | | 3 | - User press “Add Order” button on “Result Search” screen.  [Alternative 1] [Alternative 2] [Alternative 3] [Alternative 4] | - Added item to Cart.  - |   **Alternative:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User press “Add Cart” button. | Add item to Cart.  LCD will display “Search” screen. | | 2 | User press “Menu” button. | LCD will display “Menu” screen. | | 3 | User press “View Cart” button. | LCD will display “View Cart” screen | | 4 | User press “Back” button. | LCD will display “Search” screen. |   **Exceptions:**  -N/A  **Relationships:**  - Have << include>> relationship with “Manage Order Item” use case.  **Business Rules:**  - N/A | | | |

#### <Manager> Edit Product



|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC13** | | | |
| **Use Case No.** | **UC13** | **Use Case Version** | 1.0 |
| **Use Case Name** | Edit Product | | |
| **Author** | Le Long Ho | | |
| **Date** | 3/10/2016 | **Priority** | Normal |
| **Actor:**  - Manager  **Summary:**  - This use case allows User to edit product in inventory.  **Goal:**  - User can edit product in inventory.  **Triggers:**  - User is manager.  **Preconditions:**  - User have available item in order list.  **Post Conditions:**  - Success: Edit quantity of item in order list and user can view on “View Cart” screen.  - Fail: Show error message  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - User click item in list order on “View Cart” screen. | - LCD will display “Input By Virtual Keyboard” screen. | | 2 | - Input new quantity.  - Press “Enter” button.  [Alternative 1] [Alternative 2] [Alternative 3] [Alternative 4] | - Checking inventory quantity.  - Calculate total of order.  - LCD will display “View Cart” screen. |   **Alternative:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User press “Add Cart” button. | Add item to Cart.  LCD will display “Search” screen. | | 2 | User press “Menu” button. | LCD will display “Menu” screen. | | 3 | User press “View Cart” button. | LCD will display “View Cart” screen. | | 4 | User press “Back” button. | LCD will display “Search” screen. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Available quantity less than or equal to zero | Display error message: “Not enough product” textbox. |   **Relationships:**  - Have << include >> relationship with “Manage Order Item” use case.  **Business Rules:**  - N/A | | | |

#### <Manager> Delete Product



|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC14** | | | |
| **Use Case No.** | **UC14** | **Use Case Version** | 1.0 |
| **Use Case Name** | Delete Product | | |
| **Author** | Le Long Ho | | |
| **Date** | 3/10/2016 | **Priority** | Normal |
| **Actor:**  - Manager  - Shop Assistant  **Summary:**  - This use case allows User to delete item in order list.  **Goal:**  - User can delete item in order list.  **Triggers:**  - User is manager. - User press “Delete” button.  **Preconditions:**  - User have available cart.  **Post Conditions:**  - Success: Delete item in order list. User can view on “View Cart” screen after delete item.  - Fail: Show error message  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - User press “Delete Item” button on “View Cart” screen.  [Alternative 1] [Alternative 2] [Alternative 3] [Alternative 4] | - Delete item in order list.  - Calculate total of order.  - LCD will display “View Cart” screen. |   **Alternative:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User press “Add Cart” button. | Add item to Cart.  LCD will display “Search” screen. | | 2 | User press “Menu” button. | LCD will display “Menu” screen. | | 3 | User press “View Cart” button. | LCD will display “View Cart” screen. | | 4 | User press “Back” button. | LCD will display “Search” screen. |   **Exceptions:**  -N/A  **Relationships:**  - Have <<include>> relationship with “Manage Order Item” use case.  **Business Rules:**  - N/A | | | |

#### <Manager> Search Product

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC15** | | | |
| **Use Case No.** | **UC15** | **Use Case Version** | 1.0 |
| **Use Case Name** | Search Product | | |
| **Author** | Le Long Ho | | |
| **Date** | 3/10/2016 | **Priority** | Normal |
| **Actor:**  - Manager  **Summary:**  - This use case allows User to search item in order list.  **Goal:**  - User can search item in order list.  **Triggers:**  - User is manager. - User press “Search” button.  **Preconditions:**  - User have available product in order list.  **Post Conditions:**  - Success: User can view on “View Cart” screen after search item.  - Fail: Show error message  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - User press “Search Item” button on “View Cart” screen.  [Alternative 1] [Alternative 2] [Alternative 3] [Alternative 4] | - LCD will display “View Cart” screen. |   **Alternative:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User press “Add Cart” button. | Add item to Cart.  LCD will display “Search” screen. | | 2 | User press “Menu” button. | LCD will display “Menu” screen. | | 3 | User press “View Cart” button. | LCD will display “View Cart” screen. | | 4 | User press “Back” button. | LCD will display “Search” screen. |   **Exceptions:**  -N/A  **Relationships:**  - Have <<include>> relationship with “Manage Order Item” use case.  **Business Rules:**  - N/A | | | |

