|  |  |
| --- | --- |
|  | **MINISTRY OF EDUCATION AND TRAINING** |

|  |  |
| --- | --- |
| **FPT UNIVERSITY** | |
| CAPSTONE PROJECT DOCUMENT | |
| **HANGING PRODUCT STORE** | |
| **SE07.B – GROUP 8** | |
| **Group Members** | |  |  |  | | --- | --- | --- | | Quản Tiến Dân | Team Leader | DanQTSE61144 | | Trần Quang Phúc | Team Member | Dropped out | | Nguyễn Huy Hoàng | Team Member | HoangNHSE61007 | | Huỳnh Công Đức | Team Member | DucHCSE61024 | |
| **Supervisor** | Kiều Trọng Khánh |
| **Ext.Supervisor** | N/A |
| **Capstone Project code** | HPS |

- Ho Chi Minh City, May/2015 -

*This page is intentionally left blank*

Table of Contents

[A. Report No.1 Introduction 9](#_Toc427011695)

[1. Project Information 9](#_Toc427011696)

[2. Introduction 9](#_Toc427011697)

[3. Current Situation 9](#_Toc427011698)

[4. Problem Definition 9](#_Toc427011699)

[5. Proposed Solution 10](#_Toc427011700)

[5.1 Feature functions 10](#_Toc427011701)

[5.2 Advantages and disadvantages 10](#_Toc427011702)

[6. Functional Requirements 11](#_Toc427011703)

[7. Role and Responsibility 11](#_Toc427011704)

[B. Report No.2 Software Project Management Plan 12](#_Toc427011705)

[1. Problem Definition 12](#_Toc427011706)

[1.1 Name of this Capstone Project 12](#_Toc427011707)

[1.2 Problem abstract 12](#_Toc427011708)

[1.3 Project Overview 12](#_Toc427011709)

[2. Project Organization 14](#_Toc427011710)

[2.1 Software Process Model 14](#_Toc427011711)

[2.2 Roles and responsibilities 17](#_Toc427011712)

[2.3 Tools and Techniques 17](#_Toc427011713)

[3. Project Management Plan 19](#_Toc427011714)

[3.1 Software development life cycle 19](#_Toc427011715)

[3.2 Phase Detail 21](#_Toc427011716)

[3.3 All Meeting Minute 24](#_Toc427011717)

[4. Coding Convention 24](#_Toc427011718)

[C. Report No.3 Software Requirement Specification 25](#_Toc427011719)

[1. User Requirement Specification 25](#_Toc427011720)

[1.1 Guest Requirement 25](#_Toc427011721)

[1.2 Customer Requirement 25](#_Toc427011722)

[1.3 Store Owner Requirement 25](#_Toc427011723)

[1.4 Store Owner Requirement 25](#_Toc427011724)

[2. User Requirement Specification 26](#_Toc427011725)

[2.2 External Interface Requirement 26](#_Toc427011727)

[2.3 System Overview Use Case 26](#_Toc427011728)

[2.4 List of Use Case 26](#_Toc427011729)

[3 Software System Attribute 64](#_Toc427011730)

[3.1 Usability 64](#_Toc427011731)

[3.2 Reliability 64](#_Toc427011732)

[3.3 Availability 64](#_Toc427011733)

[3.4 Security 65](#_Toc427011734)

[3.5 Maintainability 65](#_Toc427011735)

[3.6 Portability 65](#_Toc427011736)

[3.7 Performance 65](#_Toc427011737)

[4 Conceptual Diagram 65](#_Toc427011738)

[4.1 Conceptual Diagram 65](#_Toc427011739)

[D. Report No. 4 Software Design Description 67](#_Toc427011740)

[1. Design Overview 67](#_Toc427011741)

[2. System Architectural Design 67](#_Toc427011742)

[4.1 Web Application Architecture Description 68](#_Toc427011743)

[4.2 Mobile Application Architecture Description 68](#_Toc427011744)

[3. Component Diagram 69](#_Toc427011745)

[4. Detailed Description 70](#_Toc427011746)

[4.1 Class Diagram 70](#_Toc427011747)

[4.2 Class Diagram Explanation 70](#_Toc427011748)

[4.3 Interaction Diagram 76](#_Toc427011749)

[5. User Interface Design 84](#_Toc427011750)

[5.1 Guest Interface Design 84](#_Toc427011751)

[5.2 User Interface Design 87](#_Toc427011752)

**List of Tables**

[*Table 1: Roles and Responsibilities* 8](#_Toc420144130)

[*Table 2: Project Role and Responsibility* 12](#_Toc420144131)

**List of figures**

[*Figure 1: SCRUM Overview Diagram* 11](#_Toc420144138)

**Definition, Acronyms, and Abbreviations**

|  |  |
| --- | --- |
| **Name** | **Definition** |
| HPS | Hanging Product Store |
| App | Application |
| OS | Operating System |

# Report No.1 Introduction

## Project Information

* **Project Name:** Hanging Product Store
* **Project Code:** HPS
* **Product Type:** Website and Android Application
* **Start Date:** May 11, 2015
* **End Date:** September 3, 2015

## Introduction

Nowadays, because of the shopping hobbit, many people are willing to buy product even they just use it once. Therefore, they probably have items hanging in their closet that they never use again. Hence, consignment sales service appears.

Currently, in Viet Nam consignment services becomes more popular. There are some consignment sales services but they require consignor brings items to the store and some use manually process in management (give receipts to consignor for tracking).

From above mentioned reasons, this project is going to build an online system for consignment sales service.

## Current Situation

This kind of business is now becoming more popular in daily life, with variety genres of product, no capital. In other country likes US, TheRealReal is very successful. In Viet Nam, there are some consignment sales services such as LALA-Save&Share, khohanghoa.com… They are running their business and expanding their business very well.

## Problem Definition

Here is their advantages and disadvantages:

* LALA-Save&Share:
* Advantages:
* Provide online shopping method.
* Online support available.
* Support consigned-items tracking.
* Disadvantages:
* Doesn’t support online payment for consignor when consigned items are sold.
* Consignor has to bring theirs item to the store to consign.
* Support only fashion product.
* KhoHangHoa.com:
* Advantages:
* Support many types of product.
* Provide online shopping method.
* Support online payment for consignor when consigned-items are sold.
* Online support available.
* Disadvantages:
* Consignor has to bring theirs item to the store to consign.
* Consignor cannot track their consigned-items status immediately.
* TheRealReal.com:
* Advantages:
* Online support available.
* Provide online shopping method, online payment for both consignment and purchase.
* Support product recommendation by season or fashion trend and wish list.
* Support consigned-items tracking.
* Disadvantages:
* Support only fashion product.
* Consignor has to bring theirs item to the store to consign.Problem

## Proposed Solution

Our system include a web app helps user sell consigned products and an android app helps store owner to receive product.

In more detail, our system has the following functions:

### Feature functions

* Web Application:
* Admin can create account and manage account.
* Guest can make product consignment request, search product and order product. The system support Guest tracking their product status, receive notification about their product status, payment online method and price suggestion.
* Customer can edit profile, search and review their transaction history.
* Store owner can manage consignment request, manage receive product, raise their products on website and manage order product. The system support Store owner tracking consigned product status, payment online method, price suggestion, website statistics.
* Mobile Application:
* StoreOwner can use their Android phone to manage receive product.

### Advantages and disadvantages

* Advantages:
* Support price suggestion.
* Support online payment method.
* Support website statistics.
* Support notification about consigned product status.
* Support tracking product status.
* Disadvantages:
* Support only fashion product.
* Internet connection is required.

## Functional Requirements

Function requirements of the system are listed as below:

* Account management
* Product items management
* Raise products on website
* Order management
* Website’s statistics
* Price suggestion
* Online payment method
* Product status notification
* Product status tracking

## Role and Responsibility

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Full Name** | **Role** | **Position** | **Contact** |
| **1** | Kiều Trọng Khánh | Senior Manager | Instructor | [KhanhKT@fpt.edu.vn](mailto:KhanhKT@fpt.edu.vn) |
| **2** | Quản Tiến Dân | Developer | Leader | [DanQTSE61144@fpt.edu.vn](mailto:DanQTSE61144@fpt.edu.vn) |
| **3** | Trần Quang Phúc | Developer | Member | [PhucTQSE60993@fpt.edu.vn](mailto:PhucTQSE60993@fpt.edu.vn) |
| **4** | Nguyễn Huy Hoàng | Developer | Member | [HoangNHSE61007@fpt.edu.vn](mailto:HoangNHSE61007@fpt.edu.vn) |
| **5** | Huỳnh Công Đức | Developer | Member | [DucHCSE61024@fpt.edu.vn](mailto:DucHCSE61024@fpt.edu.vn) |

*Table 1: Roles and Responsibilities*

# Report No.2 Software Project Management Plan

## Problem Definition

### Name of this Capstone Project

Hanging Product Store (HPS)

### Problem abstract

With the increasing of shopping in community, the number of consignment store has started to increase too. A consignor has so many choices to consign his/her product and consignor may receive a lots of consigned product every day. Both consignor and consignee look for a way to perform business transaction online without worries. We propose a solution to solve this problem: a website where consignor and consignee can meet and do consignment business online. For consignor, we provide an easy way to get consignees’ attentions, make contract with them and track consigned product’s status. For consignee, we provide a useful tool to check the consigned products and deal with consignors.

### Project Overview

#### Current Situation and Disadvantages

Although there are some shops running this service, most of them are not really effective in helping consignor tracking product’s status, negotiating price between consignor and consignee. When a consignor, especially inexperienced consignor is in need of consignment, they couldn't know how to find a good consignee and wonder if that consignee is reliable. And when product has already been consigned, how could he/she know if consigned product is in store or is being sold? That problem goes to consignees as well. With many consigned products, how could they track each item efficiently? They also need a better way to automatically notify consignors when any actions is applied to consigned products.

#### The Proposed System

##### Web Site

* Admin can manage account
* Guest can make product consignment request.
* Guest can receive notification about their product status.
* Guest track product status.
* Guest search product.
* Guest order product.
* Customer can edit profile
* Customer can review transaction history
* Store owner can receive suggested price for product.
* Store owner can manage receive product.
* Store owner can raise product on website.
* Store owner can manage consigned product.
* Store owner can manage order product.
* Store owner can have online payment method to payment for customer.
* Store owner can track consigned product status.
* Store owner can manage order product.
* System support send sms to notify product’s status
* System support online payment method.
* System support price suggestion.

##### Mobile Application

* Store owner can raise product on website.

#### Boundaries of System

* People who have computer, laptop or smartphone with internet connection and could run web browser can use this website.
* The complete product of this project includes:
* The website application.
* The mobile application.
* All related documents.
* Language: Vietnamese.

#### Development Environment

##### Hardware requirement

For mobile:

|  |  |  |
| --- | --- | --- |
| Mobile | Minimum Requirements | Recommended |
| Internet Connection | 2 Mbps | 4 Mbps |
| Operating System | Android 4.4 | Android 5.0 |
| Hardware | Wifi supported | Wifi supported |
| Memory | 1GB | 2GB |

For server:

|  |  |  |
| --- | --- | --- |
| Windows | Minimum Requirements | Recommended |
| Internet Connection | Cable, Wi-Fi (4 Mbps) | Cable, Wi-Fi (8 Mbps) |
| Operating System | Window Server 2008 | Window Server 2008 |
| Computer Processor | Intel® Core 2 Duo | Intel® Core(TM) i5 CPU , M 460 @ 2.53Gz |

##### Software requirement

* GitHub & TortoiseGit: used for source control.
* NetBean 8.0.2 and Apache Tomcat 8.0.15: used to develop and implement.
* Android Studio 2.1: use to code client application
* Database: Microsoft SQL Server 2008.
* Genymotion 2.4.0: use to run and test Android app.
* StarUML: used to create models and diagrams.
* Skype: use for meeting and conservation between group members

