**CPSC-5360 Software Engineering**

Spring 2016

Instructor - Stefan Andrei

**Project Report**

**The Simulation of an Online Purchasing of a Product**

Due on May 08, 2016

**Submitted by**

Tapaskumar Patel (L20375097)

Bhumit Patel (L20382154)

Krupaliben Desai (L20375840)

Gargi Thakker (L20371763)

**Acknowledgement:**

We would like to express my deepest appreciation to all those who provided us the possibility to complete this report. We extend a special gratitude to Dr Stefan Andrei, professor of Computer Science department at Lamar University, whose contribution in stimulating suggestions and encouragement helped us to coordinate this project; especially in writing this report.

The supervision and support that he gave truly helped in the progression and smoothness of the project. Without his help we could not have presented this progress report up to the present standard.

Date: 05/08/2016

Tapaskumar Patel (L20375097)

Bhumit Patel (L20382154)

Krupaliben Desai (L20375840)

Gargi Thakker (L20371763)

**Abstract:**

The main objective of the project is to develop a system that customer to buy mobiles, books, accessories from anywhere through online. This application advertises some of the products for shopping. To buy products, customer has to create an account. Those who does not have an account, they can only view the available product. They can’t buy it. Once the customer has created account, not only he can view the products, he can also add the product to the cart and also he can place an order to buy those products. This application can be designed from scrape using client-side languages, such as JavaScript and HTML, combined with the server-side Java language through Java Server page. The server side, mostly Java, contains all the implementation related to setting up the database, creating session models for joining different user-interface (UI) pages. It is responsible for taking information from the database and making it available to the UI by mapping the category or item ID to the respective IDs stored in the database. The client side is responsible for showing the entire user interface, containing the CSS, HTML, and JavaScript.

Table of Contents

[**1.** **INTRODUCTION** 5](#_Toc450505915)

[**2.** **PROJECT OVERVIEW** 6](#_Toc450505916)

[2.1 USE CASE DIAGRAM 9](#_Toc450505917)

[2.1.1 USE CASE DIAGRAM FOR SERVER: 9](#_Toc450505918)

[2.1.2 USE CASE DIAGRAM FOR CUSTOMER: 10](#_Toc450505919)

[2.2 DOMAIN MODEL: 11](#_Toc450505920)

[2.3 SEQUENCE DIAGRAMS: 12](#_Toc450505921)

[2.4 CLASS DIAGRAM: 13](#_Toc450505922)

[**3.** **IMPLEMENTATION** 14](#_Toc450505923)

[**4.** **TESTING MODELS** 18](#_Toc450505924)

[4.1 Testing Types 18](#_Toc450505925)

[4.2 Testing Methods 18](#_Toc450505926)

[**5.** **LANGUAGE AND TECHNOLOGIES USED FOR FRONT END AND BACK END** 19](#_Toc450505927)

[5.1 JAVA: 19](#_Toc450505928)

[5.2 JSP: 19](#_Toc450505929)

[5.3 MySQL: 19](#_Toc450505930)

1. **INTRODUCTION**

Business Modeling describes how the company operates its business and break down its daily task to small component for easy understanding with the help of Use Cases, domain model and other glossaries.

Here we are creating the business model for “The Simulation of Online Purchasing of a product”. The online shop has a simple objective i.e. to sell the product to its customer. Along with its objectives, it provide various services like user registration and login, listing inventory, placement and delivery of order and processing payment and receipts.

The business model consists two major actors. One is Server and another is customer.

The server can perform the following actions on the model:

1. User Registration and Login

The server can handle user registration and validate user login information.

1. Manage Inventory

The server can add, edit and delete its inventory and manage all listings.

1. View Orders

It can view the orders place by the customer and manage accordingly.

1. Manage Delivery

It handles all customer orders and ensure delivery of the product.

1. Receive Payment

It can receive payment from customer and validate the payment and process receipt.

The customer can perform the following actions on the model:

1. Registration and Login

The customer can register themselves in the system and later login with their credentials.