#### <Manager> Add Employee

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC16** | | | |
| **Use Case No.** | **UC16** | **Use Case Version** | 1.0 |
| **Use Case Name** | Add Employee | | |
| **Author** | Le Long Ho | | |
| **Date** | 3/10/2016 | **Priority** | Normal |
| **Actor:**  - Manager  **Summary:**  - This use case allows User to add new employee to database.  **Goal:**  - Add new employee to database.  **Triggers:**  - User is manager.  **Preconditions:**  - N/A.  **Post Conditions:**  - Success: Add new employee to database. New employee can sign in system.  - Fail: Show error message  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - User press “Manage Employee” button on “Menu” screen. | - LCD will display “Manage Employee” screen. | | 2 | - User press “Add Employee” button. | - LCD will display “Input By Virtual Keyboard” screen. | | 3 | - Input new employee’s information. - User press “Submit” button    [Alternative 1] [Alternative 2] [Alternative 3] | - Added new employee to database.  - LCD will display “Manage Employee” screen. |   **Alternative:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User press “Add Employee” button. | Add employee to database.  LCD will display “Manage Inventory” screen. | | 2 | User press “Menu” button. | LCD will display “Menu” screen. | | 3 | User press “Back” button. | LCD will display “Search” screen. |   **Exceptions:**  -N/A  **Relationships:**  - Have << include>> relationship with “Inventory” use case.  **Business Rules:**  - N/A | | | |

#### <Manager> Edit Employee

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC13** | | | |
| **Use Case No.** | **UC13** | **Use Case Version** | 1.0 |
| **Use Case Name** | Edit Employee | | |
| **Author** | Le Long Ho | | |
| **Date** | 3/10/2016 | **Priority** | Normal |
| **Actor:**  - Manager  **Summary:**  - This use case allows User to edit product in inventory.  **Goal:**  - User can edit product in inventory.  **Triggers:**  - User is manager.  **Preconditions:**  - User have available item in order list.  **Post Conditions:**  - Success: Edit quantity of item in order list and user can view on “View Cart” screen.  - Fail: Show error message  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - User click item in list order on “View Cart” screen. | - LCD will display “Input By Virtual Keyboard” screen. | | 2 | - Input new quantity.  - Press “Enter” button.  [Alternative 1] [Alternative 2] [Alternative 3] [Alternative 4] | - Checking inventory quantity.  - Calculate total of order.  - LCD will display “View Cart” screen. |   **Alternative:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User press “Add Cart” button. | Add item to Cart.  LCD will display “Search” screen. | | 2 | User press “Menu” button. | LCD will display “Menu” screen. | | 3 | User press “View Cart” button. | LCD will display “View Cart” screen. | | 4 | User press “Back” button. | LCD will display “Search” screen. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Available quantity less than or equal to zero | Display error message: “Not enough product” textbox. |   **Relationships:**  - Have << include >> relationship with “Manage Order Item” use case.  **Business Rules:**  - N/A | | | |

#### < Manager> Delete Employee

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC18** | | | |
| **Use Case No.** | **UC18** | **Use Case Version** | 1.0 |
| **Use Case Name** | Delete Employee | | |
| **Author** | Le Long Ho | | |
| **Date** | 3/10/2016 | **Priority** | Normal |
| **Actor:**  - Manager  - Shop Assistant  **Summary:**  - This use case allows User to delete item in order list.  **Goal:**  - User can delete item in order list.  **Triggers:**  - User is manager. - User press “Delete” button.  **Preconditions:**  - User have available cart.  **Post Conditions:**  - Success: Delete item in order list. User can view on “View Cart” screen after delete item.  - Fail: Show error message  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - User press “Delete Item” button on “View Cart” screen.  [Alternative 1] [Alternative 2] [Alternative 3] [Alternative 4] | - Delete item in order list.  - Calculate total of order.  - LCD will display “View Cart” screen. |   **Alternative:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User press “Add Cart” button. | Add item to Cart.  LCD will display “Search” screen. | | 2 | User press “Menu” button. | LCD will display “Menu” screen. | | 3 | User press “View Cart” button. | LCD will display “View Cart” screen. | | 4 | User press “Back” button. | LCD will display “Search” screen. |   **Exceptions:**  -N/A  **Relationships:**  - Have <<include>> relationship with “Manage Order Item” use case.  **Business Rules:**  - N/A | | | |