## Project Organization

### Software Process Model

To determine the right process model for the software process, we answer ten guidelines questions (Sommerville, Software Engineering, 2011):

| No. | Question | Answer | Plan-Driven | Agile |
| --- | --- | --- | --- | --- |
| 1. | Is it important to have a very detailed specification and design before moving to implementation? If so, you probably need to use a plan-driven approach. | We need a correct specification and design to coding but a very detailed one is not necessary. | 0.5 | 0.5 |
| 2. | Is an incremental delivery strategy, where you deliver the software to customers and get rapid feedback from them, realistic? If so, consider using agile methods. | No. This is generic product, not bespoke product. A rapid feedback from target customer is not necessary. | 1 | 0 |
| 3. | How large is the system that is being developed? Agile methods are most effective when the system can be developed with a small co-located team who can communicate informally. This may not be possible for large systems that require larger development teams so a plan-driven approach may have to be used. | This system will be developed in four month by four people, so we think it is a medium system. | 0.5 | 0.5 |
| 4. | What type of system is being developed? Systems that require a lot of analysis before implementation (e.g., real-time system with complex timing requirements) usually need a fairly detailed design to carry out this analysis. A plan-driven approach may be best in those circumstances. | This is an ecommerce system. Core components require analysis carefully. | 1 | 0 |
| 5. | What is the expected system lifetime? Long-lifetime systems may require more design documentation to communicate the original intentions of the system developers to the support team. However, supporters of agile methods rightly argue that documentation is frequently not kept up to date and it is not of much use for long-term system maintenance. | This system may operate in two or three years. | 1 | 0 |
| 6. | What technologies are available to support system development? Agile methods often rely on good tools to keep track of an evolving design. If you are developing a system using an IDE that does not have good tools for program visualization and analysis, then more design documentation may be required. | We have a good tool to tracking change. The IDE support tracking phase. | 0.5 | 0.5 |
| 7. | How is the development team organized? If the development team is distributed or if part of the development is being outsourced, then you may need to develop design documents to communicate across the development teams. You may need to plan in advance what these are. | We are a small with four people and we are all living in Ho Chi Minh city. We can have meeting every day. | 0 | 1 |
| 8. | Are there cultural issues that may affect the system development? Traditional engineering organizations have a culture of plan-based development, as this is the norm in engineering. This usually requires extensive design documentation, rather than the informal knowledge used in agile processes. | We are all young people and we share common software development knowledge. However, we come from difference part of country and we never met before. | 0.5 | 0.5 |
| 9. | How good are the designers and programmers in the development team? It is sometimes argued that agile methods require higher skill levels than plan-based approaches in which programmers simply translate a detailed design into code. If you have a team with relatively low skill levels, you may need to use the best people to develop the design, with others responsible for programming. | Although we are confident about software development skill. We are still student and we have no or little experience. | 1 | 0 |
| 10. | Is the system subject to external regulation? If a system has to be approved by an external regulator (e.g., the Federal Aviation Authority [FAA] approve software that is critical to the operation of an aircraft) then you will probably be required to produce detailed documentation as part of the system safety case. | No | 0 | 0 |
| Total | | | 6 | 3 |

As the result, the plan-driven process we choose is waterfall model because of its easy to use and commonly used in software process.



Because of the small size of the team (four people included project manager), we may overlap five phases in waterfall model to cope with change better and to reduce the wasted time (due to the blocking nature of the waterfall model – the following phase must wait until the previous phase has finished).

During Requirement Definition, we will design some component that has clear requirement. During coding, we will fix design problem and so on. We will modify the document produced in each phase to reflect the changes made.

### Roles and responsibilities

|  |  |  |  |
| --- | --- | --- | --- |
| No. | Full name | Team Role | Responsibilities |
| **1** | Kieu Trong Khanh | * Senior Manager | * Monitor project * Mentor Project Manager * Specify user requirement * Review documents * Support technology |
| **2.** | Quan Tien Dan | * Project Manager * Business Analyst * Developer * Tester | * Monitor project * Plan and execute project * Design * Programing * Testing |
| **3.** | Tran Quang Phuc | * Team member * Business Analyst * Developer * Tester | * Assist in tools and techniques * Design * Programing * Testing |
| **4** | Nguyen Huy Hoang | * Team member * Business Analyst * Developer * Tester | * Assist in tools and techniques * Design * Programing * Testing |
| **5** | Huynh Cong Duc | * Team member * Business Analyst * Developer * Tester | * Assist in tools and techniques * Design * Programing * Testing |

*Table 2: Project Role and Responsibility*

### Tools and Techniques

* Mobile: Android Lollipop 4.0 and later
* Web Server: Installed Tomcat 8.0.15.0
* Database Management System: MSSQL Server 2008 R2 Enterprise
* API Service: Amazon Product Advertising API
* Communication technique: HTTP, SMTP
* Web service: Java + Jersey
* Client technology: HTML5, CSS, Javascript,

## Project Management Plan

### Software development life cycle

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| Phase | Description | Deliverables | Resource needed | Dependencies and Constraints | Risks |
| Requirement Definition | The process of establishing what services are required and the constraints on the system’s operation and development. | Software requirement specification:   * System models * User and system requirements   Executable prototype  Acceptance test plan | 52 man days |  | Lack of domain knowledge.  Communication misunderstanding.  Conflict objectives, conflict viewpoints. |
| System and Software Design | Design a software structure and an overall system architecture that realizes the specification. | System design description:   * System architecture * Database specification * Interface specification * Component Specification   System integration test plan | 52 man days | Platform information  Software requirement specification  Data description | Change to requirements that require major design rework are proposed.  Specifications of essential interfaces are not available on schedule.  The size of the system has been underestimated. |
| Implementation and Unit Testing | Translate design structure into an executable program or program unit.  Unit testing involves verifying that each unit meets its specification. | Program Units  Test case | 90 man days | System design description | Software tool cannot be integrated.  The time required to develop the software is underestimated.  The underlying technology on which the system is build is superseded by new technology. |
| Integration and System Testing | Integrated the individual program unit.  Test the complete system to ensure that the software requirements have been meet. | Executable software  Test result | 58 man days | System design description  System integration test plan  Acceptance test plan | Reusable software components contain defects that mean they cannot be used as planned. |
| Operation and Maintenance | This is the longest life cycle phase.  The system is installed and put into practical use.  Maintenance involves correcting errors which were not discovered in earlier stages of the life cycle, improving the implementation of system units and enhancing the system’s services as new requirements are discovered. | Installation guide  User guide | 92 man days | Executable software  Software requirement specification | A competitive product is marketed before the system is completed.  The database used in the system cannot process as many transactions per second as expected. |

### Phase Detail

#### Phase 1: Requirement Definition

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| Collect requirements | Find which systems currently provide similar service, their advantages and dis advantages. | DanQT, PhucTQ, HoangNH, DucHC |
| Identify and clarify main functions | Define main functions of system. | DanQT, PhucTQ, HoangNH, DucHC |
| Create System Introduction | Complete Introduction Report. | DanQT, PhucTQ, HoangNH, DucHC |
| Software Project Management Plan. | Prepare Project Management Plan. | DanQT, PhucTQ, HoangNH, DucHC |
| Website Prototype. | Build a prototype of system. | DanQT, PhucTQ, HoangNH, DucHC |
| SRS | Create SRS document. | DanQT, PhucTQ, HoangNH, DucHC |

#### Phase 2: System and Software Design

|  |  |  |
| --- | --- | --- |
| Task | Description | Author |
| Architecture Design | Create an architecture design that can meet system requirements. | DanQT, PhucTQ, HoangNH, DucHC |
| Detailed Design | Create a detailed design with diagrams and relevant artifacts. | DanQT, PhucTQ, HoangNH, DucHC |
| Database Design | Design database. | DanQT, PhucTQ, HoangNH, DucHC |
| Technology research | Research solution for ordered functions. | DanQT, PhucTQ, HoangNH, DucHC |

#### Phase 3: Implementation and Unit Testing

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| Price suggestion and payment function. | Suggest price for consignor and consignee.  Perform online payment method. | DanQT |
| Price negotiation between consignor and consignee. | Submit desirable price.  Accept suggested price.  Refuse suggested price. | PhucTQ |
| Raise product on web and advanced search function. | Raise product on web by season.  Search product function.  Support shopping online. | HoangNH |
| Consign product and track consigned-product status. | Consign product.  Support tracking and notification consigned-product status. | DucHC |
| Unit testing. | Create and perform unit testing for price suggestion and online payment function. | DanQT |
| Create and perform unit testing for price negotiation function. | PhucTQ |
| Create and perform unit testing for raise product on web and advanced search function. | HoangNH |
| Create and perform unit testing for consign product and tracking consigned-product status function. | DucHC |

#### Phase 4: Integration and System Testing

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| Integration Testing | Write test case and test for integrate functions. | DanQT, PhucTQ, HoangNH, DucHC |
| System Testing | Write test case and test for whole system. | DanQT, PhucTQ, HoangNH, DucHC |

Table : Phase 4: Integration and System Testing

#### Phase 5: Operation and Maintenance

|  |  |  |
| --- | --- | --- |
| **Task** | **Description** | **Author** |
| Installation Guide | Complete installation guide | DanQT, PhucTQ, HoangNH, DucHC |
| User’s Guide | Complete user’s guide | DanQT, PhucTQ, HoangNH, DucHC |

Table : Phase 5: Operation and Maintenance

### All Meeting Minute

All meeting minute are placed in Meeting Minute at GitHub repository.

## Coding Convention

* Naming Convention:
* For variable and function name: Camel case
* For class name: Pascal case
* Code Examples
* Follow “Code Conventions for the Java TM Programming Language, by Sun Microsystems, rev April 20, 1999”. <http://www.oracle.com/technetwork/java/codeconventions-150003.pdf>

# Report No.3 Software Requirement Specification

## User Requirement Specification

### Guest Requirement

A guest of this system has the following abilities:

* Login
* Register
* Search product
* Consign product
* Order product
* Track consignment status or cancel consignment with consignmentID.

### Customer Requirement

A Customer of this system has the following abilities:

* Search product
* Consign product
* Order product
* Track consignment status or cancel consignment.
* Edit profile.
* View consignment history.

### Store Owner Requirement

A Store Owner of this system has the following abilities:

* View consignment requests.
* View consignment request detail.
* Accept or refuse consignment request.
* Receive product.
* View product status.
* Cancel consignments.
* Publish product on web.
* Pay consignor with online payment.
* View selling statistic.
* Edit profile.

### Store Owner Requirement

An Admin of this system has the following abilities:

* Create new user.
* De-active user.
* Active user.
* Update user profile.

## User Requirement Specification

### External Interface Requirement

#### User Interface

* General requirement for graphic user interface design of mobile device is the GUI should be simple, clear, intuitive, and reminiscent.
* The interface design is an iterate process includes design, sketching, prototyping, user assessment.
* Some design principles are taken into consideration:
  + Material design for android Lollipop – Google [Ref: <https://www.google.com/design/spec/material-design/introduction.html>]

#### Hardware Interface

* N/A

#### Software Interface

* Run with FireFox (version 37.0.1), Chrome (version 42.0.2311.90 m) browser and must support Javascript and HTML5.

#### Communication Protocol

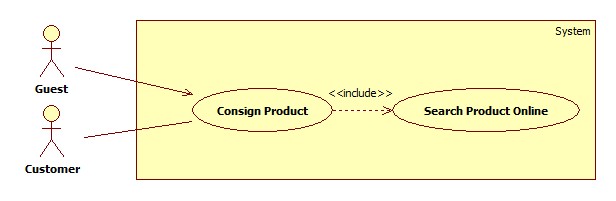
* Using HTTP protocol

### System Overview Use Case

### List of Use Case

#### <User> Overview Use Case

##### <Guest> Consign Product Use Case



**Use Case Specification**

##### <Guest> Track product status Use Case

****

**Use Case Specification**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – HPS002** | | | |
| **Use Case No.** | HPS002 | Use Case Version | 2.0 |
| **Use Case Name** | Track product status | | |
| **Author** | Huynh Cong Duc | | |
| **Date** | May 29, 2015 | Priority | High |
| **Actor**:   * Guest   **Summary**:   * This use case allows guest to track product status by code.   **Goal**:   * Guest views product’s current status successfully.   **Triggers**:   * Guest sends tracking product’s status command.   **Preconditions**: N/A  **Post Conditions**:   * **Success:** View product’s current status. * **Fail:** Show error message.   **Main Success Scenario**:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Guest sends tracking product’s status command. | System shows guest following:   * “Mã sản phẩm”: free text input * “Kiểm tra”: send command to track product’s status. | | 2 | Guest inputs “Mã sản phẩm” and sends command to track product’s status. | System finds product with provided code and shows guest with following information:   * “Hình ảnh”: A large field to show image of product. * “Tên sản phẩm”: label. * “Loại”: label. * “Hãng”: label * “Ngày ký gửi”: label. * “Trạng thái”: label. * “Người ký gửi”:   + “Tên”: label.   + “Địa chỉ”: label.   + “Điện thoại”: label.   + “Email”: label. * “Cửa hàng”: label.   System also shows following:   * “Hủy ký gửi”: send command to cancel consignment.   [Exception: 1, 2] |   **Alternative Scenario**: N/A  **Exceptions**:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Inputted value of “Mã sản phẩm” field is empty. | System shows error message: “Xin nhập mã sản phẩm”. | |  | System cannot find product with inputted “Mã sản phẩm”. | System shows error message: “Sản phẩm với mã số này không tồn tại”. |   **Relationships**:   * Extending Use-case: Cancel Consigned Product (Guest sends a request to cancel a consigned products) and Extend Consigned Product (Guest send a request to extend product).   **Business Rules**:   * Guest can only track product’s status by code. * Code is ID of consignment. * User can only cancel product before it is ordered. | | | |