1. Browse Inventory

They can view the inventory listing and select between different options.

1. Place Orders

They can select product and place an order with necessary information.

1. Make Payment

They can make payment for the product they have selected for checkout.

1. Order Receipt

They can order receipt of the payment they have made.

1. **PROJECT OVERVIEW**

The online shopping-cart application is a web-based system. It can be accessed using Internet Explorer 8.0 and above, Mozilla Firefox 2.0, and Google Chrome.

The two interface types found in the online shopping-cart application are as follows:

1. **User Interface**: Users are able to view the home page of the shopping-cart application, browse the different categories, browse and add any number of items from any categories in the shopping cart, look for information about each product, delete the items in the shopping cart, save the cart for later viewing, check out or continue shopping after adding the item to the cart, and check out the items by completing the required information in the order form.

2. **Admin Interface**: The administrator is able to view the users’ information that was entered during checkout in the database, can update the information, price, shipping costs of the items, add or remove items from the main display.

**Hardware Interface**

The online shopping-cart application shall provide minimum hardware requirements. The following hardware configurations are required for a PC using the online shopping-cart application:

* Pentium processor
* 32 MB of free hard-drive space
* 128 MB of RAM

**Software Interface**

This section lists the requirements that are needed to run the system efficiently. The operating system needed for the system to run effectively, the interface to run the application, the driver for running Java web applications, the integrated development environment to develop the application, and the third-party tool used for editing purposes are as follows:

1. Operating System: Windows (Vista/7/8/10) or MAC OS X

2. Web Brower: Internet Explorer (8.0 and above), Mozilla Firefox (3.0 and above), or Google Chrome

3. Drivers: Java Runtime Environment

4. Integrated Development Environment: Eclipse Mars or Apache Tomcat

5. Third-Party Tool: Microsoft Word

**Product Function**

The online shopping-cart application would have the following basic functions:

1. Display all the categories available for shopping on the system’s main page.

2. Display all the items linked to each category listed on the main page.

3. Allow the administrator to add new items to the existing list of available items.

4. Allow users/administrator to remove items.

5. Allow the administrator to modify the price of each item.

6. Allow the administrator to update the description about each item.

7. Allow the administrator to view and edit information about each user that checkouts the items from the system.

**User Characteristics**

The users of the online shopping-cart application, based on their roles, are customers (users) and the administrator (owner). These users are identified based on their experience and technical expertise.

**1. Admin**: The administrator is the owner of this online shopping-cart application. One must have a basic understanding of computers and the internet as well as prior knowledge for operating the eclipse and Java programming languages. The administrator is responsible for maintaining all the training documents required for the system. The administrator can perform the following functions:

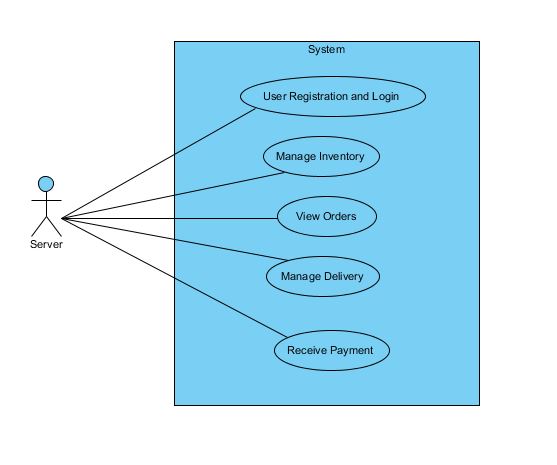
* Assign or change the price of the items, update the items in the list, and delete the items.
* Assign sales tax for different states at the time of checkout.
* View the history of the customers who purchased the items.