#### <Manager> Search Employee

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – UC19** | | | |
| **Use Case No.** | **UC19** | **Use Case Version** | 1.0 |
| **Use Case Name** | Search Employee | | |
| **Author** | Le Long Ho | | |
| **Date** | 3/10/2016 | **Priority** | Normal |
| **Actor:**  - Manager  **Summary:**  - This use case allows User to search item in order list.  **Goal:**  - User can search item in order list.  **Triggers:**  - User is manager. - User press “Search” button.  **Preconditions:**  - User have available product in order list.  **Post Conditions:**  - Success: User can view on “View Cart” screen after search item.  - Fail: Show error message  **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | - User press “Search Item” button on “View Cart” screen.  [Alternative 1] [Alternative 2] [Alternative 3] [Alternative 4] | - LCD will display “View Cart” screen. |   **Alternative:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User press “Add Cart” button. | Add item to Cart.  LCD will display “Search” screen. | | 2 | User press “Menu” button. | LCD will display “Menu” screen. | | 3 | User press “View Cart” button. | LCD will display “View Cart” screen. | | 4 | User press “Back” button. | LCD will display “Search” screen. |   **Exceptions:**  -N/A  **Relationships:**  - Have <<include>> relationship with “Manage Order Item” use case.  **Business Rules:**  - N/A | | | |

3. Software System Attribute

3.1. Reliability

* System is expected to run continuously for years without errors (or in some cases recover by themselves if an error occurs).
* System can safely be shut down for repair, or another way to repair without stopping system.
* The system shall never crash or hang, other than as the result of a network error.
* Mean Time Between Failures (MTBF): The acceptable failure is once a year.
* Mean Time To Repair (MTTR): When the failure occurs, it should take at least time as possible to repair. The acceptable mean time for a particular failure must be least than 1 minute.

3.2. Availability

* The server shall be working 24/7.
* When the system goes in under-maintenance, the page or application will display message “System is maintaining at the moment. Please check again later”.

3.3. Security

* Guarantee the data and application protection from being stolen and modified by encoding and decoding data.

3.4. Maintainability

* All code shall be fully documented. All program files shall include comments concerning authorship and date of last change.
* The code shall be modular to permit future modifications.

3.5. Performance

* This is the system’s performance characteristics:

• Capacity: 1 end user a time.

• Response time for a transaction:

• Average: 2 second

• Maximum: 7 seconds.

3.6. Usability

* The system is intuitive and easy to use.
* All the texts, label, and message will be written in English.

D. Report No.4 Software Design Specification

1. Design overview

* This document describes the technical and user interface design of **HHPOS System**. It includes the architectural design, the detailed design of common functions and business functions.
* The architectural design describes the overall architecture of the system and the architecture of each main component and subsystem.
* The detailed design describes static and dynamic structure for each component and functions. It includes class diagrams, class explanations and sequence diagrams for each use cases.
* Document overview:
* Section 2: gives an overall description of the system architecture design.
* Section 3: gives component diagrams that describe the connection and integration of the system.
* Section 4: gives the detail design description which includes class diagram, class explanation, and sequence diagram to details the application functions.
* Section 5: describe screens design.
* Section 6: describe a fully attributed ERD.
* Section 7: describe algorithms.