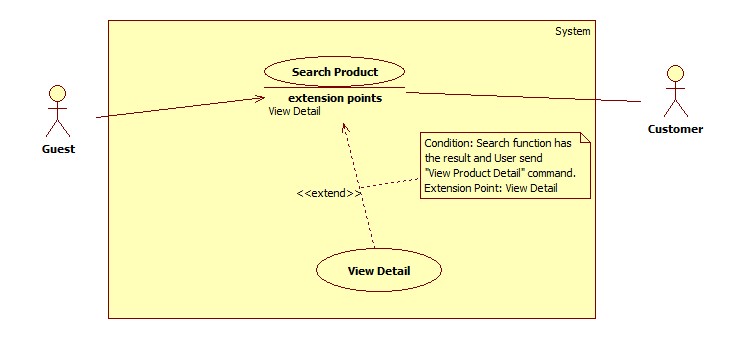
##### <Guest> Cancel Consigned Product Use Case

****

**Use Case Specification**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – HPS003** | | | |
| **Use Case No.** | HPS003 | Use Case Version | 2.0 |
| **Use Case Name** | Cancel consigned product | | |
| **Author** | DucHC | | |
| **Date** | May 29, 2015 | Priority | High |
| **Actor**:   * Guest   **Summary**:   * This use case allows guest to cancel a consigned product.   **Goal**:   * Guest cancels a consigned product successfully.   **Triggers**:   * Guest sends canceling consigned product command.   **Preconditions**:   * Guest is viewing a consigned product’s information. * Status of product is neither “Sold” nor “Completed”.   **Post Conditions**:   * **Success:** Product’s status is changed to “Canceled”. * **Fail:** Show error message.   **Main Success Scenario**:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Guest sends canceling consigned product command.  [Alternative 1, 2] | System shows a confirming message if guest really wants to cancel this consigned product with two options “Đồng ý” and “Không”. | |  | Guest chooses “Đồng ý” option.  [Alternative 3] | Product’s status is changed to “Đã hủy” or “Đăng ký hủy”. |   **Alternative Scenario**:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Guest sends canceling consigned product command when product is not received | System shows a message to make sure they really want to cancel it with 2 options: “Đồng ý” or “Không”. | | 2 | Guest sends canceling consigned product command when product is already received but not ordered. | System shows a message to make sure they really want to cancel it with a price for storing product and 2 options: “Đồng ý” hoặc “Không”. | | 3 | Guest chooses “No” option. | Message will disappear. |   **Exceptions**: N/A  **Relationships**:   * Extended Use-case: Track product status.   **Business Rules**:   * Guest can only cancel product before it’s ordered. * If guest cancels before store owner gets product, consign product process will be stopped. * If guest cancels after store owner gets product, guest will be informed to take back product in a specific time and pay a forfeit for canceling consignment. After that time and guest doesn’t come to take back product, store owner will have it. | | | |

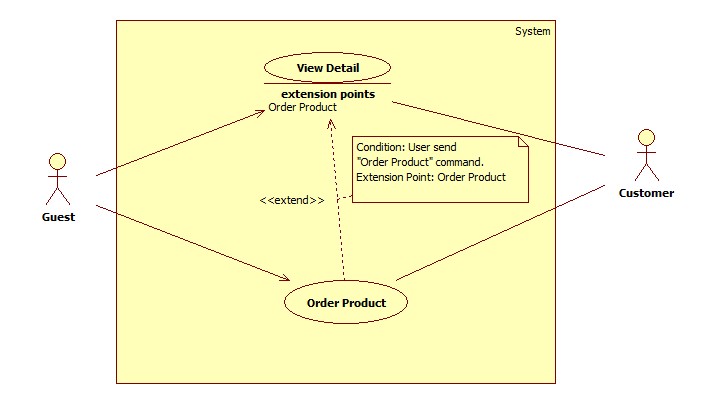
##### <Guest> Search Product Use Case



**Use Case Specification**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE –** HPS004 | | | |
| **Use Case No.** | HPS004 | Use Case Version | 2.0 |
| **Use Case Name** | Search Product | | |
| **Author** | Nguyen Huy Hoang | | |
| **Date** | May 22, 2015 | Priority | High |
| **Actor**:   * Guest/Customer   **Summary**:   * This use case allows user to search product information by its name and category.   **Goal**:   * A list of products will be shown to user.   **Triggers**:   * User can search product to get product’s information. * User can choose top product’s category include: “Nam”, “Nữ”, “Trẻ em” or “Tất cả”, inputs to a free text input and send “Search Product” command.   **Preconditions**:   * User is in home page or category page.   **Post Conditions**:   * Success: Show search results which include a list of product. * Fail: Show an error message.   **Main Success Scenario**:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User choose product’s category include: “Nam”, “Nữ”, “Trẻ em” or “Tất cả”. |  | | 2 | User inputs the product’s name into free text input. |  | | 3 | User send “Search Product” command. | [Exception 1]  System will show a list of product as a grid and order by product name.  [Alternative 1] |   **Alternative Scenario**:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Search not found | System will show a message: “Không tìm thầy sản phẩm nào phù hợp.” |   **Exceptions**: N/A  **Relationships**:   * Extending Use-cases: Extended by View Detail (Search function has the result and user send “View Product Detail” command).   **Business Rules**:   * Search by product name and category. * All products with relative name which user inputted will be show. The results will show list of product as a gird. | | | |

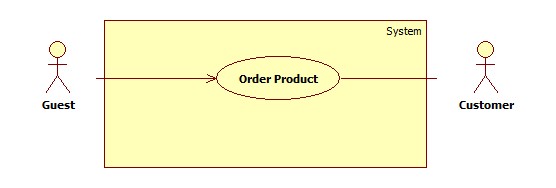
##### <Guest> View Detail Use Case



**Use Case Specification**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE –** HPS005 | | | |
| **Use Case No.** | HPS005 | Use Case Version | 2.0 |
| **Use Case Name** | View Detail | | |
| **Author** | Nguyen Huy Hoang | | |
| **Date** | May 22, 2015 | Priority | High |
| **Actor**:   * Guest/Customer   **Summary**:   * This use case allows user to view product detail information.   **Goal**:   * A product detail will be shown to user.   **Triggers**:   * User send “View Product Detail” command.   **Preconditions**:   * At least 1 product item available.   **Post Conditions**:   * Success: Show product detail information in detail page. * Fail: Show an error message.   **Main Success Scenario**:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User click “XEM CHI TIẾT” button. | System will show product information which contains:   * “Tên” : Label * “Tình trạng”: Label * “Số Serial” : Label * “Mô tả”: Textarea * “Hình ảnh”: Image * “Tên chủ cửa hàng”: Label * “Email”: Label * “Số điện thoại”: Label * “Địa chỉ”: Label * A command to order product. |   **Alternative Scenario**: N/A  **Exceptions**: N/A  **Relationships**:   * Extending Use-cases: Extend from Search Product (Search keyword has the result and user click a result item). Extended by Order Product(User send “Order Product” command).   **Business Rules**:   * Details of product will be displayed to user. | | | |

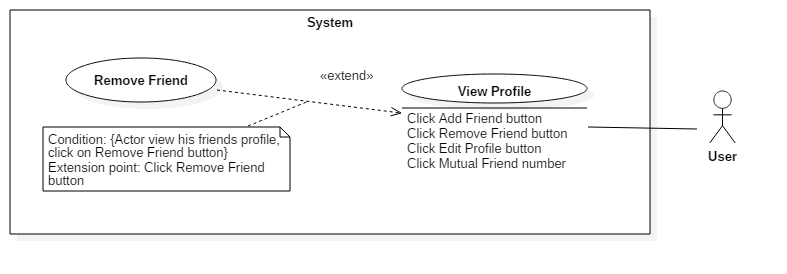
##### <Guest> Order Product Use Case



**Use Case Specification**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE –** HPS006 | | | |
| **Use Case No.** | HPS006 | Use Case Version | 2.0 |
| **Use Case Name** | Order Product | | |
| **Author** | HoangNH | | |
| **Date** | May 22, 2015 | Priority | High |
| **Actor**:   * Guest/Customer   **Summary**:   * This use case allows user order product.   **Goal**:   * Product will be ordered by user.   **Triggers**:   * User send “Order Product” command.   **Preconditions**:   * User is in product detail page.   **Post Conditions**:   * Success: Show success message, create an order of this product with user information * Fail: Show an error message.   **Main Success Scenario**:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | User send “Order Product” command. | System will navigate to enter user’s information include:   * “Tên”: Free text input, max length: 50. * “Email”: Free text input, regular expression: “^[a-zA-Z0-9\_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+$". * “Địa Chỉ”: Free text input. * “Số Điện Thoại”: Free text input, min length: 9, max length: 11. * A command to finish action.   [Alternative 1] | | 2 | User input information. |  | | 3 | User send “Confirm Information” command. | [Exception 1,2]  [Alternative 2]  System create an order of this product with user information. | |  |  |  |   **Alternative Scenario**:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | If product have been ordered | System will show message: “Món hàng này đã có A người đặt, có thể bạn sẽ không mua được món hàng, vui lòng liên hệ với chủ cửa hàng để biêt thêm." With A is number of orders. | | 2 | If product is not special product. | System will send sms to notify price to user. |   **Exceptions**:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User not fills all required textbox | Show message: “Vui lòng nhập thông tin vào nhưng ô bắt buộc.” | | 2 | User enter phone number is existed in another order of this product. | Show message: “Số điện thoại đã sử dụng để đặt món hàng này”. |   **Relationships**:   * Extending Use-cases: Extend from Product Detail (Search function has the result and user send “View Product Detail” command).   **Business Rules**:   * All require fill must be filled. * User can’t use same phone number to order same product. | | | |

#### <Customer> Overview User Case

****

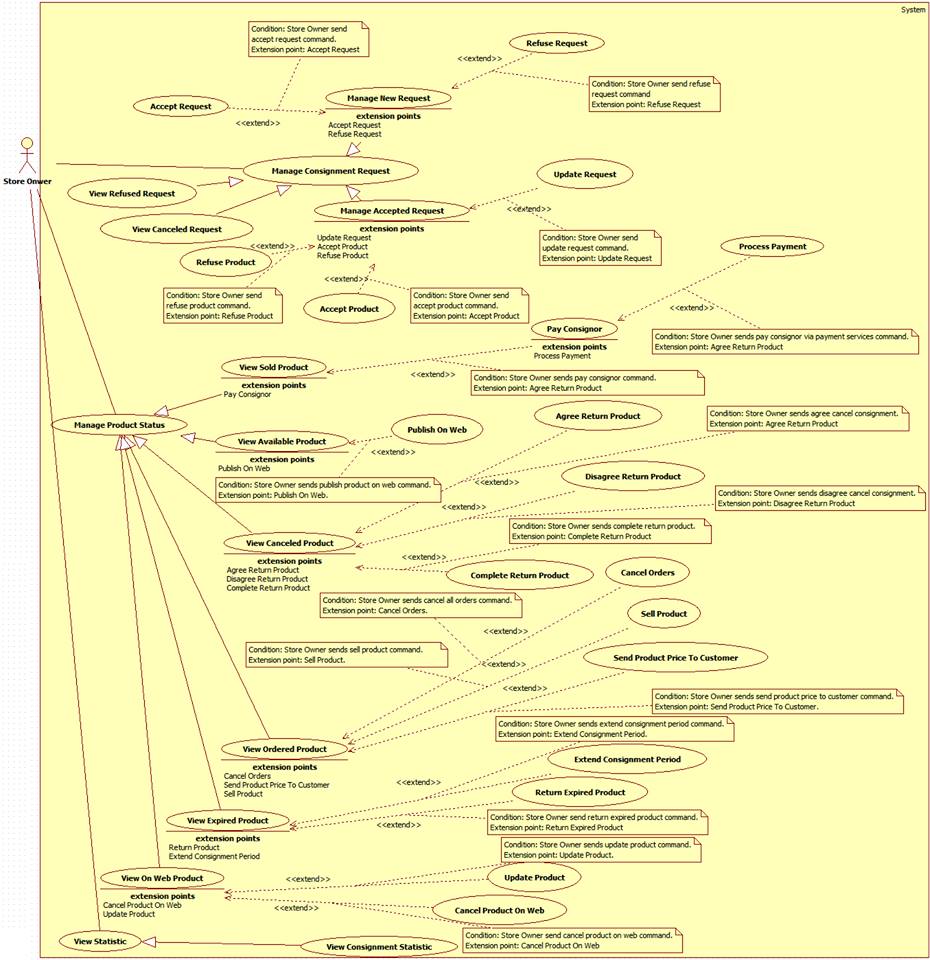
**Use Case Specification**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE – MDA006** | | | |
| **Use-case No.** | MDA006 | **Use-case Version** | 1.0 |
| **Use-case Name** | Remove Friend | | |
| **Author** | Pham Van Thang | | |
| **Date** | 31/07/2015 | **Priority** | High |
| **Actor:**   * User   **Summary:**   * This use case allows user remove his friends.   **Goal:**   * User and his friend are no more friend relation.   **Triggers**   * User click “Remove Friend” button on View Profile screen.   **Preconditions:**   * Guest logged in system successfully. * User in profile of his friend screen.   **Post Conditions:**   * **On Success**: Friend relationship is removed. * **On Failure**: Show error message.   **Main Success Scenario:**   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Click on “Remove friend” button. | Show confirmation dialog. | | 2 | Click on “Đồng ý” button.  [Alternative 1] | Delete relationship between user and target user.  Show successful message.  [Exception 1, 2] |   **Alternative Scenario:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Click on “Hủy” button. | Dismiss dialog. |   **Exceptions:**   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Internet is not available. | System show error message: “Không có kết nối Internet”. | | 2 | Connect to web service is error. | System show error message: “Xảy ra lỗi khi kết nối”. |   **Relationships:**   * Extended by View Profile (Click Remove Friend button).   **Business Rules:**   * Successful message is show as format: “Đã hủy kết bạn với A thành công.” with A is target user full name. | | | |