**2. Users**: The users of this online shopping-cart application are all customers who would shop to test the application. These users are anyone with shopping experience and the know-how to browse through a shopping-cart application. They must have basic understandings about computers and the internet. The users should be able to perform the following functions using this system:

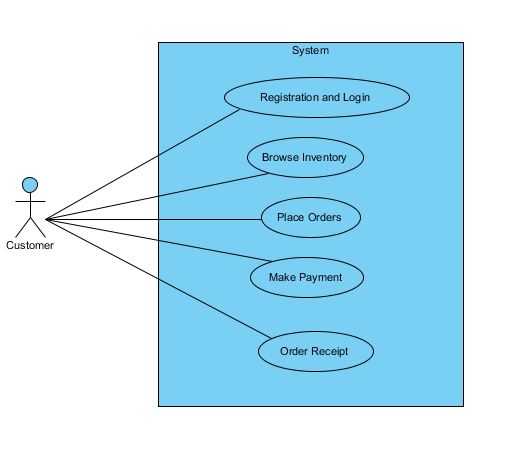
* View, browse, and select a category on the home page.
* View, add, and update items in the cart.
* Delete items from the cart.
* Check out the items from the application or continue shopping.
* Sign-on/login using a username and password.
* Place the order by completing the order form.

## 2.1 USE CASE DIAGRAM

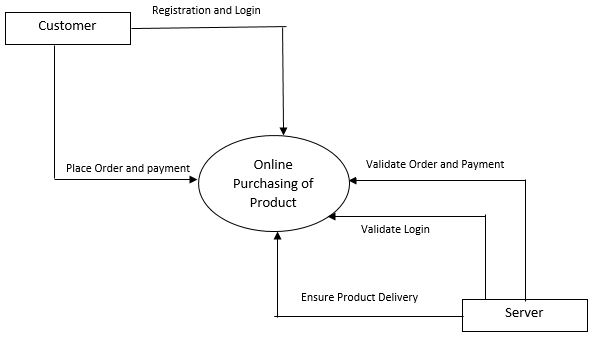
### 2.1.1 USE CASE DIAGRAM FOR SERVER:



### 2.1.2 USE CASE DIAGRAM FOR CUSTOMER:

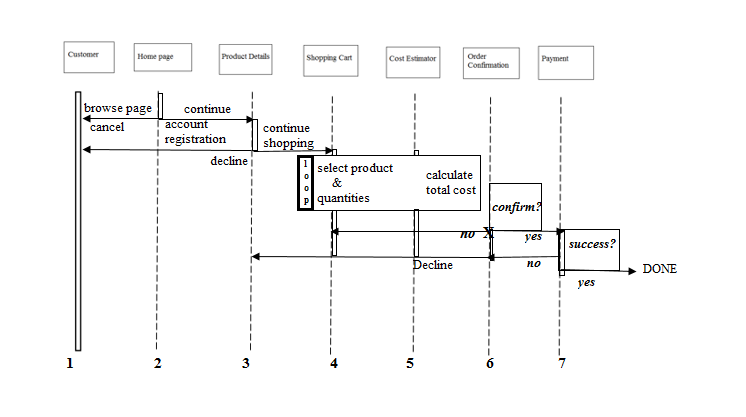


## 2.2 DOMAIN MODEL:

.

In the above domain model, Customer adopts the pet, returns the pet or provide information about lost or found pet. Similarly, the Human Society System keeps record of all the pets and provides information about disaster preparedness related to pet, animal population control and legislative activities related to pet to the customers.

## 2.3 SEQUENCE DIAGRAMS:

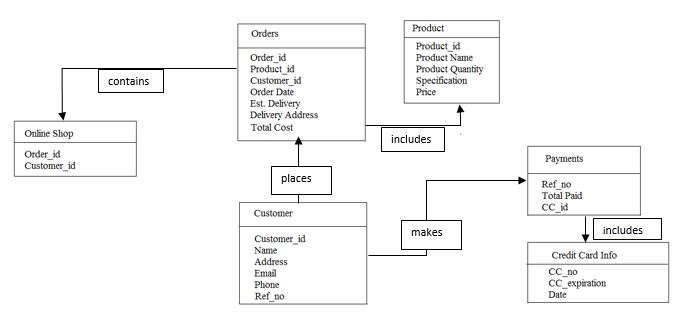


The order of sequence that take place in the system are listed below:

