**Post Office Management System**

21-12-2017

**Software Requirement Specification, Design & Testing**

**SE-801**



**Course Name:** Software Project Lab-3

**Course Code:** SE-801

**Submitted by**

Tanvir Ahmed Siddiki

BSSE 0639

**Supervisor**

Emon Kumar Dey

Assistant Professor

**Submitted to**

BSSE 4th Year Program Committee

IIT, University of Dhaka

**Submission Date**

21-12-2017

**Institute of Information Technology**

**University of Dhaka**

# **Letter of Transmittal**

21th December, 2017,

BSSE 4th Year Program Committee,

Institute of Information Technology,

University of Dhaka.

Sir,

I have prepared the report on Software Requirements Specification, Software Design and Testing of **‘Post Office Management System’** for your approval. This report details the requirements that I have gathered for the project.

The primary purpose of this report is to summarize my findings from the work that I have completed as my Software Requirements Specification and Analysis, Design and Testing course project. This report includes the details of each step we followed to collect the requirements.

Sincerely Yours,

Tanvir Ahmed Siddki (BSSE 0639)

Institute of Information Technology

University of Dhaka

Session: 2011-12

# **Executive Summary**

A Post Office Management System is proposed where all the basic information’s of parcels and moneys orders of post office and as well as the current location of these products will provide according to their respective branches. This is totally a web based system which can be used by any branch of post office in Bangladesh. This system will reduce a lot of time and human effort to find out the postal orders.

# 

# **Acknowledgement**

By the grace of Almighty Allah, I have completed my report on Software Requirements of Post Office Management System.

I am very much grateful to my honorable supervisor sir Emon Kumar Dey for his supervision throughout the working time. He helped me a lot by sharing his invaluable knowledge with me. I would also like to thank course instructors, Amit Seal Ami and Nadia Nahar for their valuable supervision, advice, instruction and time throughout the project.

Everyone was very much helpful. I just cannot thank them enough.

**Table of Contents**

[**Letter of Transmittal** i](#_Toc501478878)

[**Executive Summary** ii](#_Toc501478879)

[**Acknowledgement** iii](#_Toc501478880)

[**Chapter 1** 1](#_Toc501478881)

[**Introduction** 1](#_Toc501478882)

[1.1 Purpose 1](#_Toc501478883)

[1.2 Intended Audience 1](#_Toc501478884)

[**Chapter 2** 2](#_Toc501478885)

[**Inception** 2](#_Toc501478886)

[2.1 Introduction 2](#_Toc501478887)

[2.2 Identifying Stakeholders 2](#_Toc501478888)

[2.3 Recognizing Multiple Viewpoints 3](#_Toc501478889)

[2.4 Working towards Collaboration 3](#_Toc501478890)

[2.5 Asking the First Question 4](#_Toc501478891)

[2.6 Conclusion 4](#_Toc501478892)

[**Chapter 3** 5](#_Toc501478893)

[**Elicitation** 5](#_Toc501478894)

[3.1 Introduction 5](#_Toc501478895)

[3.2 Eliciting Requirements 5](#_Toc501478896)

[3.2.1 Collaborative Requirement Gathering 5](#_Toc501478897)

[3.2.2 Quality Function Deployment 5](#_Toc501478898)

[3.2.2.1 Normal Requirements 6](#_Toc501478899)

[3.2.2.2 Expected Requirements 6](#_Toc501478900)

[3.2.2.3 Exciting Requirements 6](#_Toc501478901)

[3.3 Usage Scenario 7](#_Toc501478902)

[**Chapter 4** 8](#_Toc501478903)

[**Scenario Based Modeling** 8](#_Toc501478904)

[4.1 Definition of Use case 8](#_Toc501478905)

[4.2 Use Case Diagrams 8](#_Toc501478906)

[4.2.1 Level-0 of Use Case: 8](#_Toc501478907)

[4.2.2 Level-1 of Use Case: 10](#_Toc501478909)

[4.2.2.1 Level-1.1 of Use Case: 11](#_Toc501478911)

[4.2.2.2 Level-1.2 of Use Case: 12](#_Toc501478913)

[4.2.2.3 Level-1.3 of Use Case: 13](#_Toc501478915)

[4.2.2.3.1 Level-1.3 of Use Case: 14](#_Toc501478917)

[4.3 Activity & Swim Lane Diagram 15](#_Toc501478919)

[**Chapter 5** 36](#_Toc501478941)

[**Data Model** 36](#_Toc501478942)

[5.1 Data Modeling Concept 36](#_Toc501478943)

[5.2 Data Objects 36](#_Toc501478944)

[5.2.1 Grammatical Parsing (Noun Identity) 36](#_Toc501478945)

[5.2.2 Identify Data Objects 38](#_Toc501478946)

[5.2.3 Relationship Between Data Objects 40](#_Toc501478947)

[5.2.4 Entity Relationship Diagram 41](#_Toc501478949)

[5.2.5 Database Table 42](#_Toc501478951)

[**Chapter 6** 45](#_Toc501478952)

[**Class Based Model** 45](#_Toc501478953)

[6.1 Introduction 45](#_Toc501478954)

[6.2 General Classification 45](#_Toc501478955)

[6.3 Selection Criteria 45](#_Toc501478956)

[6.4 Class Identification 46](#_Toc501478957)

[6.5 Class Card: 48](#_Toc501478958)

[6.6 Class Diagram 52](#_Toc501478959)

[**Chapter 7** 53](#_Toc501478961)

[**Flow-Oriented Model** 53](#_Toc501478962)

[7.1 Introduction 53](#_Toc501478963)

[7.2 Data flow diagram 53](#_Toc501478964)

[**Chapter 8** 57](#_Toc501478971)

[**Behavioral Model** 57](#_Toc501478972)

[8.1 Introduction 57](#_Toc501478973)

[8.2 Identifying Events 57](#_Toc501478974)

[8.3 State Transition Diagram 57](#_Toc501478975)

[8.4 Sequence Diagram 60](#_Toc501478979)

[**Chapter 9** 61](#_Toc501478981)

[**Software Design** 61](#_Toc501478982)

[9.1 Architectural Design 61](#_Toc501478983)

[9.1.1 Representing the system in context 61](#_Toc501478984)

[9.1.2 Define Archetypes 62](#_Toc501478986)

[9.1.3 Refining Archetypes into Components 62](#_Toc501478987)

[9.1.4 Describing Instantiation of the System 63](#_Toc501478988)

[9.2 Component Level Design 65](#_Toc501478993)

[9.2.1 Identifying all design classes that correspond to the problem domain 65](#_Toc501478994)

[9.2.2 Identify all design classes that correspond to the infrastructure domain 65](#_Toc501478995)

[9.2.3 Elaborate all design classes that are not acquired as reusable components 66](#_Toc501478996)

[9.2.3.1 Collaboration Details 71](#_Toc501478997)

[9.2.3.2 Appropriate Interfaces 72](#_Toc501478998)

[9.2.3.3 Elaborate Attributes 72](#_Toc501478999)

[9.2.3.4 Describe Processing Flow 74](#_Toc501479000)

[9.2.4 Persistent Data 76](#_Toc501479003)

[9.2.5 Develop and elaborate behavioral representations for a class or component 77](#_Toc501479004)

[9.2.6 Elaborating deployment diagrams 79](#_Toc501479007)

[9.3 User Interface Design 80](#_Toc501479009)

[9.3.1 User Analysis: 80](#_Toc501479010)

[9.3.2 User Interface (MOC): 80](#_Toc501479011)

[**Chapter 10** 88](#_Toc501479025)

[**Testing** 88](#_Toc501479026)

[10.1 Introduction 89](#_Toc501479027)

[10.2 Test plan identifier 89](#_Toc501479028)

[10.2.1 Summary of items and features to be tested 89](#_Toc501479029)

[10.2.2 Requirement and history of items 89](#_Toc501479030)

[10.2.3 High-level description of testing goals 89](#_Toc501479031)

[10.2.4 Reference Document 89](#_Toc501479032)

[10.3 Scope of Testing 89](#_Toc501479033)

[10.4 Test items 90](#_Toc501479034)

[10.5 Features to be tested 90](#_Toc501479035)

[10.6 Features not to be tested 90](#_Toc501479036)

[10.7 Approach 90](#_Toc501479037)

[10.8 Item pass/fail criteria 90](#_Toc501479038)

[10.9 Suspension criteria and Resumption requirements 91](#_Toc501479039)

[10.10 Test Deliverables 91](#_Toc501479040)

[10.11 Environmental needs 91](#_Toc501479041)

[10.12 Staffing and training needs 91](#_Toc501479042)

[10.13 Responsibilities 92](#_Toc501479043)

[10.14 Risks and contingencies 92](#_Toc501479044)

[10.15 Testing cost: 92](#_Toc501479045)

[10.16 Approval 92](#_Toc501479046)

[10.17 Test Cases 93](#_Toc501479047)

[**Chapter 11** 110](#_Toc501479048)

[**User Manual** 110](#_Toc501479049)

[12.1 Introduction 110](#_Toc501479050)

[12.2 User’s Guide 110](#_Toc501479051)

[**Chapter 12** 121](#_Toc501479052)