##### <Customer> Logout Use Case Diagram

**Use Case Diagram**

#### <Store Owner> Overview Use Case



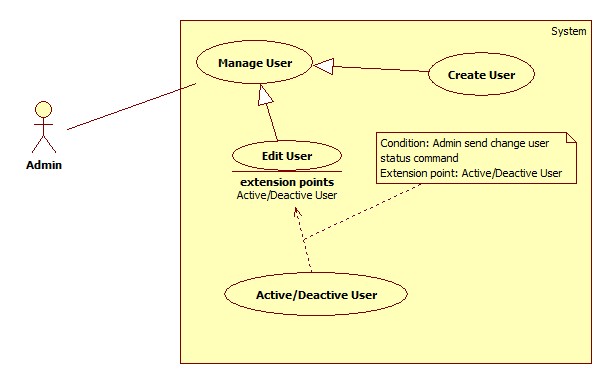
##### <StoreOwner> Use case

**Use Case Specification**

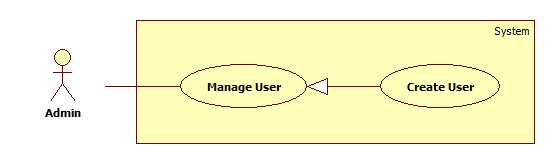
##### < StoreOwner > Use case

**Use Case Specification**

#### <Admin> Overview Use Case

****

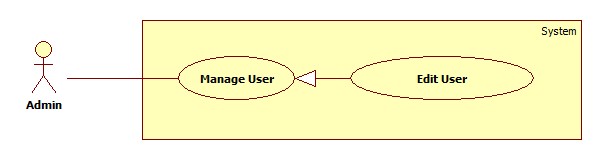
##### < Admin > Create User Use case



**Use Case Specification**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE –** HPS001 | | | |
| **Use Case No.** | HPS001 | Use Case Version | 2.0 |
| **Use Case Name** | Create User | | |
| **Author** | Nguyen Huy Hoang | | |
| **Date** | August 7, 2015 | Priority | High |
| **Actor**:   * Admin   **Summary**:   * This use case allows admin add new user.   **Goal**:   * New user is added to database.   **Triggers**:   * Admin send “add new user” command.   **Preconditions**:   * Guest must login with role Admin.   **Post Conditions**:   * Success: New user is inserted to database. * Fail: Show an error message, nothing is changed in database.   **Main Success Scenario**:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Admin send “add new user” command. | System pop-up for admin input include the following information:   * “Tên Đăng Nhập”: free text input, min length: 6, max length: 50, required. * “Mật Khẩu”: free text input, min length: 8, max length: 20, required. * “Điện Thoại: free text input, format: number, * “Email”: free text input, regular expression: “^[a-zA-Z0-9\_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+$". * “Tên”: free text input. Max length : 50. * “Đia Chỉ”: free text input. * “Tài Khoản Paypal”: free text input. * “Quyền Hạn”: select option include: “Chủ Cửa Hàng” and “Thành Viên”, required. * A command to finish action. | | 2 | Admin input information and send command to finish add new user.  [Alternative 1] | [Exception 1,2]  Add new user to database and show list new user. |   **Alternative Scenario**:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Admin dismiss pop-up. | Nothing to change. |   **Exceptions**:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User not fills all required field. | Show message: “Vui lòng nhập thông tin vào nhưng ô bắt buộc.” | | 2 | User invalid format | Show message: “Vui lòng nhập đúng thông tin”. |   **Relationships**: N/A  **Business Rules**:   * Status of new user is set to “active”. | | | |

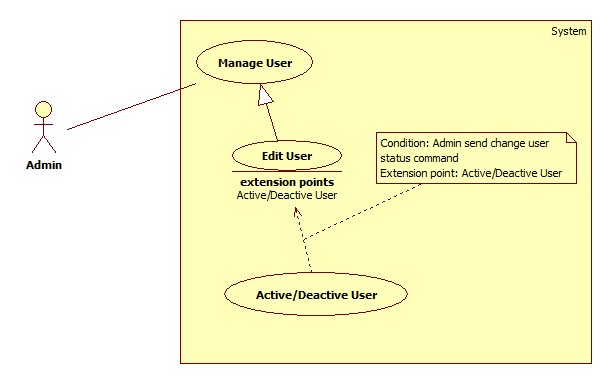
##### < Admin > Edit User Use case



**Use Case Specification**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE –** HPS002 | | | |
| **Use Case No.** | HPS002 | Use Case Version | 2.0 |
| **Use Case Name** | Edit User | | |
| **Author** | Nguyen Huy Hoang | | |
| **Date** | August 7, 2015 | Priority | High |
| **Actor**:   * Admin   **Summary**:   * This use case allows admin edit user information.   **Goal**:   * User information is updated to database.   **Triggers**:   * Admin send “edit user” command.   **Preconditions**:   * Guest must login with role Admin.   **Post Conditions**:   * Success: User information is updated to database. * Fail: Show an error message, nothing is changed in database.   **Main Success Scenario**:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Admin send “edit user” command. | System pop-up for admin input include the following information:   * “Điện Thoại: free text input, format: number, * “Email”: free text input, regular expression: “^[a-zA-Z0-9\_.+-]+@[a-zA-Z0-9-]+\.[a-zA-Z0-9-.]+$". * “Tên”: free text input. Max length : 50. * “Đia Chỉ”: free text input. * “Tài Khoản Paypal”: free text input. * A command to finish action. | | 2 | Admin change information and send command to finish edit user.  [Alternative 1] | [Exception 1]  Add new user to database and show list new user. |   **Alternative Scenario**:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | Admin dismiss pop-up. | Nothing to change. |   **Exceptions**:   |  |  |  | | --- | --- | --- | | No | Actor Action | System Response | | 1 | User invalid format | Show message: “Vui lòng nhập đúng thông tin”. |   **Relationships**: N/A  **Business Rules**:   * Admin can’t change username and password of user. * Old information of user will be filled. | | | |

##### < Admin > Active/Deactive User Use case



**Use Case Specification**

|  |  |  |  |
| --- | --- | --- | --- |
| **USE CASE –** HPS003 | | | |
| **Use Case No.** | HPS003 | Use Case Version | 2.0 |
| **Use Case Name** | Active/Deactive User | | |
| **Author** | Nguyen Huy Hoang | | |
| **Date** | August 7, 2015 | Priority | High |
| **Actor**:   * Admin   **Summary**:   * This use case allows admin change user status.   **Goal**:   * User status is updated to database.   **Triggers**:   * Admin send “change user status” command.   **Preconditions**:   * Guest must login with role Admin.   **Post Conditions**:   * Success: User status is updated to database. * Fail: Show an error message, nothing is changed in database.   **Main Success Scenario**:   |  |  |  | | --- | --- | --- | | Step | Actor Action | System Response | | 1 | Admin send “change user status” command. | System show new status of user. |   **Alternative Scenario**: N/A  **Exceptions**: N/A  **Relationships**:   * Extending Use-cases: Extend from Edit User (Admin send “change user status” command).   **Business Rules**:   * If current status of user is Active, change to Deactivate, and if current status of user is Deactivate, change to Active. | | | |

## Software System Attribute

### Usability

#### Graphic User Interface

* Texts and labels should be written in Vietnamese.

#### Usability

* Staff should need less than one week of training to use this system

#### Installation

* The system should be easy to deploy. Customer can deploy successfully and learn to configure, maintain the system within one day of training.
* The attached manual guide must be clear. Installation can be done by users themselves.

### Reliability

* Information about order of StoreOwner can be backup and restore easily using import/export function.

### Availability

* User need to have an internet connection to use every functions of the application in the best way.

### Security

* Each role of user has a specific permission to interact with system.

### Maintainability

* The system is divided into separated modules.

### Portability

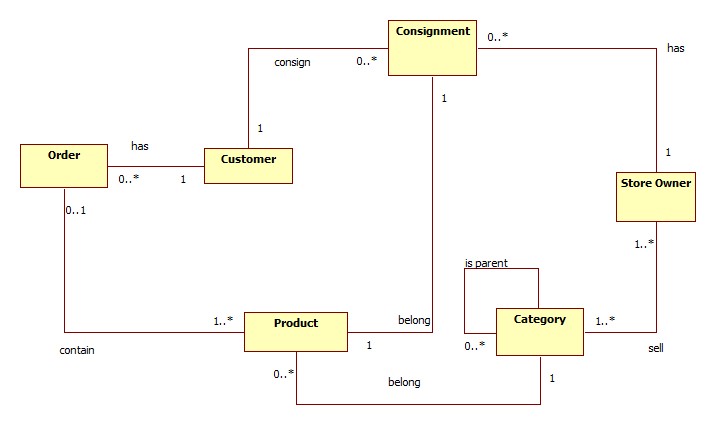
* The system can be deployed into many type of servers those have Tomcat 8 Server.

### Performance

* The process of manage consignment’s request should be within 1 minute.

## Conceptual Diagram

### Conceptual Diagram



**Data Dictionary**

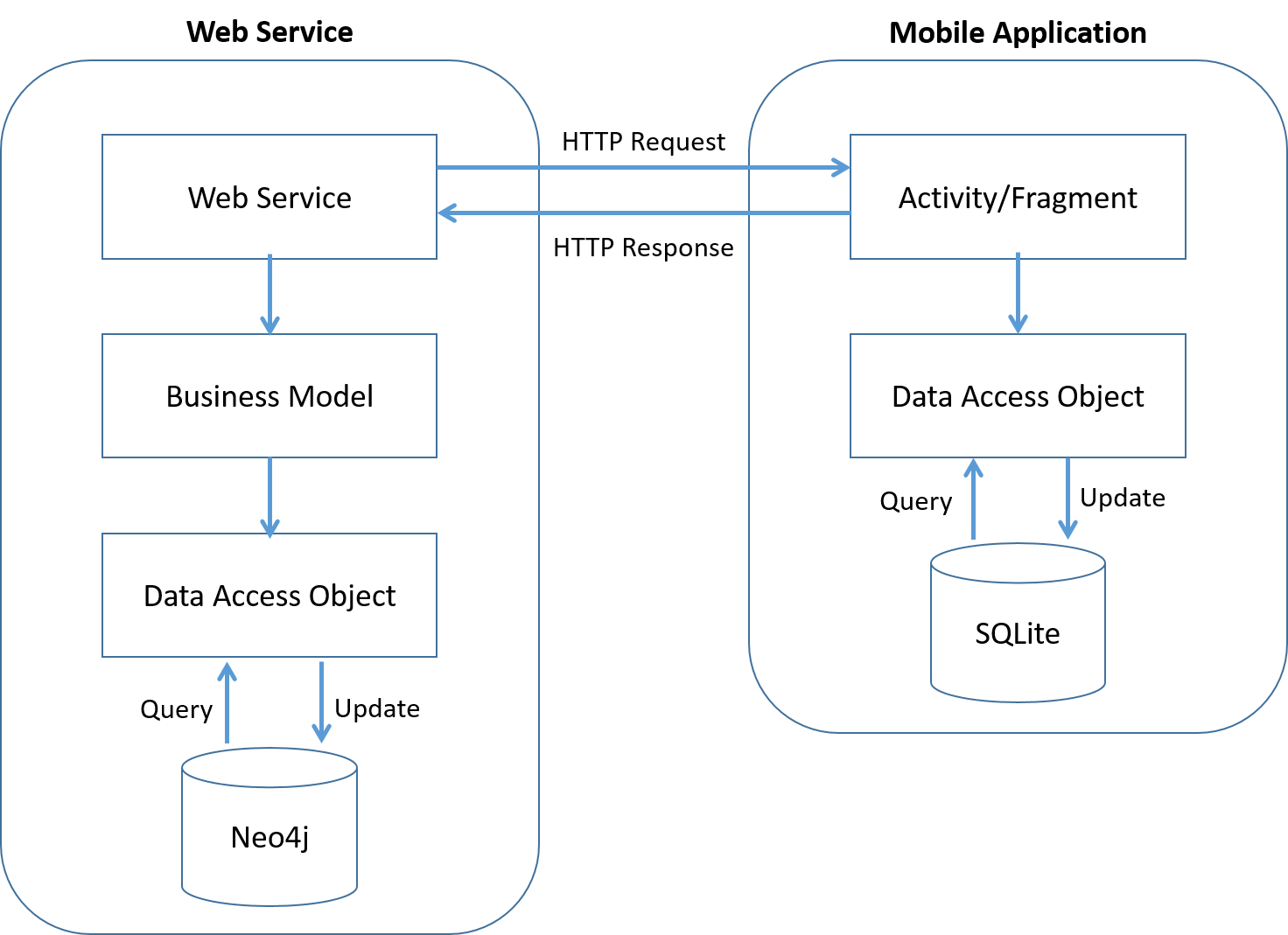
|  |  |
| --- | --- |
| **Entity Data dictionary: describe content of all entities** | |
| **Entity Name** | **Description** |
| Consignment | Describe all consignments in the system. |
| Customer | Describe all customers in the system. |
| Store Owner | Describe all store owners in the system. |
| Product | Describe all products in the system. |
| Category | Describe all product categories in the system. |
| Order | Describe all orders in the system. |