2. System Architectural Design

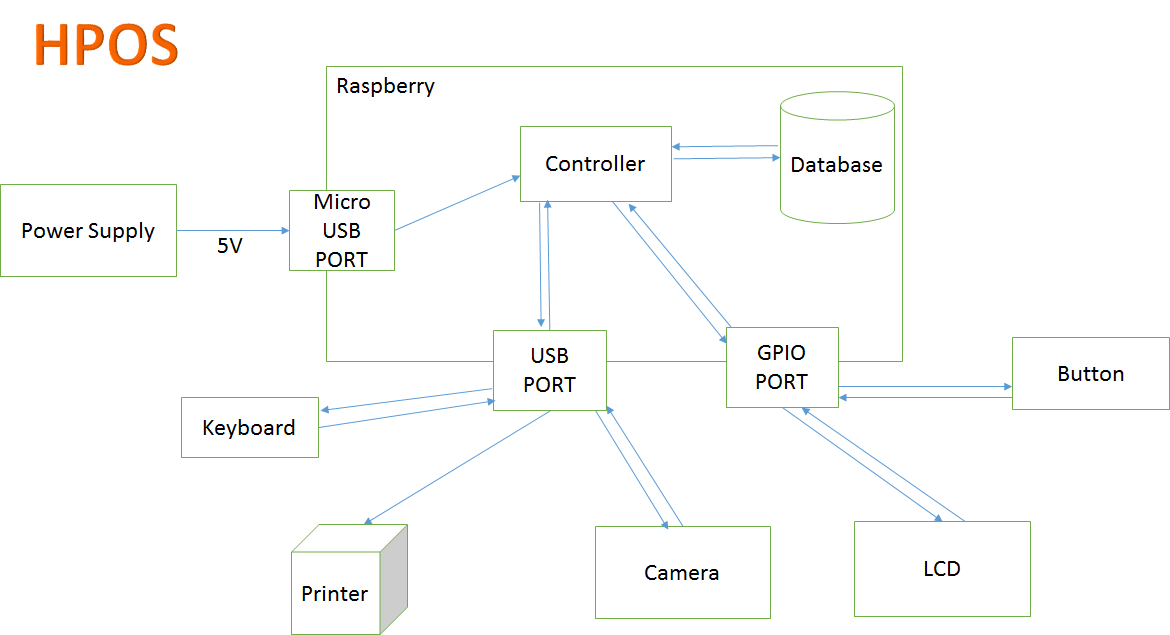


Figure: System overview architecture

2.1 External Interface Requirement

2.1.1. User interface

User interface use Graphical User Interface must be simple, clear and easy to use.

2.1.2. Hardware interface

Raspberry Pi B3 with SDRAM 1GB, Quad-core, 1.2GHz ARMv8 CPU

Camera 5.0 megapixcel.

LCD touch screen 3.5 inch.

SD card Sandisk 16GB class 10.

Power Supply 5V.

3. Component diagram

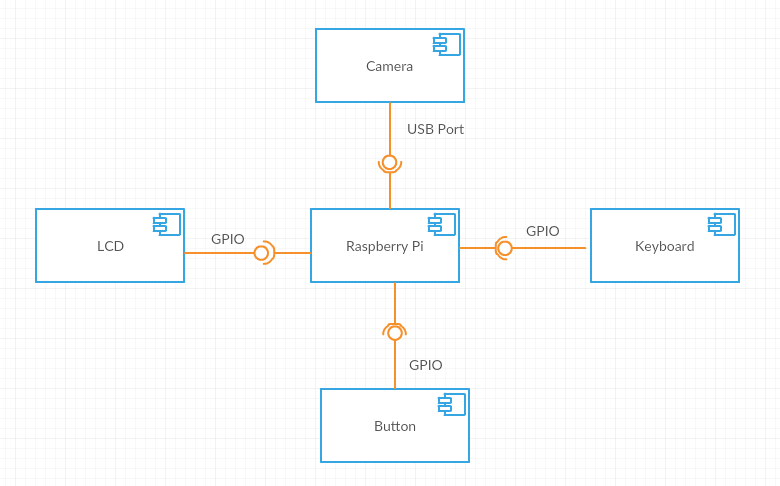


Figure: Component diagram

4. Detail description component

4.1. Hardware layer

4.1.1. Raspberry Pi B3

4.2. Software layer

Raspbian is an unofficial port of Debian Wheezy armhf with compilation settings adjusted to produce optimized "hard float" code that will run on the Raspberry Pi. This provides significantly faster performance for applications that make heavy use of floating point arithmetic operations. All other applications will also gain some performance through the use of advanced instructions of the ARMv6 CPU in Raspberry Pi.

Although Raspbian is primarily the efforts of Mike Thompson (mpthompson) and Peter Green (plugwash), it has also benefited greatly from the enthusiastic support of Raspberry Pi community members who wish to get the maximum performance from their device.

5. Detail diagram

5.1. Class diagram

5.2. Class diagram explanation

|  |  |
| --- | --- |
| **Class dictionary: Describe Class** | |
| **Class name** | **Description** |
| Gsm | Describe all information and methods relative to |

Table: Class diagram explanation

5.2.1. Gsm

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| getUart\_filestream | Int | Public |  |
| setUart\_filestram | Void | Public |  |
| Init | Void | Public |  |
| sendSMS | Void | Public |  |

Table: Method of Gsm class

5.3. Flowchart diagram

6. User interface design

6.1. A

**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Menu |  | Yes | No | Menu | N/A | N/A |
| 2 | Instruction |  | Yes | No | Label | N/A | N/A |

Table: Field of “ ” screen

6.2. B

**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Menu |  | Yes | No | Menu | N/A | N/A |
| 2 | Instruction |  | Yes | No | Label | N/A | N/A |

Table: Field of “ ” screen

7. Algorithm

OpenCV (Open Source Computer Vision) is a library started by Intel in 1999. It focus on realtime image processing.