[**Conclusion** 121](#_Toc501479053)

[**Chapter 13** 122](#_Toc501479054)

[**Reference** 122](#_Toc501479055)

List of Figures

[Figure 4.1: Level-0 Use Case Diagram 9](#_Toc501304727)

[Figure 4.2: Level-1 Use Case Diagram 10](#_Toc501304729)

Figure 4.3: Level-1.1 Use Case Diagram 11

[Figure 4.4: Level-1.2 Use Case Diagram 12](#_Toc501304733)

[Figure 4.5: Level-1.3 Use Case Diagram 13](#_Toc501304735)

Figure 4.6: Level-1.3.1 Use Case Diagram 14

[Figure 4.7: Activity Diagram for Sign In 15](#_Toc501304739)

[Figure 4.8: Swim Lane Diagram for Sign In 16](#_Toc501304740)

[Figure 4.9: Activity Diagram for Sign Out 17](#_Toc501304741)

[Figure 4.10: Swim Lane Diagram for Sign Out 18](#_Toc501304742)

[Figure 4.11: Activity Diagram for Password Change 19](#_Toc501304743)

[Figure 4.12: Swim Lane Diagram for Password Change 20](#_Toc501304744)

[Figure 4.13: Activity Diagram for Add Branch 21](#_Toc501304745)

[Figure 4.14: Swim Lane Diagram for Add Branch 22](#_Toc501304746)

[Figure 4.15: Activity Diagram for Assign Managers 23](#_Toc501304747)

[Figure 4.16: Swim Lane Diagram for Assign Managers 24](#_Toc501304748)

[Figure 4.17: Activity Diagram for Edit Branch 25](#_Toc501304749)

[Figure 4.18: Swim Lane Diagram for Edit Branch 26](#_Toc501304750)

[Figure 4.19: Activity Diagram for Edit Branch Managers 27](#_Toc501304751)

[Figure 4.20: Swim Lane Diagram for Edit Branch Managers 28](#_Toc501304752)

[Figure 4.21: Activity Diagram for Add Product 29](#_Toc501304753)

[Figure 4.22: Swim Lane Diagram for Add Product 30](#_Toc501304754)

[Figure 4.23: Activity Diagram for Product Management 31](#_Toc501304755)

[Figure 4.24: Swim Lane Diagram for Update Product 32](#_Toc501304756)

[Figure 4.25: Swim Lane Diagram for Approve Request 33](#_Toc501304757)

[Figure 4.26: Activity Diagram for View Current Location 34](#_Toc501304758)

[Figure 4.27: Swim Lane Diagram for View Current Location 35](#_Toc501304759)

[Figure 5.1: Relationship between data objects 40](#_Toc501304767)

Figure 5.2: ER Diagram of Post Office Management System 41

[Figure 6.1: Class Diagram 52](#_Toc501304779)

[Figure 7.1: Level-0 Data Flow Diagram 53](#_Toc501304784)

[Figure 7.2: Level-1 Data Flow Diagram 54](#_Toc501304785)

[Figure 7.3: Level-1.1 Data Flow Diagram 55](#_Toc501304786)

[Figure 7.4: Level-1.2 Data Flow Diagram 55](#_Toc501304787)

[Figure 7.5: Level-1.3 Data Flow Diagram 56](#_Toc501304788)

[Figure 7.6: Level-1.4 Data Flow Diagram 56](#_Toc501304789)

[Figure 8.1: State Transition Diagram for System Admin 58](#_Toc501304795)

[Figure 8.2: State Transition Diagram for Branch Manager 58](#_Toc501304796)

[Figure 8.3: State Transition Diagram for Customer 59](#_Toc501304797)

Figure 8.4: Sequence Diagram 60

Figure 9.1: Architectural Context Diagram 61

[Figure 9.2: Refining ‘Authentication’ archetype into components and classes 63](#_Toc501304808)

[Figure 9.3: Refining ‘Branch Management’ archetype into components and classes 63](#_Toc501304809)

[Figure 9.4: Refining ‘Product Management’ archetype into components and classes 64](#_Toc501304810)

[Figure 9.5: Refining ‘Tracking System’ archetype into components and classes 64](#_Toc501304811)

[Figure 9.6: Processing flow for “Add Product” method 74](#_Toc501304820)

[Figure 9.7: Processing flow for “Product Management” method 75](#_Toc501304821)

[Figure 9.8: State Chart fragment for System Admin class 77](#_Toc501304824)

[Figure 9.9: State Chart fragment for Branch manager class 78](#_Toc501304825)

Figure 9.10: Elaborative deployment diagram of POMS 79

[Figure 9.11: User interface for “Start” page 80](#_Toc501304831)

[Figure 9.13: User interface for “View branch” page 81](#_Toc501304832)

[Figure 9.14: User interface for “Add branch” page 82](#_Toc501304833)

[Figure 9.15: User interface for “Branch manager” page 82](#_Toc501304834)

[Figure 9.16: User interface for “Add branch manager” page 83](#_Toc501304835)

[Figure 9.17: User interface for “Product (parcel/money order)” page 83](#_Toc501304836)

[Figure 9.18: User interface for “Number of product” window 84](#_Toc501304837)

[Figure 9.19: User interface for “Add Product” window 84](#_Toc501304838)

[Figure 9.20: User interface for “Track of product” window 85](#_Toc501304839)

[Figure 9.21: User interface for “Google Map” window 85](#_Toc501304840)

[Figure 9.22: User interface for “Product” page 86](#_Toc501304841)

[Figure 9.23: User interface for “Add track” window 86](#_Toc501304842)

[Figure 9.24: User interface for “Track product” page 87](#_Toc501304843)

# **Chapter 1**

# **Introduction**

## 1.1 Purpose

This document is the Software Requirement Specification (SRS) for Post Office Management. It contains functional, non-functional and support requirements and establishes a requirements baseline for the development of the system. The requirements contained in the SRS are independent, uniquely identified, numbered, and organized by topic. The SRS serves as official means of communicating user requirements to the developer and provides a common reference point for both the developer team and stakeholder community. The SRS will evolve over time as users and developers work together to validate, clarify and expand its contents.

## 1.2 Intended Audience

This SRS is intended for several audiences including the customers as well as the project managers, designers, developers, and testers.

1. The customer will use this SRS to verify that the developer team has created a product that is acceptable to the customer.
2. The project managers of the developer team will use this SRS to plan milestones and a delivery date, and ensure that the developing teams on track during development of the system.
3. The designers will use this SRS as a basis for creating the system’s design. The designers will continually refer back to this SRS to ensure that the system they are designing will fulfill the customer’s needs.
4. The developers will use this SRS as a basis for developing the system’s functionality. The developers will link the requirements defined in this SRS to the software they create to ensure that they have created software that will fulfill all of the customer’s documented requirements.
5. The testers will use this SRS to derive test plans and test cases for each documented requirement. When portions of the software are complete, the testers will run their tests on that software to ensure that the software fulfills the requirements documented in this SRS. The testers will again run their tests on the entire system when it is complete and ensure that all requirements documented in this SRS have been fulfilled.

# **Chapter 2**

# **Inception**

## 2.1 Introduction

Inception is the beginning phase of requirements engineering. It defines how does a software project get started and what is the scope and nature of the problem to be solved. The goal of the inception phase is to identify concurrence needs and conflict requirements among the stakeholders of a software project. To establish the ground work, I have worked with the following factors related to the inception phases:

1. Identifying Stakeholders
2. Recognizing multiple viewpoints
3. Recognizing multiple viewpoints
4. Asking the First Questions

## 2.2 Identifying Stakeholders

Stakeholder refers to any person or group who can influence or can be influenced by the system directly or indirectly. Stakeholders include end-users who interact with the system and everyone else in an organization that may be influenced by its installation**.** Stakeholder identification is the process used to identify all stakeholders for a project. It is important to understand that not all stakeholders have the same influence or effect on a project, nor will they be influenced in the same manner. It should be done in a methodical and logical way to ensure that stakeholders are not easily omitted. The following questions help us to identify stakeholders:

1. Who uses the system?
2. Who is affected by the outputs of the project?
3. Who evaluates/approves system?
4. Who maintains the system
5. Who has knowledge (specialist) about the system?
6. Whose work will influence my project?

I have gathered information about the Post Office and identified the following stakeholders for my Post Office Management System.

* **System Admin:** System admin can add the branches of post offices and also can assign branch managers to respective branches. System admin has also the authorization of parcel, money order tracking and all the administrative privileges.
* **Branch Managers:** Branch managers can add any new parcel or money orders, edit the order information’s and update. He/she get the notification of pending requests in his/her branch and also can track the current location of the parcel, assign the parcel to the next branch and deliver to the receiver’s.
* **Customers:** In the system customers are the senders or receivers of the parcel or money order. He/she got the track id of any issued product through his/her email and with this he/she can track that currently where the product is. In terms of money order customer get the pin to withdraw the money.

## 2.3 Recognizing Multiple Viewpoints

After discussing with the post office branch officers and some customers I have collected these viewpoints.