# Report No. 4 Software Design Description

## Design Overview

* This document part describes the details about technical and user interface design of HPS system. The document includes system architectural design, component diagram, detailed description, interaction diagram, user interface design and database design.
* The architectural design gives us the information of how HPS system is organized.
* The component diagram describes the architecture of HPS main component and sub-system.
* The detailed description describes the implement details of HPS system. This part include class diagram, class diagram explanation and interaction diagram.
* The user interface design give us the user graphic interface of HPS system. It describes how does HPS system look like.
* The database design show us how data of HPS system is stored and the relationship between entities in HPS database.

## System Architectural Design



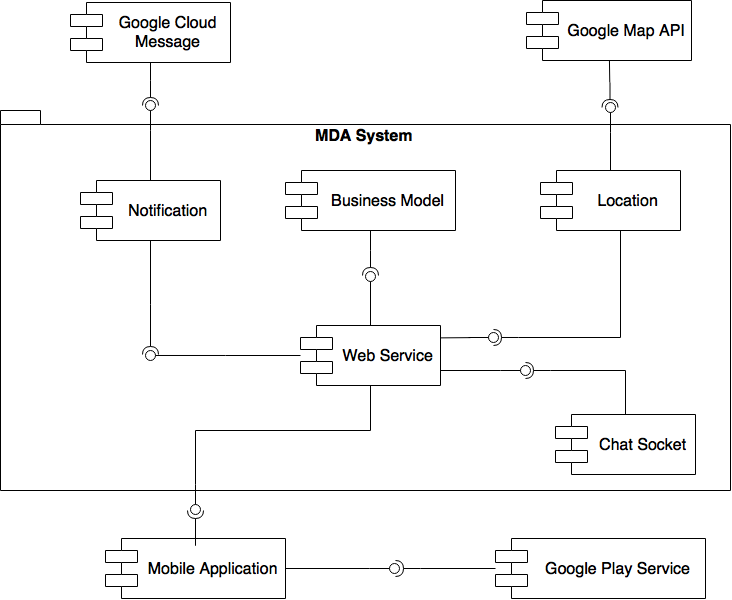
### Web Application Architecture Description

* The system is developed from MVC architecture style and customized with passive model to adapt to JAVA MVC, RESTful web service technologies.
* **Model** is a part of application which uses to storage and retrieval data.
* View is the part of the application that visualizes the data to help user **easily** to view, update or customize them.
* **Controller** is the parts of the application that acts like event handler to handles user interaction. Typically controller read data from a request and returns respective view for user.
* **Web** **Service** is the parts of the application which use to provide interface for mobile application to do some background operation such as get consignment, update consignment status…

### Mobile Application Architecture Description

* The application is developed as an Android native application. In general, the application architecture follow by Android architecture.
* **Activity** is basic core of an Android application that handles user input, call web service…
* **Data** **Transfer** Object is the parts of the application that mapping SQLite database to storage and retrieval the consignment’s orders which assigned for employee.

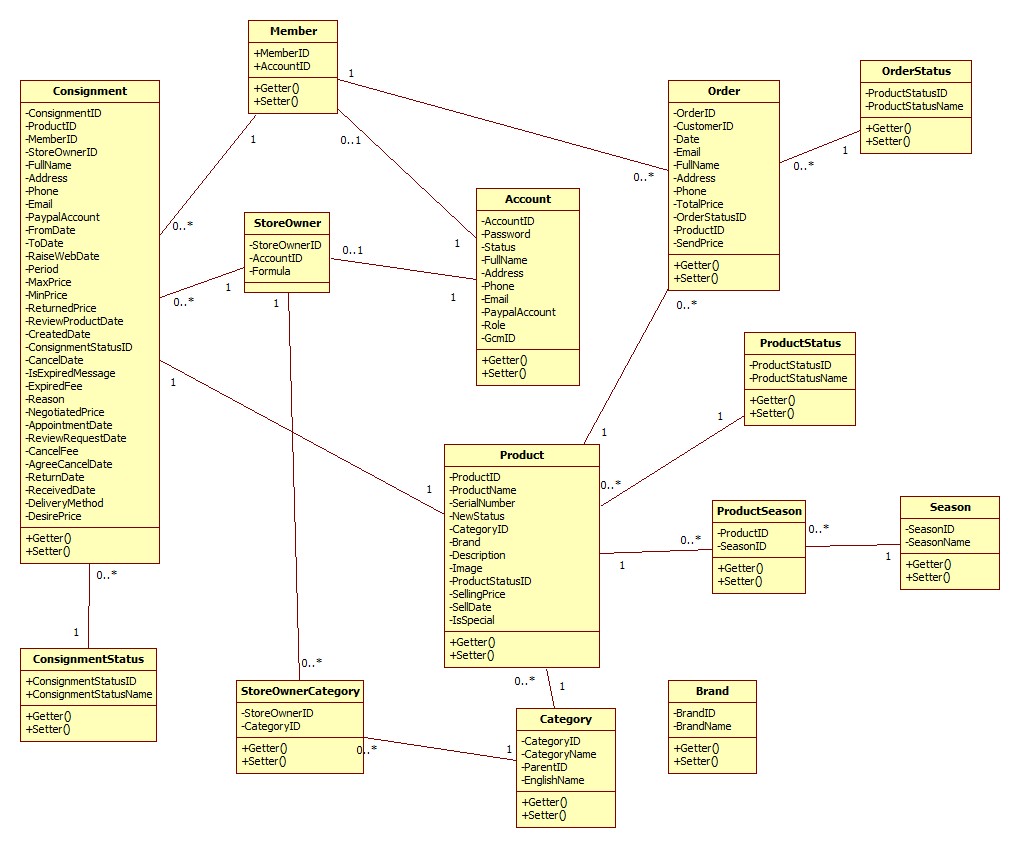
## Component Diagram



|  |  |
| --- | --- |
| **Component dictionary: describe component** | |
| **Component Name** | **Description** |
| Web Service | Content all web service of the system. |
| Business Model | Handle business process. |
| Notification | Use to send notification message to mobile. |
| Google Cloud Message | Use to push notification to mobile application. |
| Location | Use to process user location a place. |
| Google Map API | Use to work with Google Map API to get input data for Current Location and Current Place. |
| Chat Socket | Use to establish chat connection and send chat messages. |
| Mobile Application | Android Application. |
| Google Play Service | Use to process google map and google cloud message services. |

## Detailed Description

### Class Diagram



### Class Diagram Explanation

#### Consignment

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| ConsignmentID | String | Private | Unique identifier of Consignment |
| ProductID | int | Private | ID of Product |
| MemberID | int | Private | ID of Member |
| StoreOwnerID | int | Private | ID of StoreOwner |
| FullName | String | Private | Full name of customer |
| Address | String | Private | Address of customer |
| Phone | String | Private | Phone of customer |
| Email | String | Private | Email of customer |
| PaypalAccount | String | Private | Paypal account of customer |
| FromDate | DateTime | Private | Date Store Owner can receive |
| ToDate | DateTime | Private | The last date Store Owner can receive |
| RaiseWebDate | DateTime | Private | Date publish product on website |
| Period | Integer | Private | Period of consignment |
| MaxPrice | float | Private | Max price suggest to customer |
| MinPrice | float | Private | Min price suggest to customer |
| ReturnedPrice | float | Private | Price return to customer |
| ReviewProductDate | DateTime | Private | Date receive or refuse Product |
| CreatedDate | DateTime | Private | Date create |
| ConsignmentStatusID | int | Private | ID of ConsignmentStatus |
| CancelDate | DateTime | Private | Date Customer request cancel consignment |
| IsExpiredMessage | int | Private | Send message expired consignment |
| ExpiredFee | float | Private | Fee of expired consignment |
| Reason | String | Private | Reason of canceled consignment |
| NegotiatedPrice | float | Private | Negotiated price |
| AppointmentDate | DateTime | Private | Appointment date to receive product |
| ReviewRequestDate | DateTime | Private | Date accept or refuse consignment request |
| CancelFee | float | Private | Fee of cancel consignment |
| AgreeCancelDate | DateTime | Private | Date StoreOwner agree request cancel consignment |
| ReturnDate | DateTime | Private | Date return product to customer |
| ReceivedDate | DateTime | Private | Receive product Date |
| DeliveryMethod | int | Private | Method to delivery product |
| DesirePrice | float | Private | Desire price of customer |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute Type | Public | Get attribute value |

#### Product

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| ProductID | int | Private | Unique identifier of Product |
| ProductName | String | Private | Name of Product |
| SerialNumber | String | Private | Serial number of Product |
| NewStatus | int | Private | Product new status |
| CategoryID | int | Private | ID of Category |
| Brand | String | Private | Brand of Product |
| Description | String | Private | Description of Product |
| Image | String | Private | Image of Product |
| ProductStatusID | int | Private | ID of ProductStatus |
| SellingPrice | float | Private | Selling Price |
| SellDate | DateTime | Private | Selling Date |
| IsSpecial | int | Private | Is special product |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | Void | Public | Set value of attribute |

#### Order

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| OrderID | int | Private | Unique identifier of Order |
| CustomerID | int | Private | ID of Customer |
| Date | DateTime | Private | Order date |
| Email | String | Private | Email of customer |
| FullName | String | Private | Full name of customer |
| Address | String | Private | Address of customer |
| Phone | String | Private | Phone of Product |
| TotalPrice | float | Private | Price of Order |
| OrderStatusID | int | Private | ID of OrderStatus |
| ProductID | int | Private | ID of Product |
| SendPrice | int | Private | Price is sent to customer |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | Void | Public | Set value of attribute |

#### Account

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| AccountID | String | Private | Unique identifier of Account |
| Password | String | Private | Password of user |
| Status | String | Private | Status of account |
| FullName | String | Private | Full name of user |
| Address | String | Private | Address of user |
| Phone | String | Private | Phone of user |
| Email | String | Private | Email of user |
| PaypalAccount | String | Private | Paypal account of user |
| Role | String | Private | Role of user |
| GcmID | String | Private | Google cloud message to receive notification on mobile application. |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | Void | Public | Set value of attribute |

#### Store Owner

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| StoreOwnerID | int | Private | Unique identifier of StoreOwner |
| AccountID | String | Private | AccountID of StoreOwner |
| Formula | float | Private | Formula to calculate consignment price suggestion |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | Void | Public | Set value of attribute |

#### Member

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| MemberID | int | Private | Unique identifier of Member |
| AccountID | String | Private | AccountID of Member |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | Void | Public | Set value of attribute |

#### StoreOwnerCategory

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| StoreOwnerID | int | Private | Unique identifier of StoreOwnerCategory |
| CategoryID | String | Private | Unique identifier of StoreOwnerCategory |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | Void | Public | Set value of attribute |