1. Browse Products
2. Customer landed on the product’s home page, created account etc.
3. Navigate to the product’s gallery
4. Select products and their quantities
5. Calculate the total costs instantly based on items on cart
6. Dialog box showing if customer wishes to check out
7. Make payment
8. Order Placement completed

## 2.4 CLASS DIAGRAM:

Class Diagram depicts the relationship among various entities of the system. Such diagram is used to obtain a conceptual model of the entities used in the application. This diagram gives the complete details of the entities and their tasks.

****

1. **IMPLEMENTATION**

**State chart diagram:**

State chart diagrams represent the behavior of entities capable of dynamic behavior by specifying its response to the receipt of event instances. State diagrams require that the system described is composed of a finite number of states. Typically, used for describing the behavior of classes, but statechart may also describe the behavior of other model entities such as use-cases, actors, subsystems, operations, or methods. The abstract behavior is analyzed and represented in series of events that could occur in one or more possible states. Each diagram usually represents objects of a single class and track the different states of its objects through the system.

Every company having customers maintains customer accounts and supports a complete life cycle of the account from its creation until it is closed. There are differences in what the stages (states) are in the account's life cycle, and what are conditions or events causing account to change its state.

Here we provide an example of user account life cycle in the context of online shopping, shown as UML protocol state machine diagram. For the user account to be created, it has to meet some initial requirements.

For example, user id (used as a login name) must be unique, at least for the existing accounts. After account was created, it might need to be verified. Verification depends on the company and could include e-mail, phone, and/or address verification. If account was not verified during some predefined period of time, that account could be moved to the suspended accounts.

For example, website intrusion detection system locks user account for predefined period of time, if there were several unsuccessful login attempts using incorrect account password. After account lock times out, account is activated back automatically. Some user accounts could be inactive for a long period of time. Company policy or business rules could require moving such dormant for a year or two accounts to the suspended state.

Design Pattern:

According to our domain model we got two different objects that has different states. User State & Order Status. These different objects have different states and events to triggers. So in this project we will use state design pattern. The state design pattern is one of the popular behavioral design patterns. This pattern is used when an Object changes its behavior based on its internal states.

Following are the three design patterns that can be used for implementation:

1. The Facade pattern: The facade pattern is a software engineering design pattern commonly used with Object-oriented programming. The name is by analogy to an architectural facade.

A facade is an object that provides a simplified interface to a larger body of code, such as a class library. A facade can:

1. make a software library easier to use, understand and test, since the facade has convenient methods for common tasks;
2. make code that uses the library more readable, for the same reason;
3. reduce dependencies of outside code on the inner workings of a library, since most code
4. uses the facade, thus allowing more flexibility in developing the system;

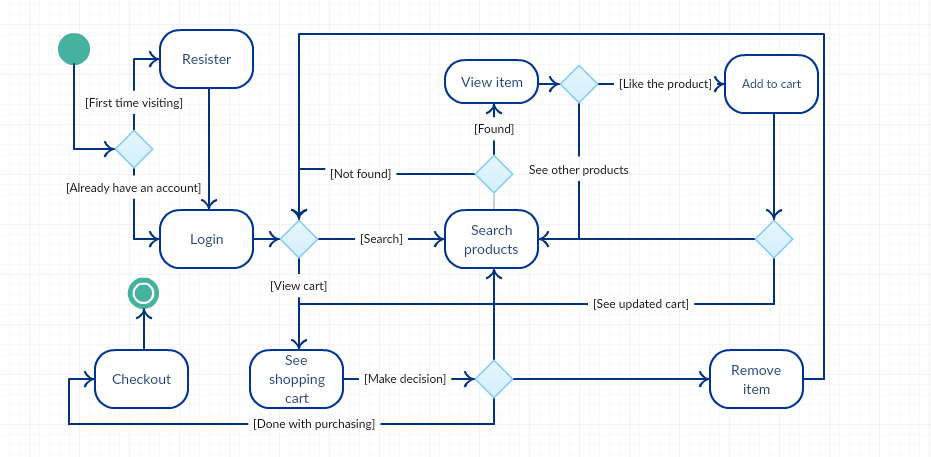
The Facade design pattern is often used when a system is very complex or difficult to understand because the system has a large number of interdependent classes or its source code is unavailable. This pattern hides the complexities of the larger system and provides a simpler interface to the client. It typically involves a single wrapper class which contains a set of members required by client. These members access the system on behalf of the facade client and hide the implementation details.