**System Administrator viewpoints:**

1. Want to control the whole system
2. Maintain a database that consists all information
3. Data recovery system in case of data lose
4. Get notification in case of any changes or updates
5. Less costly

**Branch Managers viewpoints:**

1. Allow the system to be accessed through desktop with web based application.
2. Get notification.
3. Want to know detail about the senders and receivers
4. Want to edit information any time
5. Track the product and deliver

**Customers viewpoint:**

1. Well secure

2.Allow the system to be accessed via Desktop Computer

3.Allow to get current information of his product

## 2.4 Working towards Collaboration

I have found different requirements from the different stakeholders. So I follow some rules to identify the final requirements. Rules are:

1. Identify the common and conflicting requirements
2. Categorize the requirements
3. Take priority points from the stakeholders for conflicting requirements

4. Take final decision

**Common Requirements:**

I have found the following requirements which are required by all the stakeholders:

1. Easy to use and Well secure
2. Less time consuming
3. Web based interface
4. Can be accessible from any computer that has an active internet connection
5. Get notification for any changes and updates

**Conflicting Requirements:**

I have also found some requirements those are conflicting each other’s.

1. Easy to access and strong security
2. Want fixed data and want to change data at any time
3. Less costly but rich functionality

**Final Requirements:**

I select the final requirements for the system by categorizing and prioritizing.

The Finalizing Requirements are:

1. Easy to use and maintain
2. Allow general viewers to see only the public information of the applicants
3. Allow the admin officer to control the whole system
4. Data recovery system
5. Get notification of important updates and changes

## 2.5 Asking the First Question

I have asked three sets of question:

1. I set my first set of context-free questions focuses on the customer and other stakeholders, overall project goals and benefits. This set of context-free question is explained above. These questions helped us to identify all stakeholders, measurable benefit of the successful implementation and possible alternatives to custom software development
2. Next set of question helped us to gain a better understanding of problem and allows the customer to voice his or her perception about the solution.
3. The final set of question focused on the effectiveness of the communication activity itself

## 2.6 Conclusion

Inception phase helped me to establish basic understanding about Post Office Bangladesh to identify the people who will be benefited this system becomes automated, define the nature of the software and establish a preliminary communication with our stakeholders.

# **Chapter 3**

# **Elicitation**

## 3.1 Introduction

Elicitation is a task that helps the customer to define what is required. To complete the elicitation step we face many problems like problems of scope, problems of volatility and problems of understanding. However, this is not an easy task. To help overcome these problems, we have worked with the Eliciting requirements activity in an organized and systematic manner.

## 3.2 Eliciting Requirements

Unlike inception where Q&A (Question and Answer) approach is used, elicitation makes use of a requirements elicitation format that combines the elements of problem solving, elaboration, negotiation, and specification. It requires the cooperation of a group of end-users and developers to elicit requirements. To elicit requirements, we completed following four works.

1. Collaborating Requirements Gathering
2. Quality Function Development
3. Usage Scenarios
4. Elicitation Work Products

### 3.2.1 Collaborative Requirement Gathering

Many different approaches to collaborative requirements gathering have been proposed. Each makes use of a slightly different scenario. I have completed following steps:

1. The meeting was conducted with the manager of a post office branch and after exploring the expectation of a web based automated system there was proposed a system which is named POMS (Post Office Management System)
2. They were asked about the problems they are facing using the manual existing system.
3. At last I select my final requirements list from the meeting.

### 3.2.2 Quality Function Deployment

Quality function deployment (QFD) known as a way to represent the “voice of the customer,” is a process for capturing customer requirements and translating them into requirements that can be used by designers, producers, and suppliers. It concentrates on maximizing customer satisfaction from the Software engineering process. With respect to our project the following requirements are identified by a QFD.

#### 3.2.2.1 Normal Requirements

Normal requirements consist of objectives and goals that are stated during

the meetings with the customers. Normal requirements of my project is:

1. Allow system to check for valid users
2. Allow valid users to login and logout
3. Restrict access to functionality of the system based upon user roles
4. Efficient and user friendly system
5. Allow user to search valid products and money orders
6. Easy and understandable user interface
7. Allow user to track product with current location
8. Allow users to receive and deliver products

#### 3.2.2.2 Expected Requirements

These requirements are implicit to the system and may be so fundamental that

the customer does not explicitly state them. Their absence will be a cause of

dissatisfaction. These requirements are:

1. Maintain a database to store all the information products and users
2. The system should enable the admin and managers to change the password
3. The system should allow the user to log in based upon an assigned login id and password
4. The system should allow the users to search, update or delete
5. The system should be user friendly
6. System will auto generate the product id numbers
7. It will be open for extension and modification
8. The system should be stable and easily maintainable.

#### 3.2.2.3 Exciting Requirements

In the whole system I have tried to add some exciting requirements. They are,

1. The system will have the google map based tracking system through which users will get the exact current location of the product.
2. PIN number via mobile phone SMS is another exciting requirement of the system

## 

## 3.3 Usage Scenario

**Post Office Management System**

Post office management system is a system of Bangladesh government owned organization “Post Office” which provides postal delivery and public services manually. My proposed system is to automate the parcel delivery and money order services.

After signing in as System admin of post office management system he/she can view the options of add new branches, branch managers view and edit branches and managers. He/she can add all the Zilla and Upzilla branches of post office across the country. He/she can also search the branch and edit branch information’s. Branches are added sector wised through division to zilla and upzilla. He can also add the branch managers and assign the managers to respective branches. He/she can search the managers and edit personal information’s of the managers. System admin has also the authorization of track the parcels and can go through the whole system.

After logging in as Branch managers, he/she can see the option of adding new parcels and money orders, update information’s of the products and tracking options. At the time of adding a new parcel or money order it is specified with an auto generated id number and then branch managers upload all the information’s of the product including source, destination, sender and receiver’s information’s, status of the product. He/she can assign the product to the up next branch and track it. He/she can also see the product list in his branch and get notified about the pending request of products from another branch and can approve the request. He/she can track the product with its current location through google map. Branch manager can also deliver the money order to the receiver by matching the pin number of respective money order slip.

In this system, customers get their issued product id number through email and can track the current location of their product through the number. He/she can receive the money through pin number which send through mobile phone sms.

# **Chapter 4**

# **Scenario Based Modeling**

## 4.1 Definition of Use case

A use case captures a contract that describes the system behavior under various conditions as the system responds to a request from one of its stakeholders. In essence, a use case tells a stylized story about how an end user interacts with the system under a specific set of circumstances. A use case diagram simply describes a story using corresponding actors, who perform important role in the story and makes the story understandable for the users. The first step in writing a use case is to define that set of “actors” that will be involved in the story. Actors are the different people that use the system or product within the context of the function and behavior that is to be described. Actors represent the roles that people play as the system operators. Every user has one or more goals when using system.

**Primary Actor**: Primary actors interact directly to achieve required system function and derive the intended benefit from the system. They work directly and frequently with the software.

**Secondary Actor:** Secondary actors support the system so that primary actors can do their work. They either produce or consume information.

## 4.2 Use Case Diagrams

Use case diagrams give the non-technical view of overall system.

### 4.2.1 Level-0 of Use Case:

After analyzing the user-scenario I have found three actors who will directly use the system as a system operator. Primary actors are those who will play action and get a reply from the system whereas secondary actors only produce or consume information.

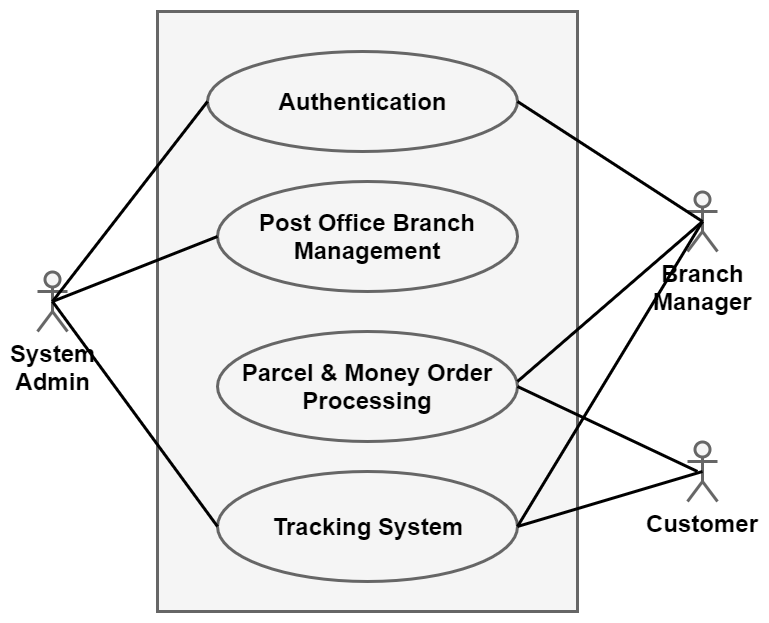
They all will be our users, who will be using the system to fulfill their procedures.



##### Figure 4.1: Level-0 Use Case Diagram

### 4.2.2 Level-1 of Use Case:

The whole system is divided into four sub-systems. They are authentication, post office branch management, parcel and money order processing, tracking system. All the actors are connected to the corresponding sub-systems.