#### ConsignmentStatus

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| ConsignmentStatusID | int | Private | Unique identifier of ConsignmentStatus |
| ConsignmentStatusName | String | Private | Name of consignment status |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | Void | Public | Set value of attribute |

#### ProductStatus

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| ProductStatusID | int | Private | Unique identifier of ProductStatus |
| ProductStatusName | String | Private | Name of product status |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | Void | Public | Set value of attribute |

#### OrderStatus

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| OrderStatusID | int | Private | Unique identifier of OrderStatus |
| OrderStatusName | String | Private | Name of order status |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | Void | Public | Set value of attribute |

#### ProductSeason

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| ProductID | int | Private | Unique identifier of ProductSeason |
| SessonID | int | Private | Unique identifier of ProductSeason |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | Void | Public | Set value of attribute |

#### Season

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| SeasonID | int | Private | Unique identifier of Season |
| SessonName | String | Private | Name of Season |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | Void | Public | Set value of attribute |

#### Category

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| CategoryID | int | Private | Unique identifier of Category |
| CategoryName | String | Private | Name of Category |
| ParentID | int | Private | ID of parent category |
| EnglishName | String | Private | English name to search product on amazon API |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | Void | Public | Set value of attribute |

#### Brand

Attribute

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| BrandID | int | Private | Unique identifier of Brand |
| BrandName | String | Private | Name of Brand |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | Void | Public | Set value of attribute |

|  |  |  |  |
| --- | --- | --- | --- |
| **Attribute** | **Type** | **Visibility** | **Description** |
| DatingMen | Boolean | Private | Wishes dating person is men or not. |
| DatingWomen | Boolean | Private | Wishes dating person is women or not. |
| DatingAge | String | Private | Age of wishes dating person. |

Method

|  |  |  |  |
| --- | --- | --- | --- |
| **Method** | **Return type** | **Visibility** | **Description** |
| Getter | Attribute type | Public | Get attribute value |
| Setter | Void | Public | Set value of attribute |

### Interaction Diagram

#### Sequence Diagram

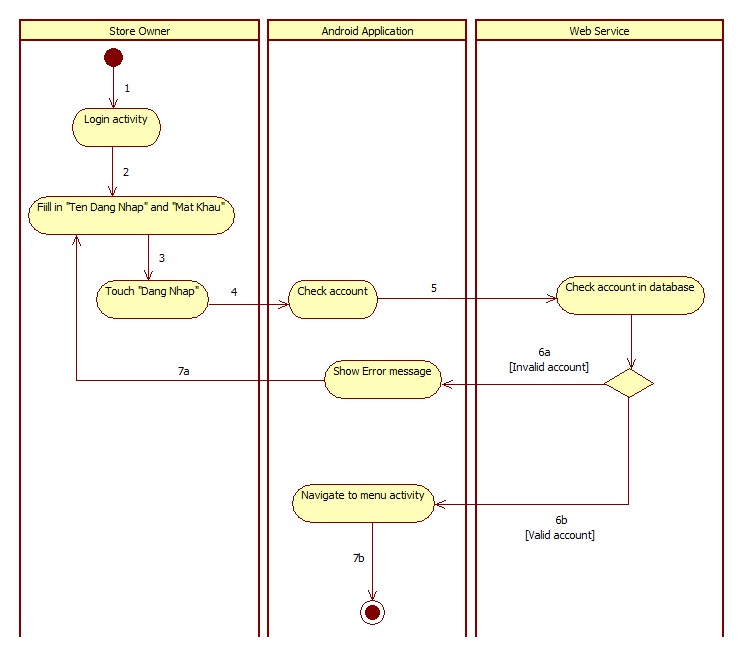
**Summary:** This diagram shows how Guest can login into the application with system account.

#### Activity Diagram

**Summary:** This diagram shows how Guest can login into the

##### <StoreOwner> Login

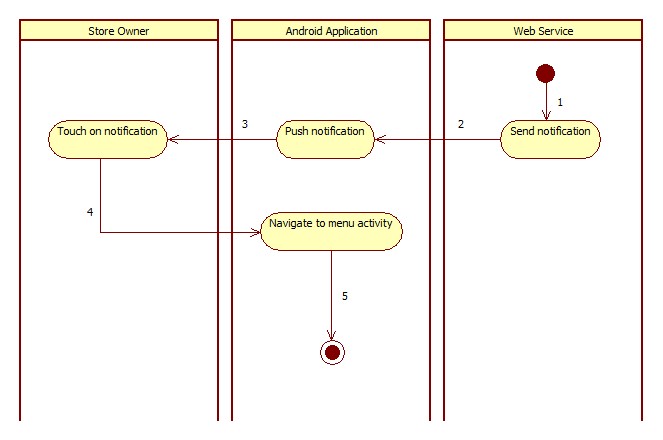
**Summary:** This diagram shows how Store Owner login into system

****

|  |  |  |
| --- | --- | --- |
| 1. Start | 2. Store Owner touch “Đăng Xuất” tab | 3. Show confirm dialog |
| 4a. Store Owner touch “Hủy” button | 4b. Store Owner touch “Xác Nhận” button | 5a. Dismiss dialog |
| 5b. call logout() and navigate to login activity | 6a. Finish | 6b. Finish |

##### <StoreOwner> Receive Nofitication

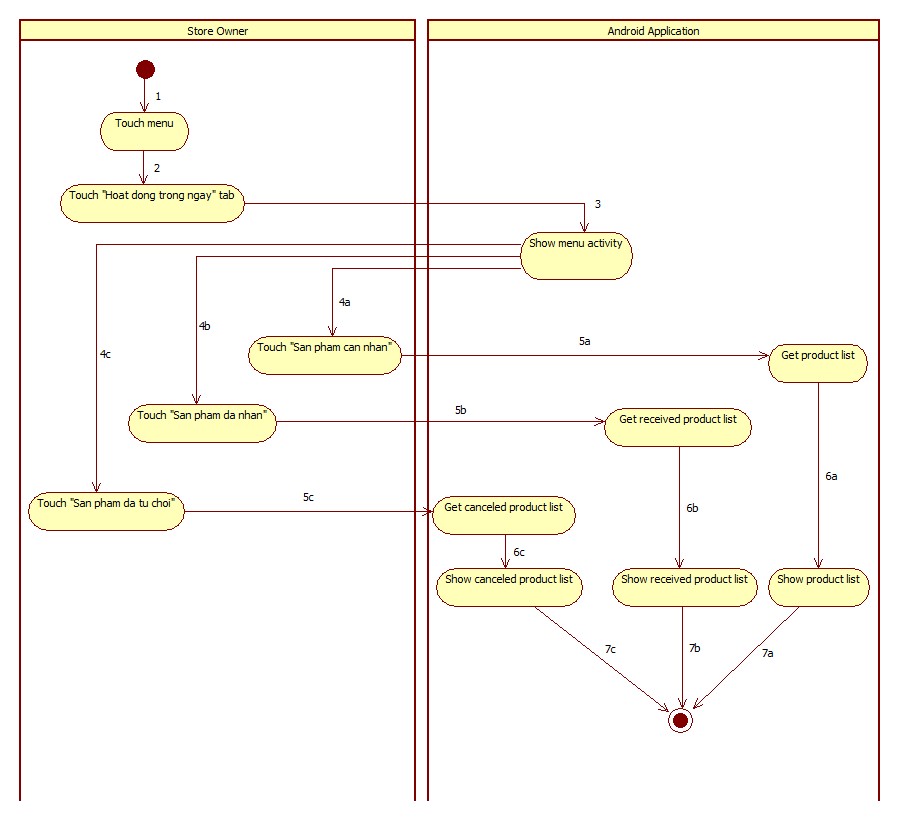
**Summary:** This diagram shows how Store Owner receive notification

****

|  |  |  |
| --- | --- | --- |
| 1. Send Notification | 2. Push Notification | 3. Store Owner touch on notification |
| 4. Navigate to menu activity | 5. Finish |  |

##### <StoreOwner> View Task

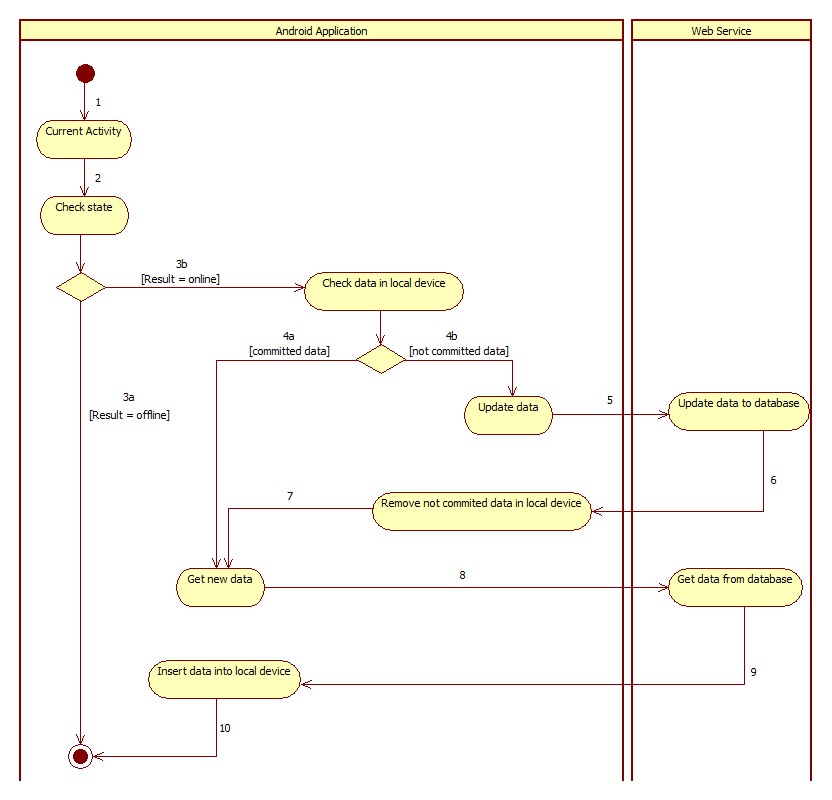
**Summary:** This diagram shows how Store Owner receive notification

****

|  |  |  |
| --- | --- | --- |
| 1. Start | 2. Touch tab | 3. Call menu activity |
| 4a. Store Owner touch “Sản phẩm cần nhận” | 4b. Store Owner touch “Sản phẩm đã nhận” | 4b. Store Owner touch “Sản phẩm đã từ chối” |
| 5a. Call getProduct() | 5b. Call getReceivedProduct() | 5c. Call getCanceledProduct() |
| 6a. Show product list | 6b. Show received product list | 6c. Show canceled product list |
| 7a. Finish | 7b. Finish | 7c. Finish |

##### <StoreOwner> Auto Synchronize

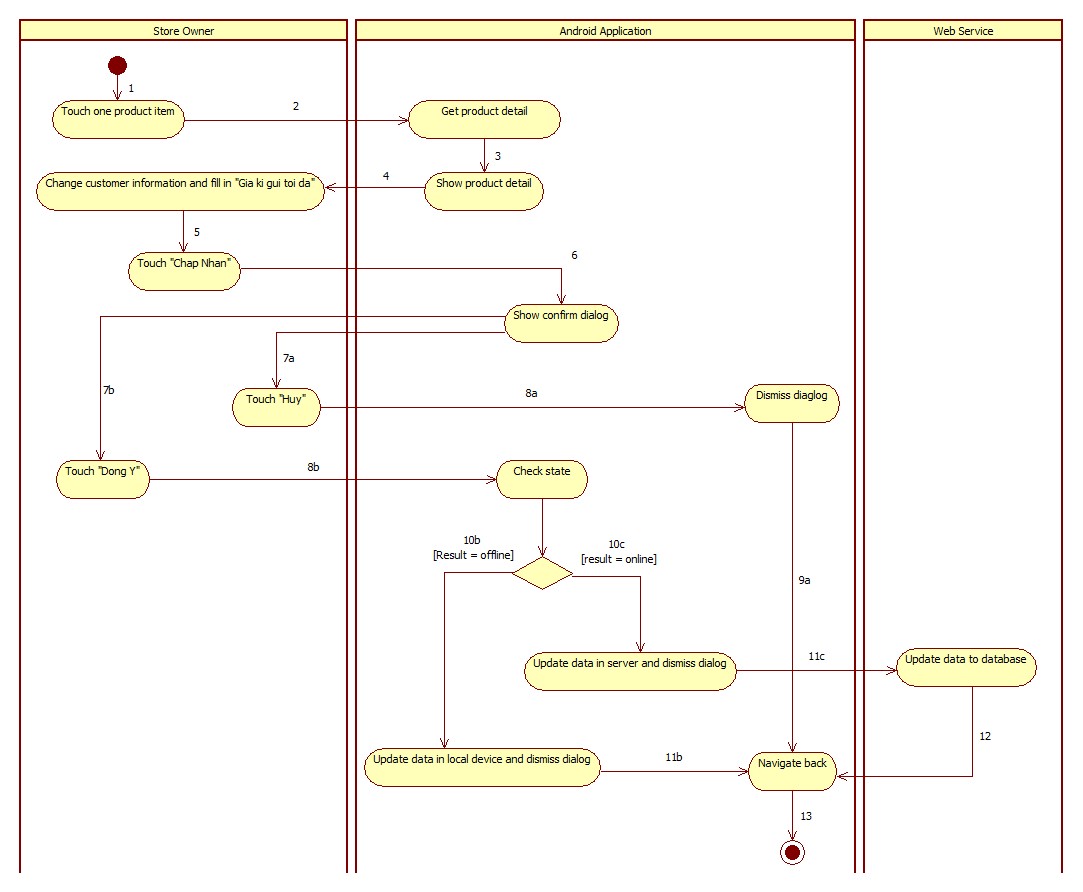
**Summary:** This diagram shows how Mobile Application auto synchronize data.

