**BuyFy**



E-Commerce Project

SHOPPING WEBSITE

A small initiative by the scholars to develop a Web-Application

Contents

1. Introduction 3
2. Purpose 3
3. Scope 3
4. Definition 3
5. Overview 3
6. Overall Description 4
7. Functionalities 4
8. Non-functional requirements 5
9. Architecture Design 6
10. Interface Requirements 7
11. Functional Requirements 7
12. ER-Diagram 8
13. Testing 9

1. Introduction

1.1 Purpose: This document is meant to delineate the features of OSS, so as to serve as a guide to the developers on one hand and a software validation document for the prospective client on the other. The Online Shopping System (OSS) for items shop web application is intended to provide complete solutions for vendors as well as customers through a single get way using the internet. It will enable customer to browse through the shop and purchase them online without having to visit the shop physically. The administration module will enable a system administrator to approve and reject requests for new shops and maintain various lists of shop category.

1.2 Scope: This system allows the customer’s to maintain their cart for add or remove the product over the internet.

1.3 Definitions:

OSS- Online shopping System

SRS- Software Requirement Specification

GUI- Graphical User Interface

Stakeholder- The person who will participate in this system.

Ex. Customer, Administrator, Visitor etc.

1.4 Overview: This system provides an easy solution for customers to buy the product without going to the shop and also to shop owner to sale the product. This proposed system can be used by any naïve users and it does not require any educational level, experience or technical expertise in computer field but it will be of good use if user has the good knowledge of how to operate a computer.

2. Overall Description: The BuyFy application enable customers to browse through the shops, and a system administrator to approve and reject requests for new shops and maintain lists of shop categories. Also the developer is designing an online shopping site to manage the items in the shop and also help customers to purchase them online without visiting the shop physically. The online shopping system will use the internet as the sole method for selling goods to its consumers.

2.1 Functionalities:

This is a web application, with following functional requirements.

* All the users will be able to login through same login page.
* Customers can explore a variety of products.
* Customers can easily walkthrough the application provided by different tabs and navigation bars.
* Lavishing and user-friendly UI for the customers.
* Smooth access throughout the application.
* Customers can create accounts for exclusive offers.
* Logged in customers can add products to the cart for later.
* Safe and quick access to the payment page.
* Customers can delete their accounts.

2.2 Non-Functional Requirements: Following Non-Functional Requirements will be there in the insurance to the internet:

(i) Secure access to consumer’s confidential data.

(ii) 24X7 availability.

(iii) Better component design to get better performance at peak time.

(iv) Flexible service based architecture will be highly desirable for future extension.

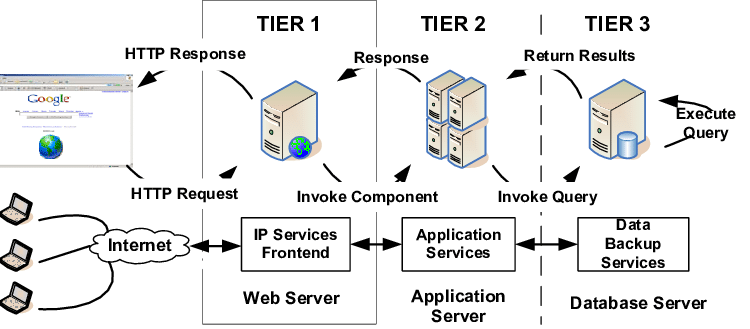
Non-Functional Requirements define system properties and constraints.

Various other Non-Functional Requirements are:

* Security
* Reliability
* Maintainability
* Portability
* Extensibility
* Reusability
* Compatibility
* Resource Utilization

|  |  |
| --- | --- |
| **Platform** | **Java/Bootstrap/Ajax/jQuery/JavaScript/**  **CSS/Tailwind CSS** |
| **Database** | **Oracle** |
| **Server** | **Apache Tomcat** |

3. Architecture Diagram: The project is developed using 3-Tier Architecture. If we consider a 3-Tier Architectural WEB application, the browser becomes the client side application. The user communicates with the WEB/Application server via the browser. In the context of this project, the web pages represent the presentation layer. Here we can write code to design frontend and can perform validations. The data may be passed from this layer to business tier which has classes, methods and database connections. Now the queries are sent to Data Tier which gets data from Oracle server and returns it to Business Tier.

The user should be presented with the home page (when unauthenticated).

4. Interface Requirements: Various interfaces for the product could be

1). Login Page

2). Registration Form

3). There will be a screen displaying information about product that the shop have.

4). If the customers select the buy button then another screen of shopping cart will be opened for checkout.

5). After ordering for the product, the system will sent one copy of the bill to the customer’s Email address.

**5. Functional Requirements:** This section provides requirement overview of the system. Various functional modules that can be implemented by the system will be

3.1 Description:

3.1.1 Registration : If customer wants to buy the product then he/she must be registered, unregistered user can’t go to the shopping cart.