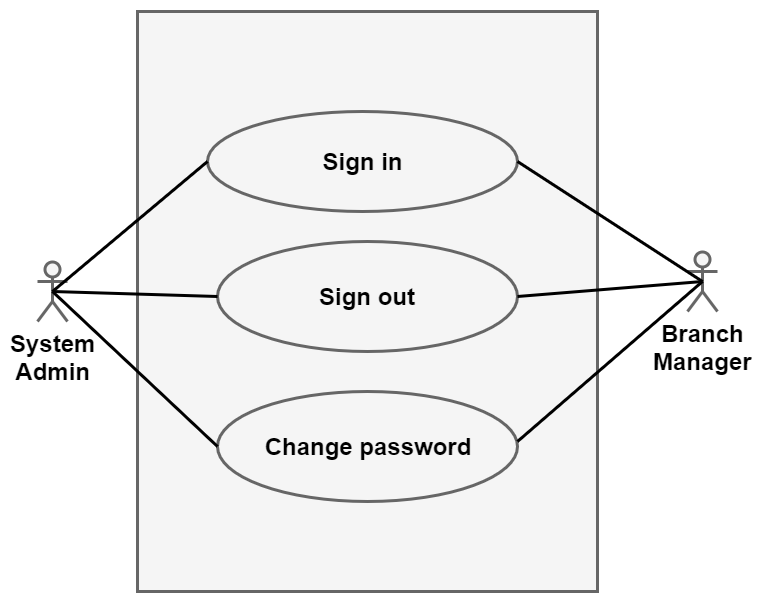


##### Figure 4.2: Level-1 Use Case Diagram

#### 4.2.2.1 Level-1.1 of Use Case:

Authentication sub-system is divided into three actions. They are log in, log out and change password. System admin and branch manager is connected with all the actions in this subsystem.

**Authentication**

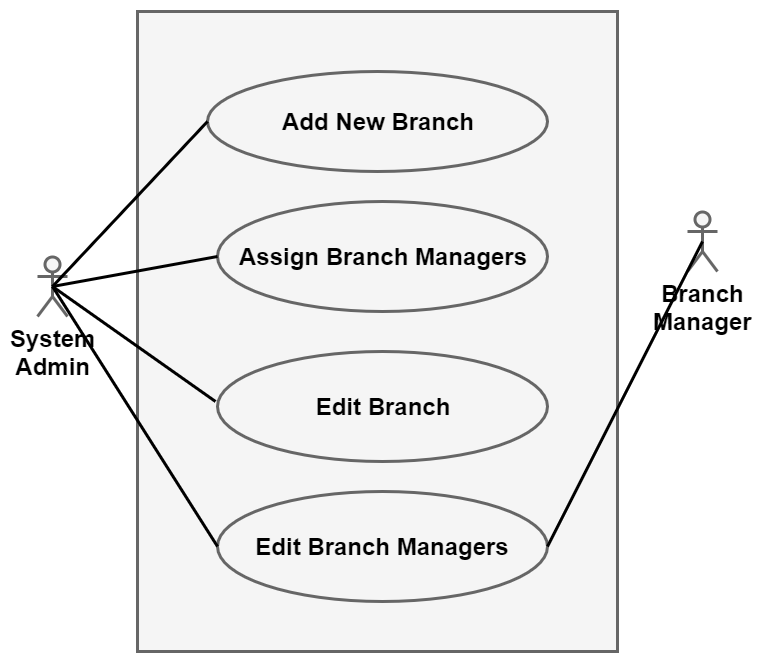


##### Figure 4.3: Level-1.1 Use Case Diagram

#### 4.2.2.2 Level-1.2 of Use Case:

Branch management sub-system is divided into four actions. Here System admin is connected with all the actions and branch manager is connected only with edit branch manager action.

**Branch Management**

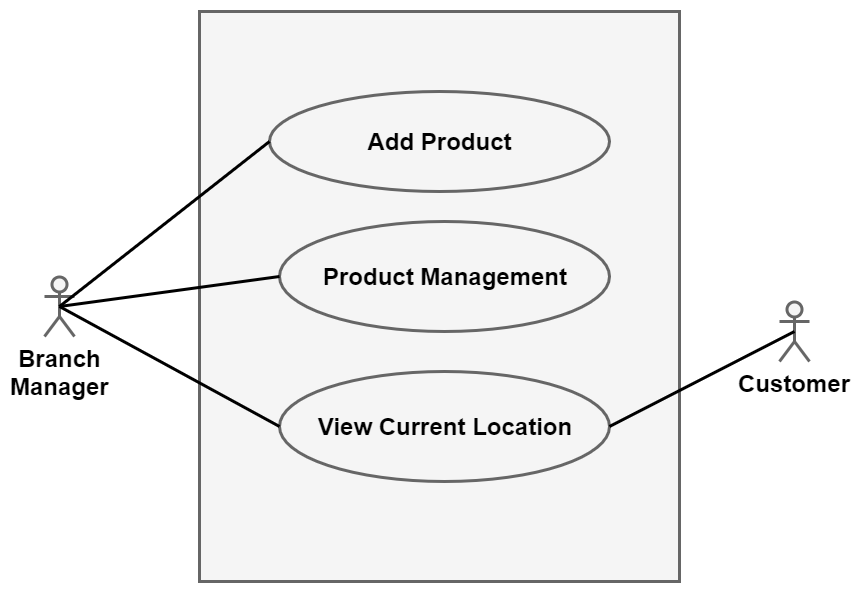


##### Figure 4.4: Level-1.2 Use Case Diagram

#### 4.2.2.3 Level-1.3 of Use Case:

Here the system has three actions where branch manager is connected with all the actions and customer is connected only with view current location action.

**Parcel & Money Order Processing**

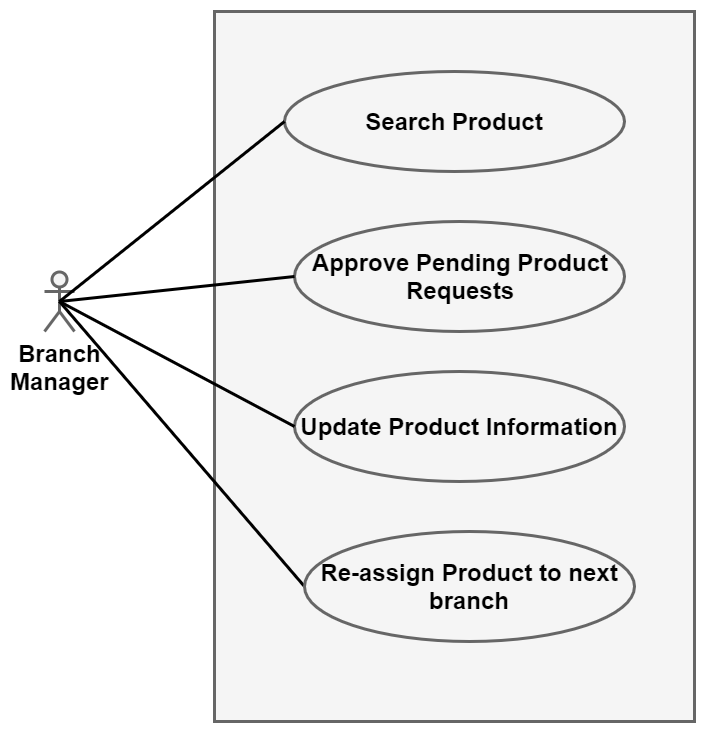
****

##### Figure 4.5: Level-1.3 Use Case Diagram

#### 4.2.2.3.1 Level-1.3 of Use Case:

Action Product Management is divided into four sub-actions. All that are connected with actor branch manager.

**Product Management**

****

##### Figure 4.6: Level-1.3.1 Use Case Diagram

## 4.3 Activity & Swim Lane Diagram

**Activity Diagram:**

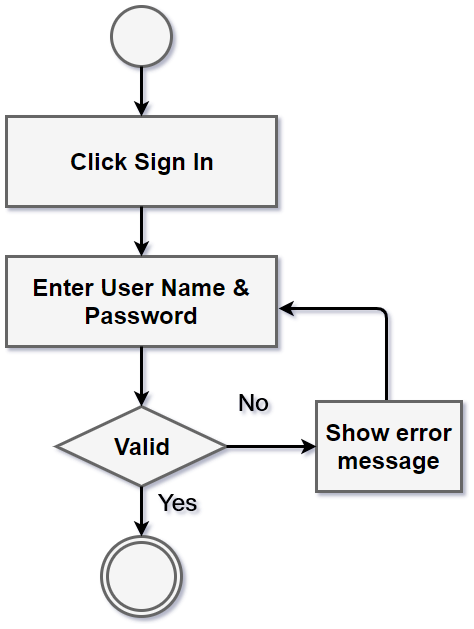
Activity diagramshows the technical view of the system for every use case  
from which we can understand how the system actually works and how the actors interact with the system.

**Swim Lane Diagram:**

Swim lane diagram of a specific activity diagram shows the responsibilities of each actor dividing them into lanes. From this diagram we can improve our understanding about how the system works and which actors play what role.