1. Visitor Pattern: Visitor pattern is one of the behavioral design pattern. Visitor pattern is used when we have to perform an operation on a group of similar kind of Objects. With the help of visitor pattern, we can move the operational logic from the objects to another class.

For example, think of a Shopping cart where we can add different type of items (Elements), when we click on checkout button, it calculates the total amount to be paid. Now we can have the calculation logic in item classes or we can move out this logic to another class using visitor pattern. Let’s implement this in our example of visitor pattern.

* The benefit of this pattern is that if the logic of operation changes, then we need to make change only in the visitor implementation rather than doing it in all the item classes.
* Another benefit is that adding a new item to the system is easy, it will require change only in visitor interface and implementation and existing item classes will not be affected.

1. The State pattern: The state pattern, which closely resembles Strategy Pattern, is a behavioral software design pattern, also known as the objects for states pattern. This pattern is used in computer programming to encapsulate varying behavior for the same routine based on an object's state object. This can be a cleaner way for an object to change its behavior at runtime without resorting to large monolithic conditional statements.



1. **TESTING MODELS**

Testing is the process of evaluating a system or its component(s) with the intent to find that whether it satisfies the specified requirements or not. This activity results in the actual, expected and difference between their results.

### 4.1 Testing Types

* + 1. **Manual Testing:** The tester takes over the role of an end user and test the Software to identify any un-expected behavior or bug.
    2. **Automation Testing**: The tester writes scripts and uses another software to test the software. It is also used to test the application from load, performance and stress point of view.

### 4.2 Testing Methods

* + 1. **Black Box Testing:** The technique of testing without having any knowledge of the interior workings of the application is Black Box testing. The tester is oblivious to the system architecture and does not have access to the source code.
    2. **White Box Testing:** White box testing is the detailed investigation of internal logic and structure of the code. The tester needs to have a look inside the source code and find out which unit/chunk of the code is behaving inappropriately.
    3. **Grey Box Testing:** Grey Box testing is a technique to test the application with limited knowledge of the internal workings of an application.

1. **LANGUAGE AND TECHNOLOGIES USED FOR FRONT END AND BACK END**

### 5.1 JAVA:

Java is a programming language offering many features that make it attractive for mathematical illustration. The rest thing to know is that Java is an object-oriented language. This provides the programmer with a natural way to divide programs into smaller, reusable pieces, called objects, thus making it easier to manage, debug and expand the code.

### 5.2 JSP:

Java Server Pages (JSP) is a technology that helps [software developers](http://en.wikipedia.org/wiki/Software_developer) create [dynamically generated web pages](http://en.wikipedia.org/wiki/Dynamic_web_page) based on [HTML](http://en.wikipedia.org/wiki/HTML), [XML](http://en.wikipedia.org/wiki/XML), or other document types. JSP is similar to [PHP](http://en.wikipedia.org/wiki/PHP), but it uses the [Java programming language](http://en.wikipedia.org/wiki/Java_(programming_language)).

### 5.3 MySQL:

The **MySQL** client program to create and use a simple database **MySQL** (sometimes referred to as the “terminal monitor” or just “monitor”) is an interactive program that enables you to connect to a MySQL server, run queries, and view the results

**Future Scope:**

We have a default admin account. If someone wants to access admin account then he must know username and password. There is no way that we can create an account for admin.