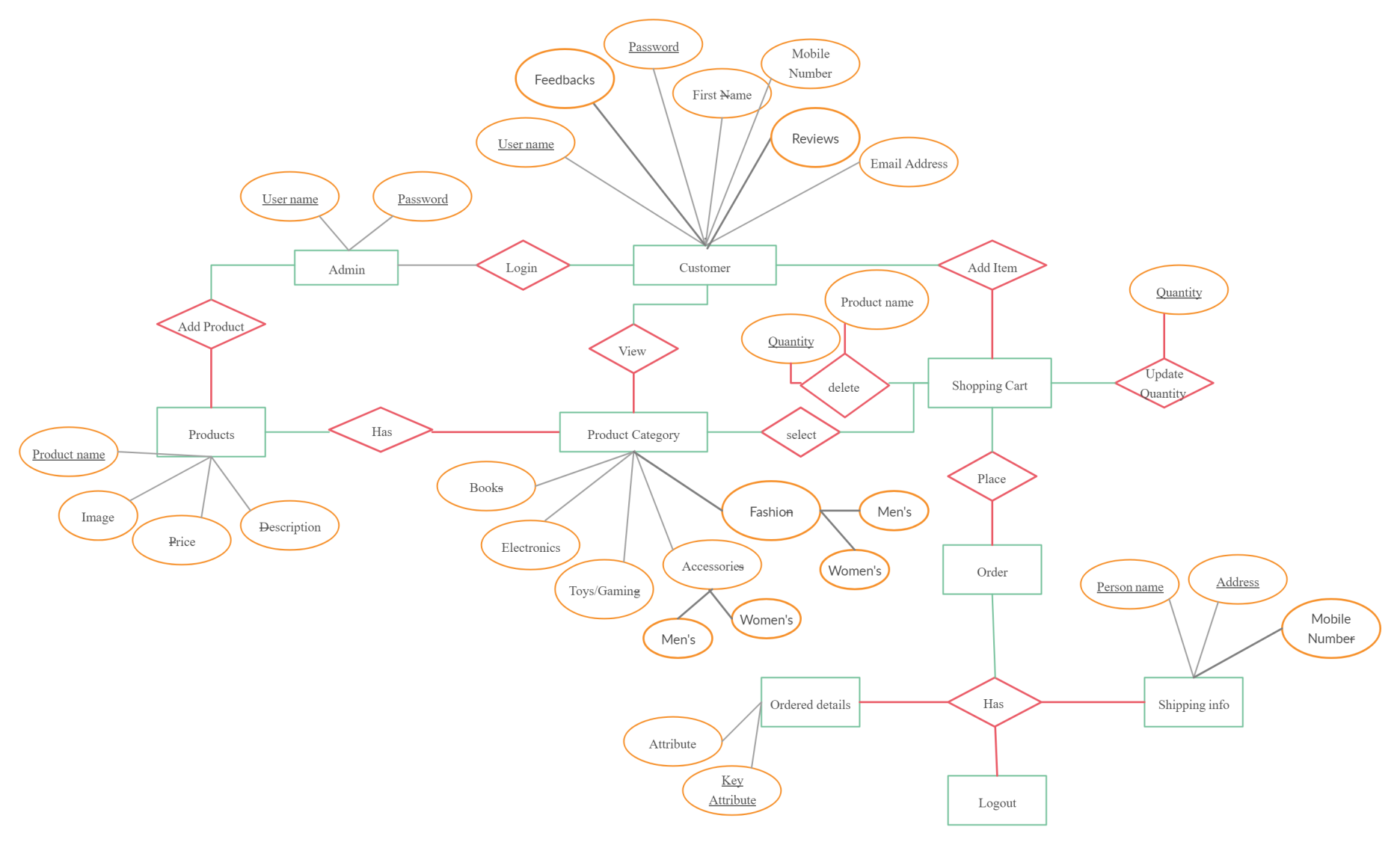
3.1.2 Login Customer logins to the system by entering valid user id and password for the shopping.

3.1.3 Changes to Cart means the customer after login or registration can make order or cancel order of the product from the shopping cart.

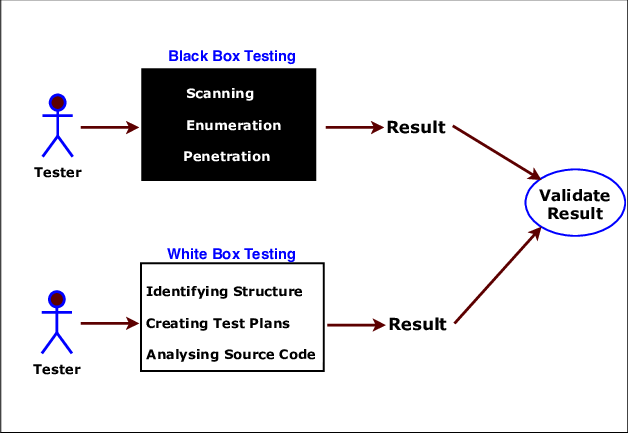
3.1.4 Payment: In this system we are dealing the mode of payment by paytm.

3.1.5 Logout: After ordering or surfing for the product customer has to logout.

3.1.6 Report Generation: After ordering for the product, the system will sent one copy of the bill to the customer’s Email-address and another one for the system data base.

 **ER Diagram**

1. Testing



7.1 White-box testing: White-box testing (also known as clear box testing, glass box testing and structural testing, by seeing the source code) tests internal structures or workings of a program, as opposed to the functionality exposed to the end-user.

This is done by the developer itself in parallel to developing of above project. This involves appropriate structuring of query and logic together.

7.2 Black-box testing: Black-box testing treats the software as a “Black Box”, examining functionality without any knowledge of internal implementation, without seeing the source code.

Some examples are:

* + Login validations
  + It should take correct Data Input Types
  + It should redirect to correct pages
  + No Data loss
  + Proper alignment of web pages

Once the application is built the black box testing is done by creating some users and test with proper test case scenarios.