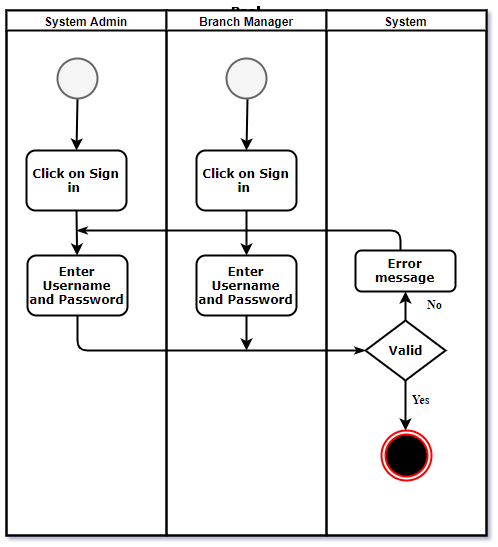
**Use Case: Sign In**

**Activity Diagram:**

****

##### Figure 4.7: Activity Diagram for Sign In

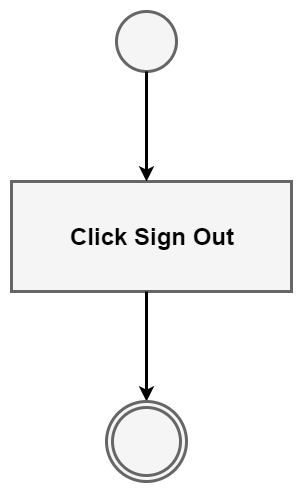
**Swim Lane Diagram:**

****

##### Figure 4.8: Swim Lane Diagram for Sign In

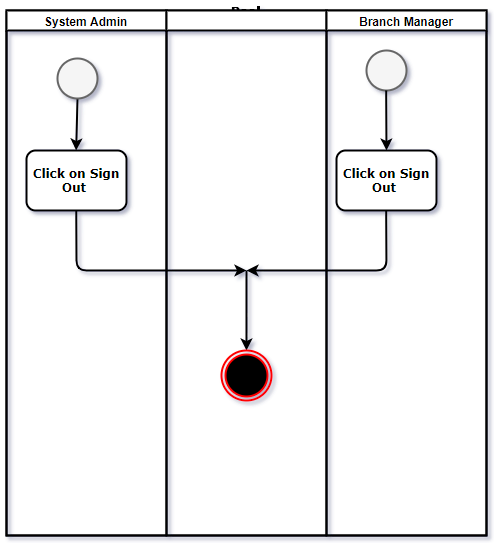
**Use Case: Sign Out**

**Activity Diagram:**

****

##### Figure 4.9: Activity Diagram for Sign Out

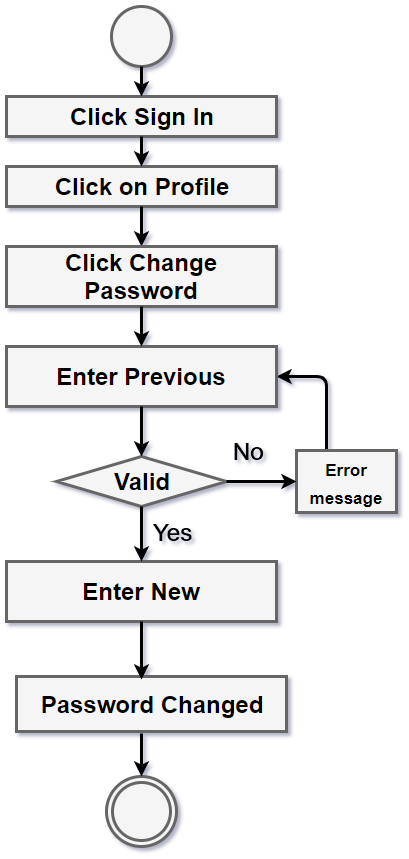
**Swim Lane Diagram:**



##### Figure 4.10: Swim Lane Diagram for Sign Out

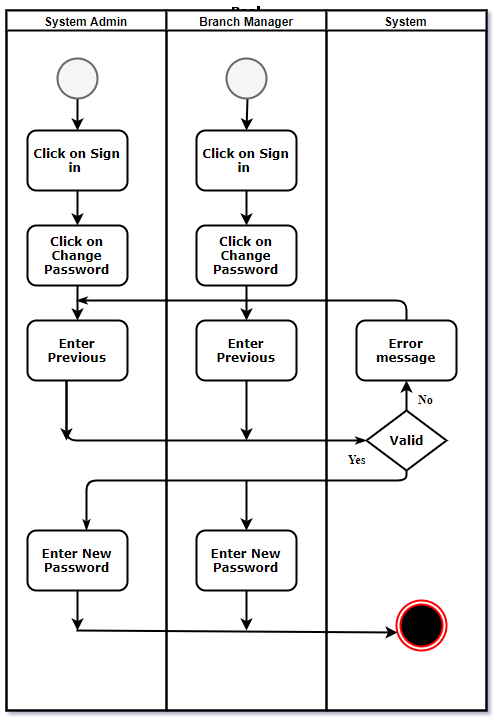
**Use Case: Change Password**

**Activity Diagram:**

****

##### Figure 4.11: Activity Diagram for Password Change

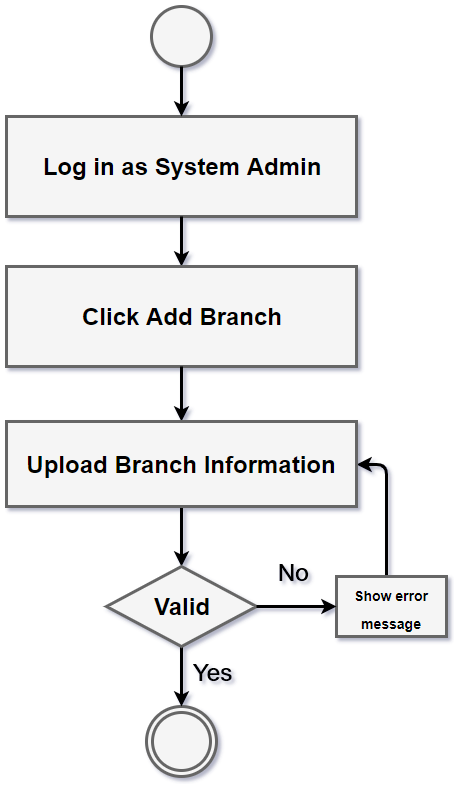
**Swim Lane Diagram:**



##### Figure 4.12: Swim Lane Diagram for Password Change

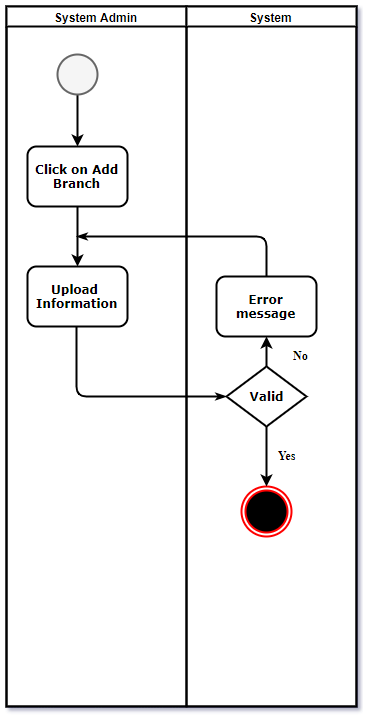