****

|  |  |  |
| --- | --- | --- |
| 1. Start | 2. Call checkState() | 3a. Finish |
| 3b. Check data in local device, if exist data have not committed, update to server | 4a. Call getNewData() | 4b. call updateData() |
| 5. Send data to web service | 4. Remove not committed data | 7. Call getNewData() |
| 8. Get data from web service | 9. Insert data into local device | 10. Finish |

##### <StoreOwner> Accept Product

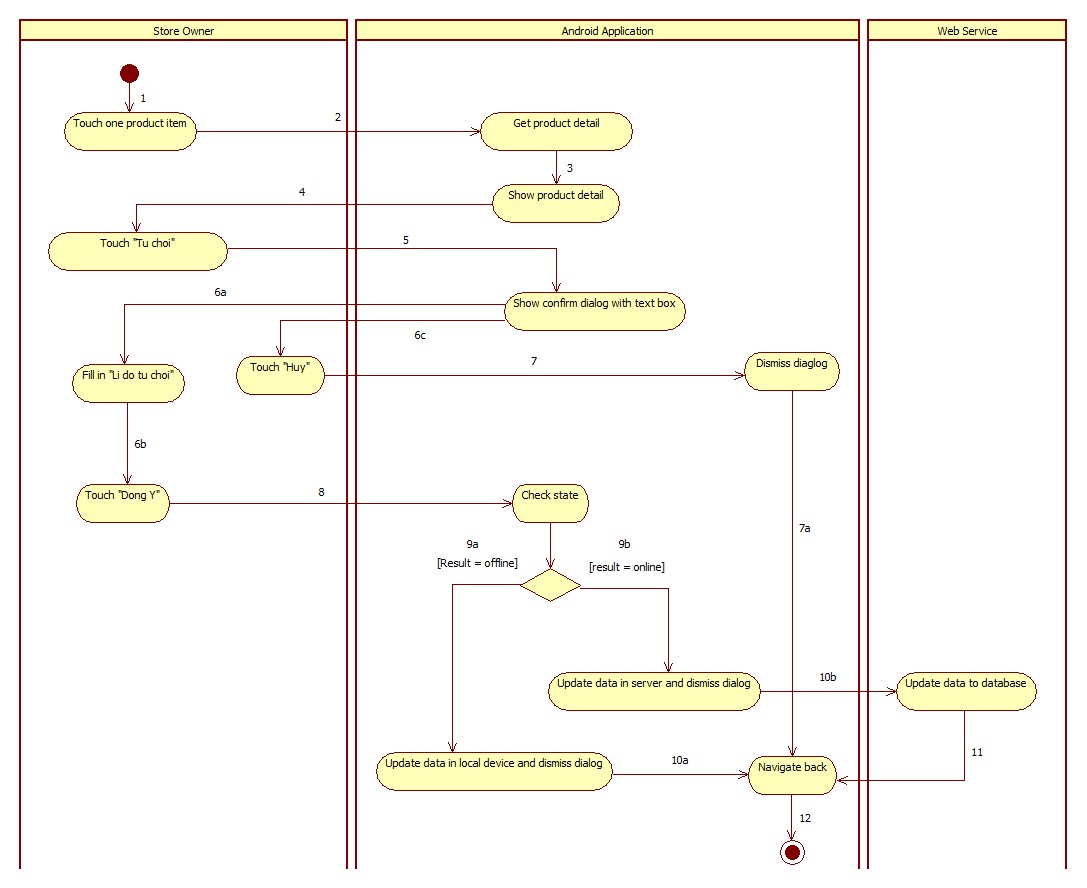
**Summary:** This diagram shows how StoreOwner receive product.



|  |  |  |
| --- | --- | --- |
| 1. Start | 2. Call getProductDetail() | 3. Show product detail |
| 4. Store Owner change customer information and enter “Giá kí gửi tối đa” | 5. Store Owner touch “Chấp Nhận” button | 6. Show confirm dialog |
| 7a. Store Owner touch “Hủy” button to cancel action | 7b. Store Owner touch “Đồng ý” button | 8a. Dismiss confirm dialog |
| 8b. Call checkState() | 9a. Navigate back | 10b. Update data in local device and dismiss dialog |
| 10c. Call service to update data in server and dismiss dialog | 11b. Navigate back | 11c. Send data to web service |
| 12. Navigate back | 13. Finish |  |

##### <StoreOwner> Cancel Product

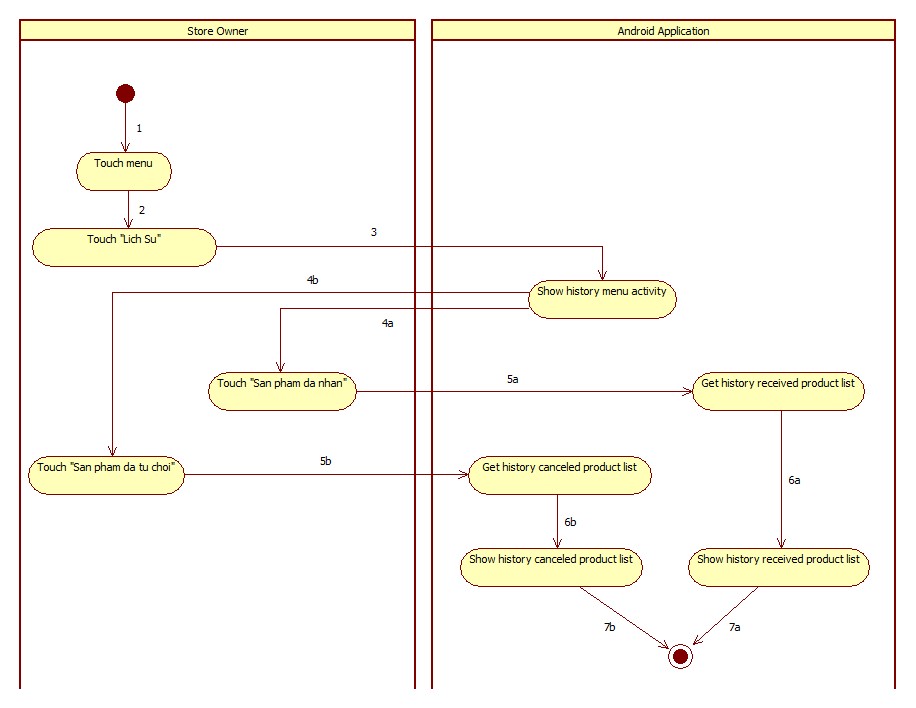
**Summary:** This diagram shows how StoreOwner cancel product.



|  |  |  |
| --- | --- | --- |
| 1. Start | 2. Call getProductDetail() | 3. Show product detail |
| 4. Store Owner touch “Từ chối” button | 5. Show confirm dialog with text box | 6a. Store Owner enter “Lí do từ chối”. |
| 6b. Store Owner touch “Đồng ý” button. | 6c. Store Owner touch “Hủy” button to cancel action. | 7. Dismiss confirm dialog |
| 7a. Navigate back | 8. Call checkState() | 9a. Update data in local device and dismiss dialog. |
| 9b. Call service to update data in server and dismiss dialog. | 10a. Navigate back | 10b. Send data to web service |
| 11. Navigate back | 12. Finish |  |

##### <StoreOwner> View History

**Summary:** This diagram shows how StoreOwner view history

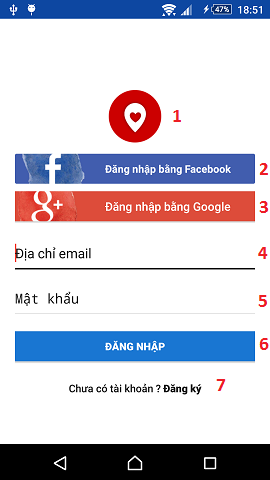
****

|  |  |  |
| --- | --- | --- |
| 1. Start | 2. Touch tab | 3. Call history menu activity |
| 4a. Store Owner touch “Sản phẩm đã nhận” | 4b. Store Owner touch “Sản phẩm đã từ chối” | 5a. Call getHistoryReceivedProduct() |
| 5c. Call getHistoryCanceledProduct() | 6a. show history received product list | 6b. show history canceled product list |
| 7a. Finish | 7b. Finish |  |

## User Interface Design

### Guest Interface Design

#### Login



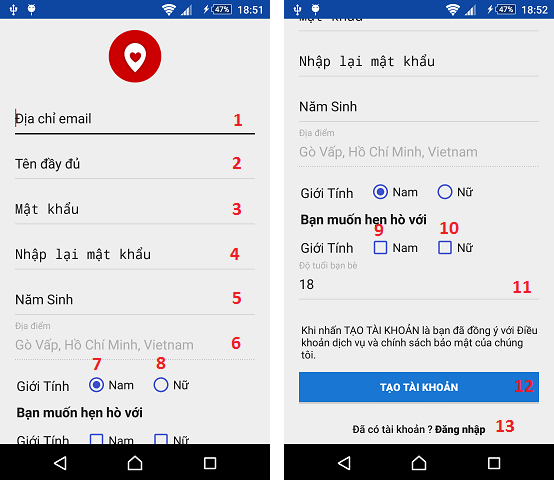
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Logo | Logo of the system | Yes | No | Image | Image | N/A |
| 4 | Username | Fill username | No | Yes | Text box | String | N/A |
| 5 | Password | Fill password | No | Yes | Password | String | N/A |

**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 2 | Sign in | Log-in into application with Facebook account | N/A | Navigate to Facebook login screen. |
| 3 | Sign in | Log-in into application with Google+ account | N/A | Show account dialog. |
| 6 | Sign in | Log-in into application with system account | N/A | Navigate to main screen. |
| 7 | Register | Touch to go to register screen | N/A | Navigate to register screen. |

#### Register



**Fields**

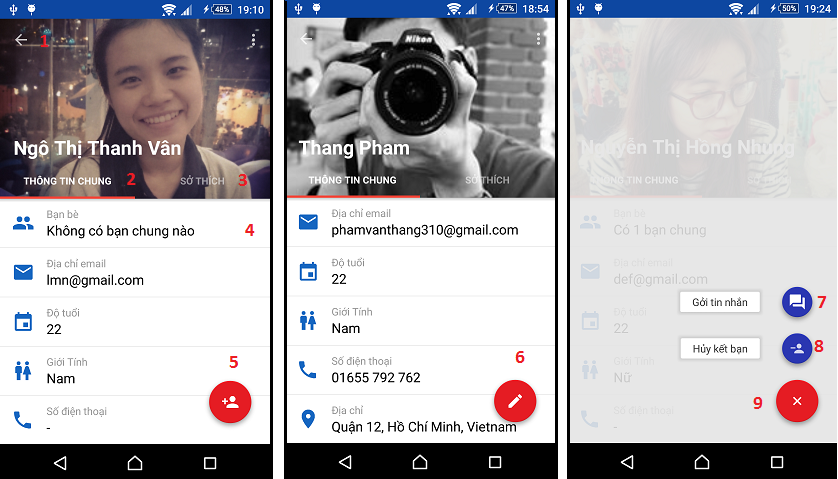
|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 1 | Email | Fill email address | No | Yes | Textbox | String | N/A |
| 2 | Full name | Fill full name | No | Yes | Textbox | String | N/A |
| 3 | Password | Fill password | No | Yes | Password | String | N/A |
| 4 | Confirm password | Fill confirm password | No | Yes | Password | String | N/A |
| 5 | Birthdate | Fill date of birth | No | Yes | Date time | String | N/A |
| 6 | Address | Fill address | Yes | Yes | Textbox | String | N/A |
| 7 | Male gender | Choose gender is male | No | Yes | Radio button | Boolean | N/A |
| 8 | Female gender | Choose gender is female | No | Yes | Radio button | Boolean | N/A |
| 9 | Male friend | Choose dating male | No | No | Check box | Boolean | N/A |
| 10 | Female friend | Choose dating female | No | No | Check box | Boolean | N/A |
| 11 | Friend age | Fill friend age | No | Yes | Textbox | String | N/A |