# System Implementation & Test

## Introduction

### Overview

In this section, contains information about the test plan, test approach, test process, test environment, test pass/fail criteria, checklist to check test this system.

### Test Approach

**White box testing**: Developers self-test code that they wrote and fix themselves

**Black box testing:** Test on each function of each module in system to ensure each module execute the right function. Then integration testing to test the function in system when integrate. Finally, system test to test the whole system.

**Goal:** Discover bug in system 🡪 fix bug 🡪 regression test 🡪 completed system.

## Database Relationship Diagram

* 1. **Physical Diagram**
  2. **Data Dictionary**

|  |  |
| --- | --- |
| **Data dictionary: descripbe content of all tables** | |
| **Table Name** | **Description** |
| Role | Describe user’s role in the system |
| User | Describe all users profiles in the system |
| History | Describe all history in the system |
| ActionType | Describe all action type of user in the system |
| Picture | Describe all pictures in the system |
| Invoice | Describe all invoice in the system |
| InvoiceDetail | Describe all detail of invoice in the system |
| Barcode | Describe all product barcode in the system |
| Product | Describe all product in the system |
| Category | Describe all category in the system |
| Category\_has\_Product | Describe all product category in the system |
| Manufacturer | Describe all manufacturer in the system |
| Country | Describe all country in the system |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Entity name** | **Attributes** | **Description** | **Domain** | **Null** |
| Role | roleID {PK} | Unique identifier of role, auto increment. | Integer | No |
| roleType | User’s role. | Text | No |
| User | userID |  |  |  |
|  | username |  |  |  |
|  | Pwd |  |  |  |
|  | pinCode |  |  |  |
|  | lastName |  |  |  |
|  | firstName |  |  |  |
|  | age |  |  |  |
|  | DOB |  |  |  |
|  | idCard |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

1. **Performance Measures**
   1. **Clustering Performance**

Clustering is performed by running K Mean Algorithm which has complexity of : O(n \* k \* I \* d)

* n : number of points
* k : number of cluster
* I : number of iteration
* d : number of attributes (3)

Clustering take almost the time of process that we can ignore the time needed to load data from database, digitalize data.

The speed of clustering will vary and increase dramatically when n increase. The purpose of this project is not about optimizing K-Mean Algorithm so it is accepted to let the process run till it completes. Moreover, the clustering is designed to run by staff, wait time is acceptable.

1. Test plan

We have two main test phases: Function test (Integration test) and System test.

For Integration test, we use the functional test. This kind of test shows us the working of each function right or wrong. We focus on the result of the function in different cases.

The System test check all the activities the Smart lock after setup such as: what the system do when it is turned on and if user do somethings, what will response. We create test case to check each function’s activity. Then, we observe and record the result.

* 1. Features to be tested

- Functional Testing:

+ Integration testing: Test all function on Raspberry Application and each external module

+ System testing: Test all function for Raspberry Application – external modules

- UI Testing: Test UI on Raspberry Application

* 1. Features not to be tested

- Hardware module: Raspberry Pi B2.

* 1. Testing tools and environment

- Testing tools:

+ A raspberry board with connect to PC/Laptop running Linux OS

+ QT Creattor

+ Minicom on Raspberry

- Environment:

+ Some where with good light condition

* 1. Test past/ fail criteria

For system testing, the criteria are:

* 90% of the test cases must pass.
* 100% of test cases about hardware module must pass.
* All test cases dealing with critical functionality must pass.
* All medium and high severity defects must be fixed.
* Test coverage must be at least 90%.

5. System Testing Test Case

5.1. Test on real device

5.1.1. Component testing

G. Appendix

1. Rasbian operating system:

http://www.raspbian.org/RaspbianAbout

2. OpenCV library to detect object: <http://docs.opencv.org/modules/objdetect/doc/cascade_classification.html>

3. Raspberry Pi Camera module document:

<http://www.ics.com/blog/raspberry-pi-camera-module#.VDqccvldX6U>

4. GPIO library for Raspberry:

<http://www.airspayce.com/mikem/bcm2835/>

5. Setup rasbian OS:

http://chiaseprojects.blogspot.com/2014/06/huong-dan-cai-at-he-ieu-hanh-raspbian.html

6. Development in/out with raspberry:

http://codientu.org/threads/10519/

7. UART raspberry guideline:

<http://www.raspberry-projects.com/pi/programming-in-c/uart-serial-port/using-the-uart>