**Use Case: Add Branch**

**Activity Diagram:**

****

##### Figure 4.13: Activity Diagram for Add Branch

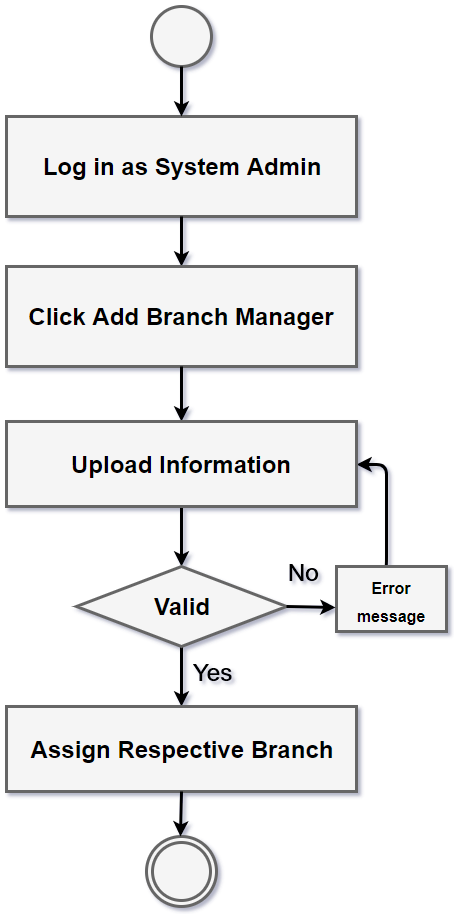
**Swim Lane Diagram:**

****

##### Figure 4.14: Swim Lane Diagram for Add Branch

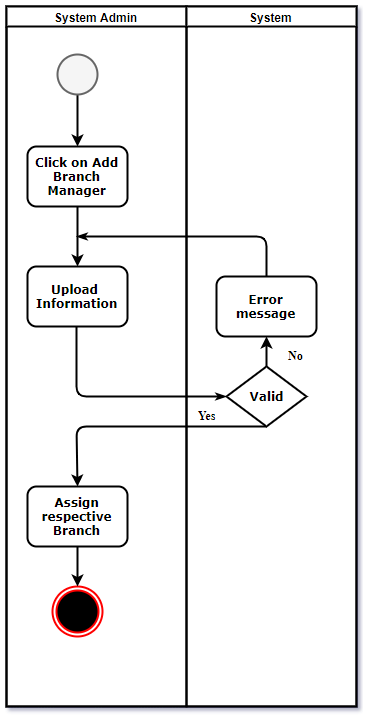
**Use Case: Assign Managers**

**Activity Diagram:**

****

##### Figure 4.15: Activity Diagram for Assign Managers

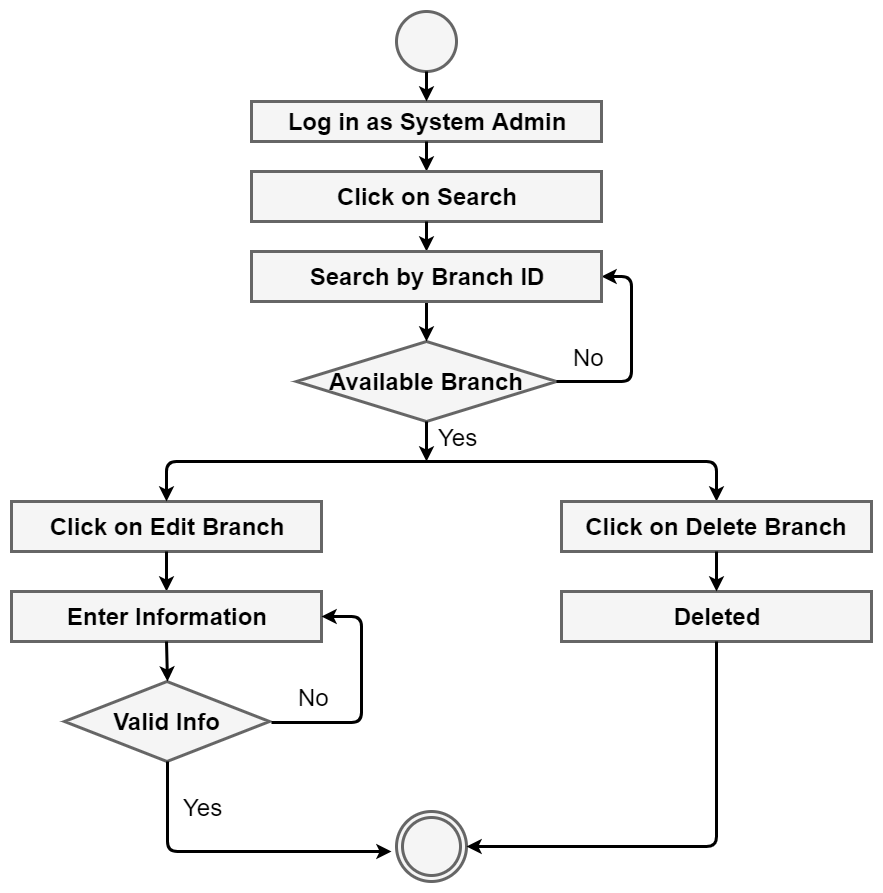
**Swim Lane Diagram:**

****

##### Figure 4.16: Swim Lane Diagram for Assign Managers

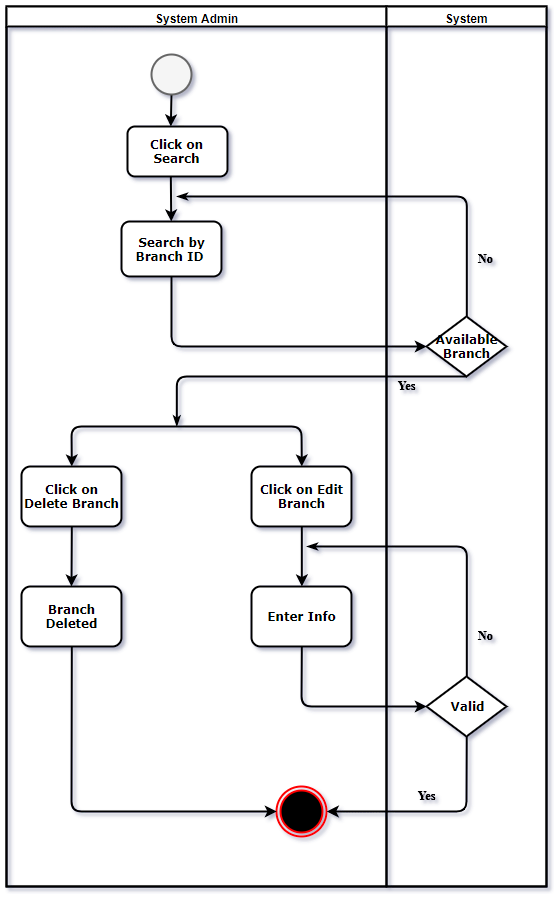
**Use Case: Edit Branch**

**Activity Diagram:**

****

##### Figure 4.17: Activity Diagram for Edit Branch

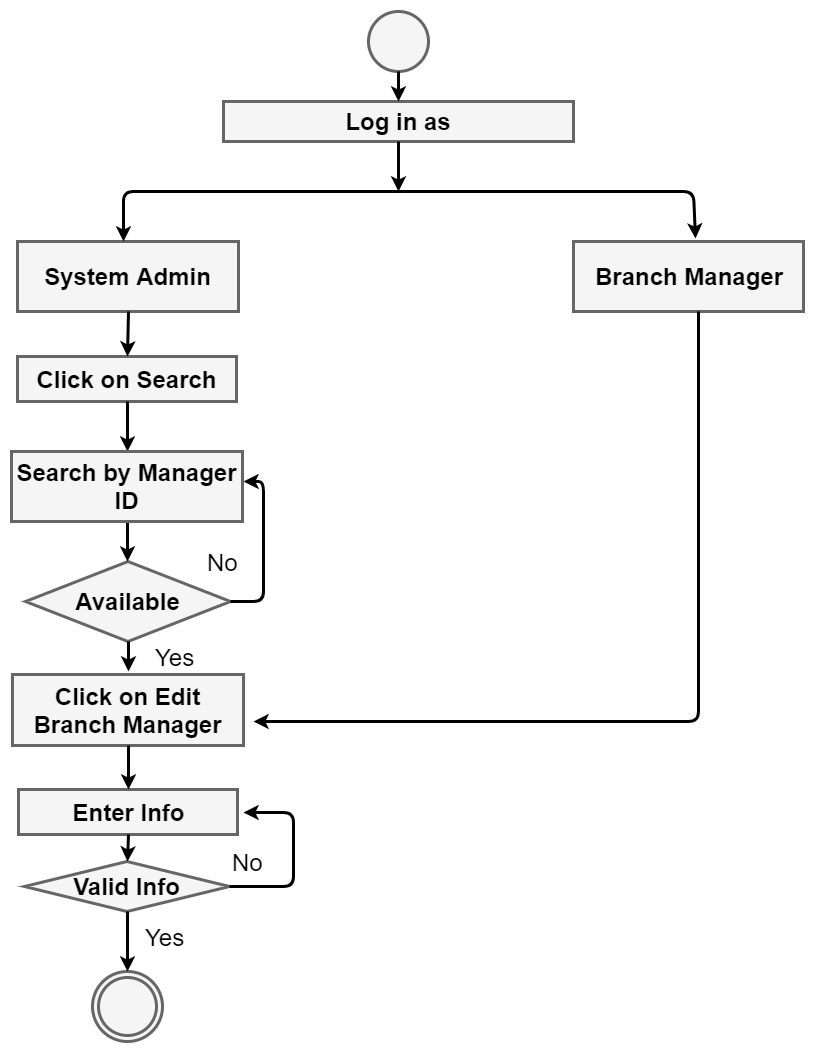