**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 12 | Register | Touch to register new account | N/A | Navigate to register interest |
| 13 | Login | Touch to move to login page | N/A | Navigate back to login page |

### User Interface Design

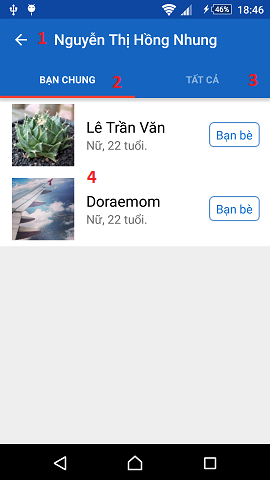
#### View Profile



**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Back | Touch to back to previous screen | N/A | Navigate to previous screen |
| 2 | Personal information tab | Touch to view personal information | N/A | Navigate to personal info tab |
| 3 | Hobby information tab | Touch to view interest information | N/A | Navigate to interest info tab |
| 4 | Mutual friend | Touch to view mutual friend | N/A | Navigate to mutual friend screen |
| 5 | Friend request | Touch to send friend request | N/A | Send friend request to this user |
| 6 | Edit profile | Touch to edit profile | N/A | Navigate to edit profile screen |
| 7 | Send message | Touch to send message | N/A | Navigate to chat screen |
| 8 | Unfriend | Touch to unfriend | N/A | Show unfriend confirmation dialog |
| 9 | Show button | Touch to show additional button | N/A | Show two action button |

#### View Friend Details



**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Back | Touch to go back previous screen | N/A | Navigate to profile info screen |
| 2 | View mutual friend | Touch to show mutual friend | N/A | Navigate to mutual friend list |
| 3 | View all Friend | Touch to show all friend | N/A | Navigate to all friend list |
| 4 | View profile | Touch to view person profile | N/A | Navigate to profile screen |

#### Change Password



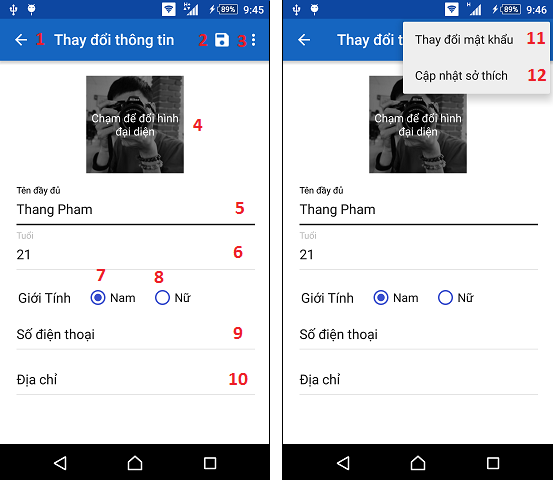
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 4 | New password | Fill new password | No | Yes | Password | String | N/A |
| 5 | Confirm password | Fill confirmation password | No | Yes | Password | String | N/A |

**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Back | Touch to go back previous screen | N/A | Navigate to profile info screen |
| 2 | Save | Touch to save new information | N/A | Update new password |
| 3 | More options | Touch to show more update options | N/A | Show list menu options |

#### Update Profile



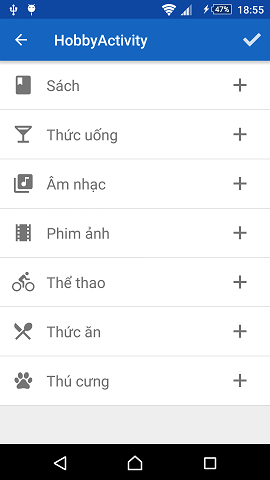
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 5 | Full name | Fill full name | No | Yes | Textbox | String | N/A |
| 6 | Birthdate | Fill date of birth | No | Yes | Date time | String | N/A |
| 7 | Male gender | Choose gender is male | No | Yes | Radio button | Boolean | N/A |
| 8 | Female gender | Choose gender is female | No | Yes | Radio button | Boolean | N/A |
| 9 | Address | Fill address | No | No | Textbox | String | N/A |
| 10 | Mobile phone | Fill mobile phone number | No | No | Textbox | String | N/A |

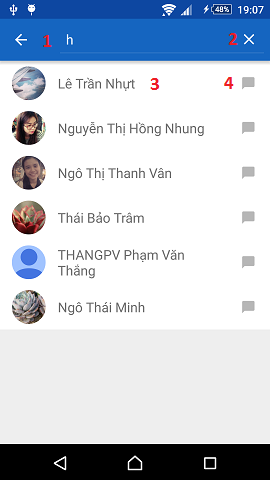
**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Back | Touch to go back previous screen | N/A | Navigate to profile info screen |
| 2 | Save | Touch to save changed information | N/A | Navigate to main screen. |
| 3 | More options | Touch to show more update options | N/A | Show list menu options |
| 4 | Change avatar | Touch to change avatar | N/A | Navigate to photo gallery |
| 11 | Change password | Touch to change password | N/A | Navigate to change password screen |
| 12 | Change interest | Touch to change interest | N/A | Navigate to change interest screen |

#### Update Interest



#### Find Friend



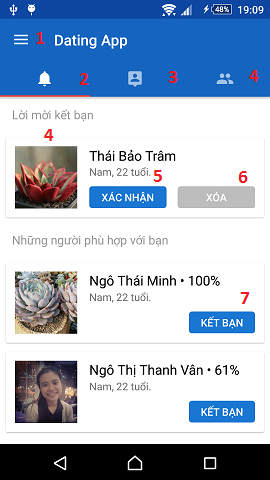
**Fields**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **No** | **Field Name** | **Description** | **Read only** | **Mandatory** | **Control Type** | **Data Type** | **Length** |
| 2 | New password | Fill new password | No | Yes | Password | String | N/A |

**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Back | Touch to go back previous screen | N/A | Navigate to list friend screen |
| 3 | View profile | Touch to view person’s profile | N/A | Navigate to profile screen |
| 4 | Chat | Touch to chat | N/A | Navigate to chat screen |

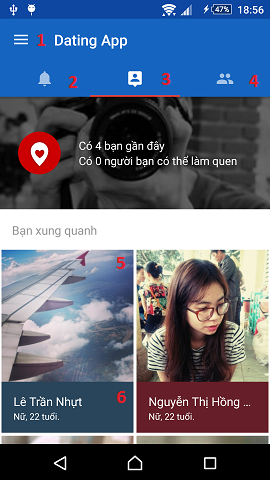
#### View Suggested Friend



**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Navigation drawer | Swipe to show left menu drawer | N/A | Show menu drawer |
| 2 | Notification tab | Touch to choose notification tab | N/A | Navigate to notification screen |
| 3 | Surround tab | Touch to choose surround person | N/A | Navigate to surrounded people screen |
| 4 | Friend tab | Touch to choose friend list | N/A | Navigate to friend list screen |
| 5 | Accept friend request | Touch to accept friend request | N/A | Show accept friend message |
| 6 | Deny friend request | Touch to deny friend request | N/A | Show deny friend message |
| 7 | Send friend request | Touch to send friend request | N/A | Show send friend request message |

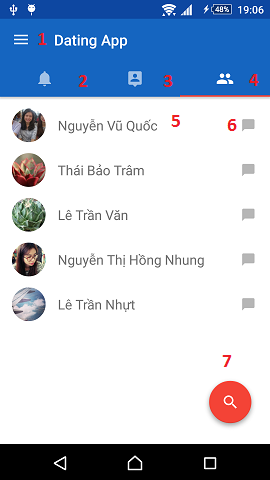
#### View Surrounded People



**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Navigation drawer | Swipe to show left menu drawer | N/A | Show menu drawer |
| 2 | Notification tab | Touch to choose notification tab | N/A | Navigate to notification screen |
| 3 | Surround tab | Touch to choose surround person | N/A | Navigate to surrounded people screen |
| 4 | Friend tab | Touch to choose friend list | N/A | Navigate to friend list screen |
| 5 | Surrounded people | Touch to show specify location on map | N/A | Navigate to map screen |
| 6 | Profile | Touch to show person’s profile | N/A | Navigate to profile screen |

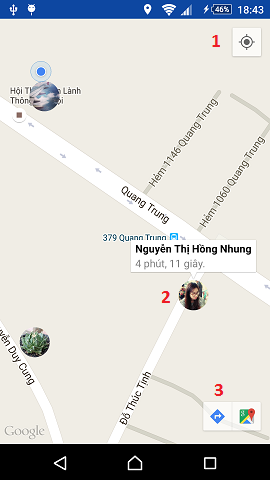
#### View List Friend



**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Navigation drawer | Swipe to show left menu drawer | N/A | Show menu drawer |
| 2 | Notification tab | Touch to choose notification tab | N/A | Navigate to notification screen |
| 3 | Surround tab | Touch to choose surround person | N/A | Navigate to surrounded people screen |
| 4 | Friend tab | Touch to choose friend list | N/A | Navigate to friend list screen |
| 5 | View profile | Touch to view person’s profile | N/A | Navigate to profile screen |
| 6 | Chat | Touch to chat | N/A | Navigate to chat screen |
| 7 | Search | Touch to search | N/A | Navigate to search screen |

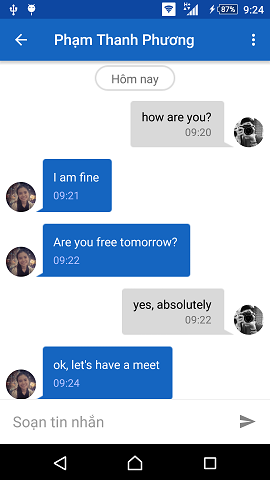
#### View Location



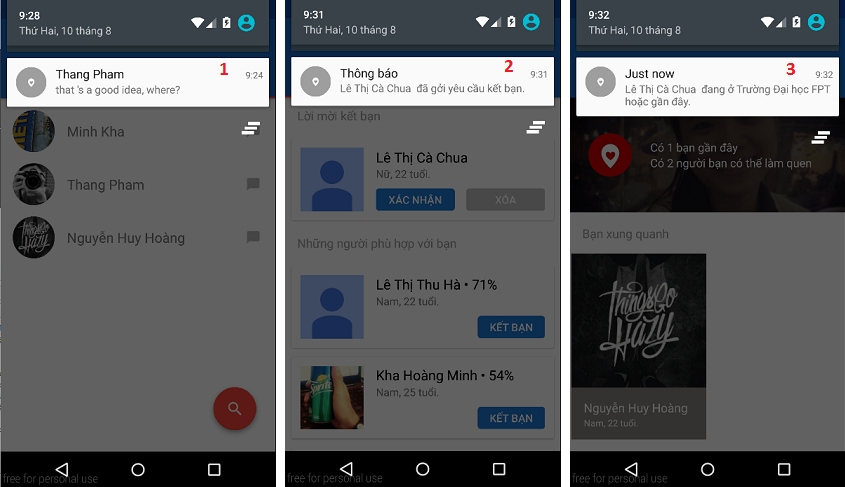
**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Current location | Touch to move to user current location | N/A | Move to user current location in map |
| 2 | Friend location | Touch to move to friend location and view information | N/A | Move to friend location in map |
| 3 | Navigation | Touch to navigate from user location to friend location | N/A | Navigate to google map application. |

#### Chat Message



#### Notification



**Button/Hyperlinks**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **No** | **Function** | **Description** | **Validation** | **Outcome** |
| 1 | Chat notification | Touch to navigate to chat screen | N/A | Navigate to chat screen |
| 2 | Friend notification | Touch to navigate to notification screen | N/A | Navigate to notification screen |
| 3 | Location notification | Touch to navigate to map screen | N/A | Navigate to map screen |

Reference:

1. Tech in Asia, September 2014, Viet Nam’s smartphone users number *<https://www.techinasia.com/oops-vietnam-22-million-smartphone-users-33-million>*
2. The Guardian, February 2015, location-based dating app user percentage *<http://www.theguardian.com/technology/2015/feb/17/mobile-dating-apps-tinder-two-thirds-men>*
3. Sun Microsystems, 1999, Java Style Coding Convention <http://www.oracle.com/technetwork/java/codeconventions-150003.pdf>