**Swim Lane Diagram:**



##### Figure 4.18: Swim Lane Diagram for Edit Branch

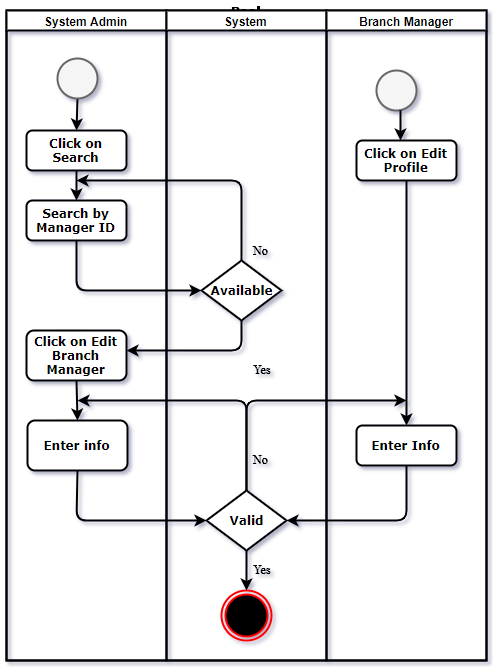
**Use Case: Edit Branch Managers**

**Activity Diagram:**

****

##### Figure 4.19: Activity Diagram for Edit Branch Managers

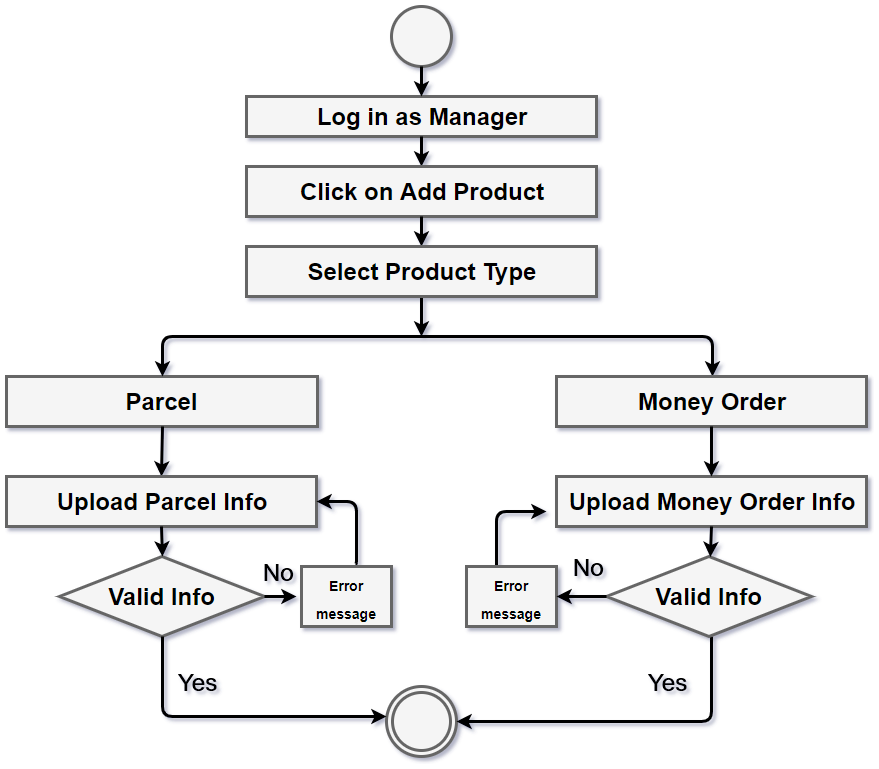
**Swim Lane Diagram:**



##### Figure 4.20: Swim Lane Diagram for Edit Branch Managers

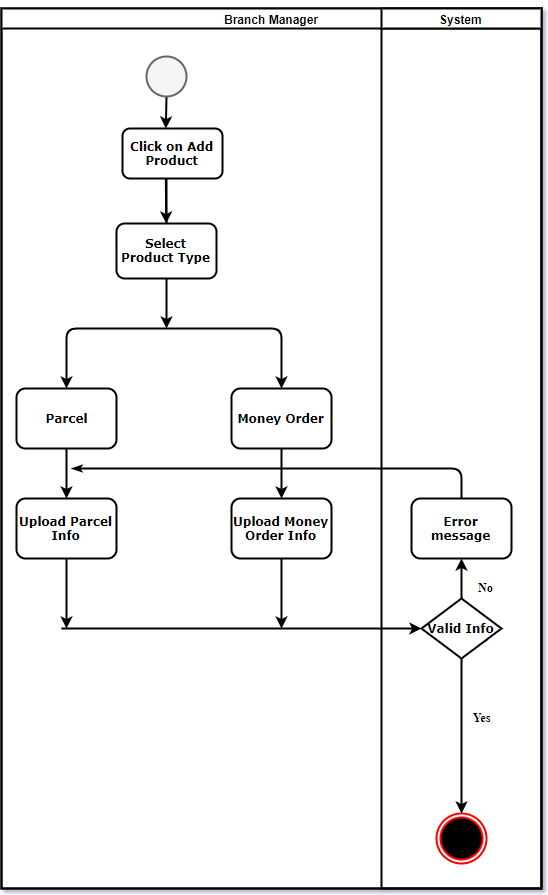
**Use Case: Add Product**

**Activity Diagram:**

****

##### Figure 4.21: Activity Diagram for Add Product

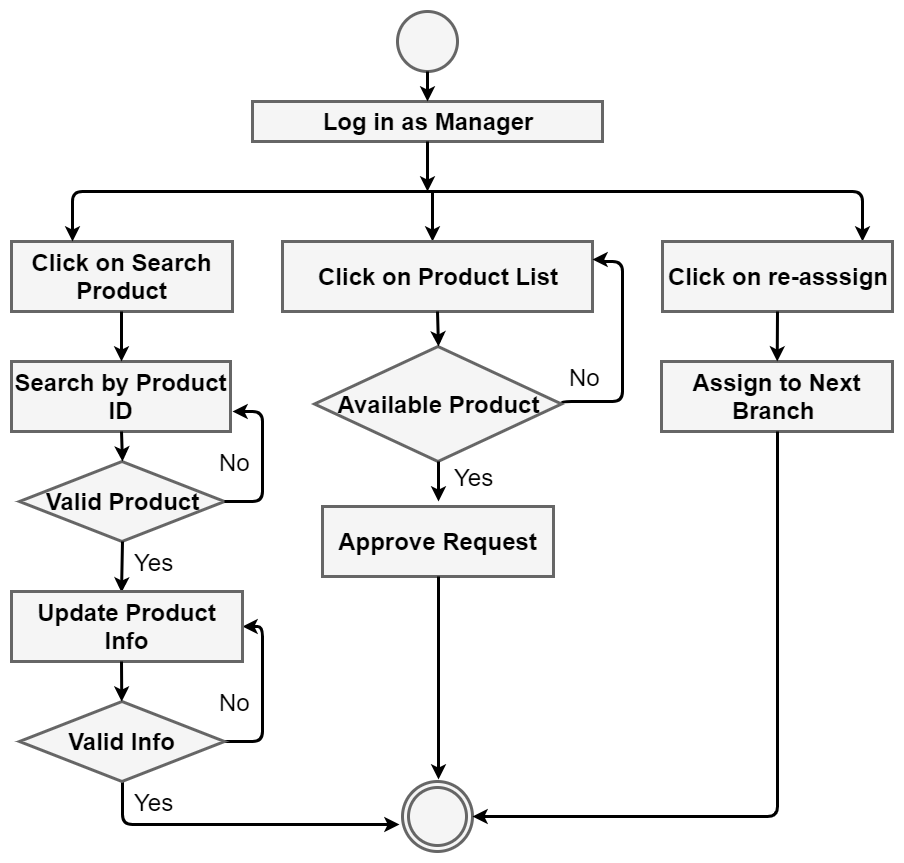
**Swim Lane Diagram:**

****

##### Figure 4.22: Swim Lane Diagram for Add Product

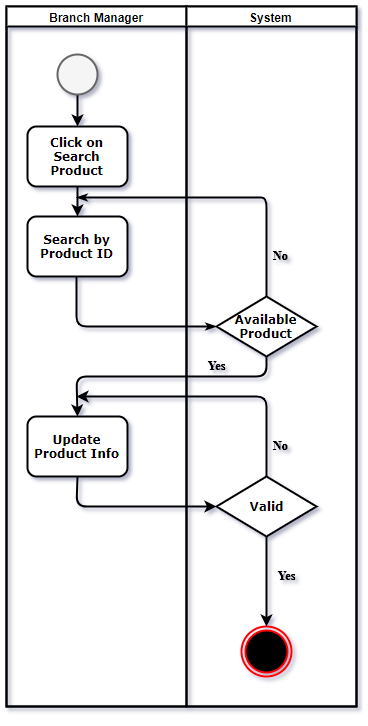
**Use Case: Product Management**

**Activity Diagram:**



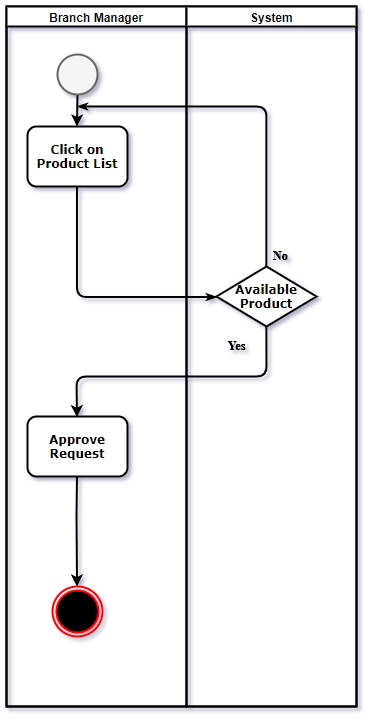
##### Figure 4.23: Activity Diagram for Product Management

**Swim Lane Diagram (Update Product):**

****

##### Figure 4.24: Swim Lane Diagram for Update Product

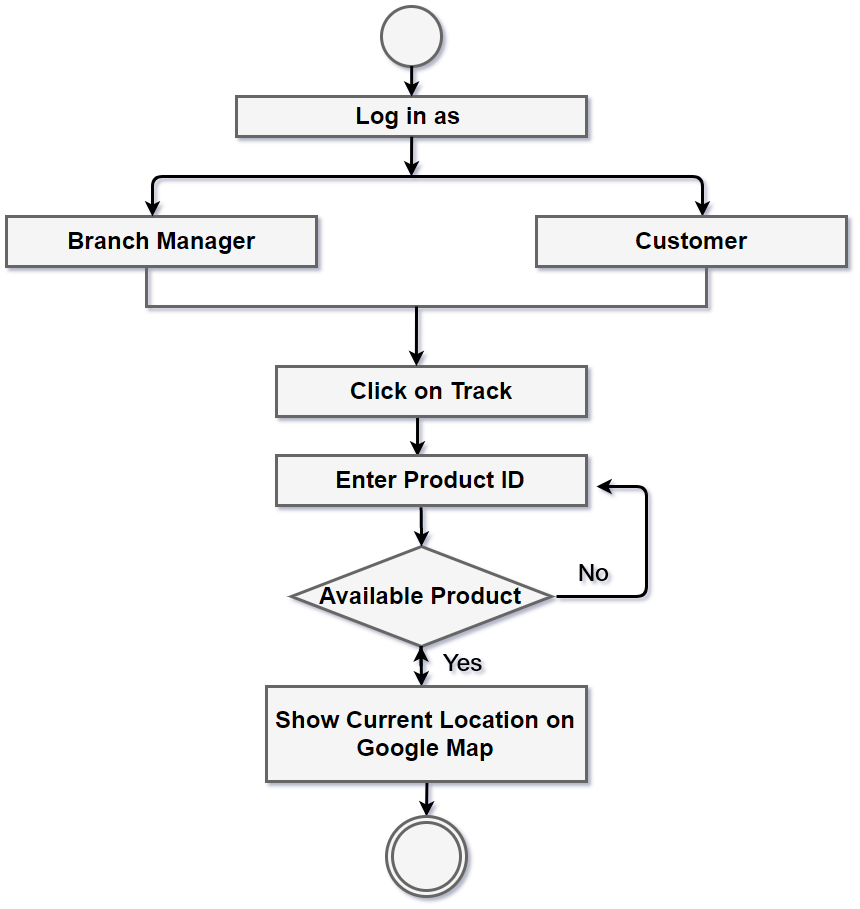
**Swim Lane Diagram (Approve Request):**

****

##### Figure 4.25: Swim Lane Diagram for Approve Request

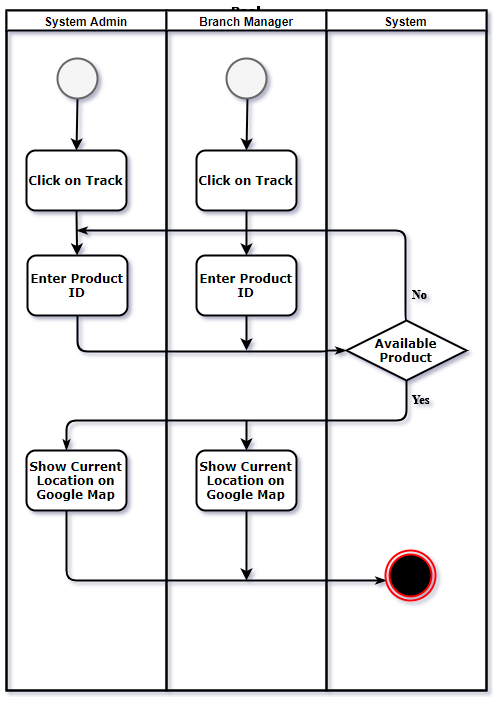
**Use Case: View Location**

**Activity Diagram:**

****

##### Figure 4.26: Activity Diagram for View Current Location

**Swim Lane Diagram:**

****

##### Figure 4.27: Swim Lane Diagram for View Current Location

# **Chapter 5**

# **Data Model**

## 5.1 Data Modeling Concept

If software requirements include the need to create, extend, or interface with a database or if complex data structures must be constructed and manipulated, a software team may choose to create a data model as part of overall requirements modeling.

## 5.2 Data Objects

A data object is representation of composite information that must be understood by software. Here, composite information means that has a number of different properties or attributes. A data object can be an external entity, a thing, an occurrence, a role, an organizational unit, a place or a structure.

### 5.2.1 Grammatical Parsing (Noun Identity)

|  |  |  |  |
| --- | --- | --- | --- |
| NID | Noun | Problem/solution space | Attributes |
|  | Post Office Management System | P | --- |
|  | Parcel | P | --- |
|  | Money Order | P | --- |
|  | Branch Details | S | branch\_ID, 5, 6, 7 |
|  | Zilla/Upzilla | S | --- |
|  | Address | S | --- |
|  | Post Code | S | --- |
|  | Branch Manager | S | manager\_ID,10,11,12,13,  14,15,16 |
|  | Personal Information | P | --- |
|  | First Name | S | --- |
|  | Last Name | S | --- |
|  | Phone Number | S | --- |
|  | Email Address | S | --- |
|  | Date of Birth | S | --- |
|  | Present Address | S | --- |
|  | Permanent Address | S | --- |
|  | Related Branch | P | --- |
|  | Product | S | product\_ID,12,19,20,21,22,  23,25,26,27, |
|  | Product Type | S | --- |
|  | Source Post Office | S | --- |
|  | Destination Post Office | S | --- |
|  | Weight | S | --- |
|  | Cost | S | --- |
|  | Status | P | --- |
|  | Amount | S | --- |
|  | Receiver’s Name | S | --- |
|  | PIN Number | S | --- |
|  | Customer | S | customer\_ID, 10,11,12,29 |
|  | Address | S | --- |
|  | System Admin | S | admin\_ID, 31 |
|  | Password | S | --- |
|  | Email | P | --- |
|  | Mobile Phone | P | --- |
|  | SMS | P | --- |
|  | Money | P | --- |
|  | Google Map | P | --- |
|  | Location | P | --- |
|  | Money Order Slip | P | --- |

### 5.2.2 Identify Data Objects

**Data Object: Branch Manager**

**Attributes:**

* Branch manager ID
* Password
* First name
* Last name
* Gender
* Phone number
* Email
* Date of birth
* Present address
* Permanent address
* Assigned branch

**Data Object: Product**

**Attributes:**

* Product ID
* Product type
* Customer ID
* Source post office
* Destination post office
* Weight
* Cost
* Status
* Money order amount
* Receiver’s name
* Receiver’s phone number
* PIN number

**Data Object: Customer**

**Attributes:**

* Customer ID
* Name
* Password
* Email
* Address
* Phone number
* Product ID

**Data Object: Branch**

**Attributes:**

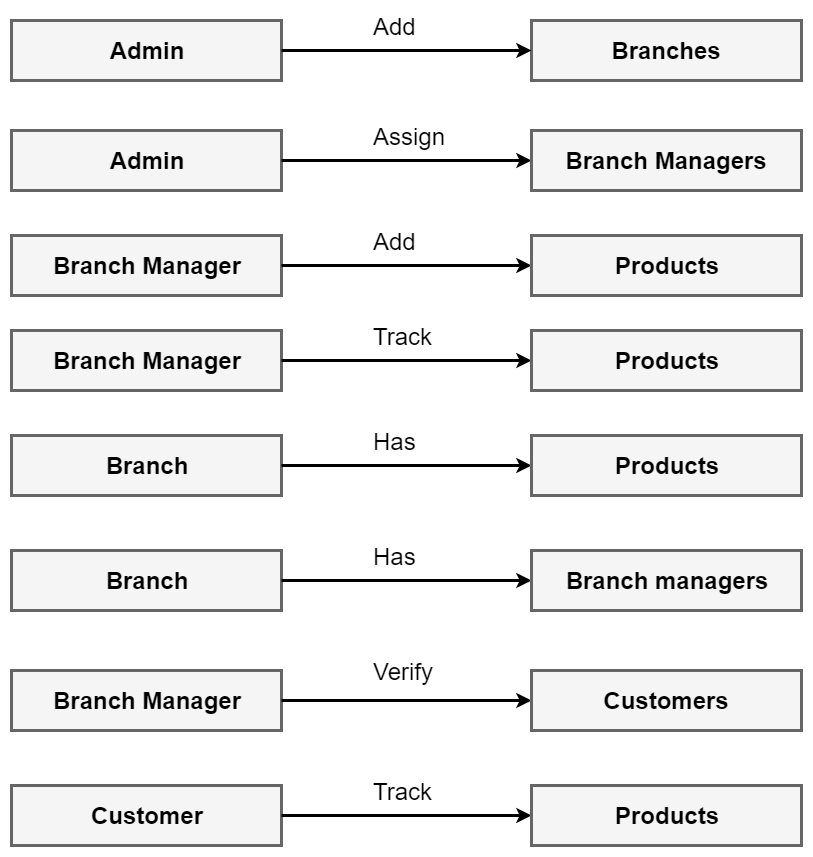
* Branch ID
* Branch name
* Address
* Manager ID
* Product ID
* Post code

**Data Object: Admin**

**Attributes:**

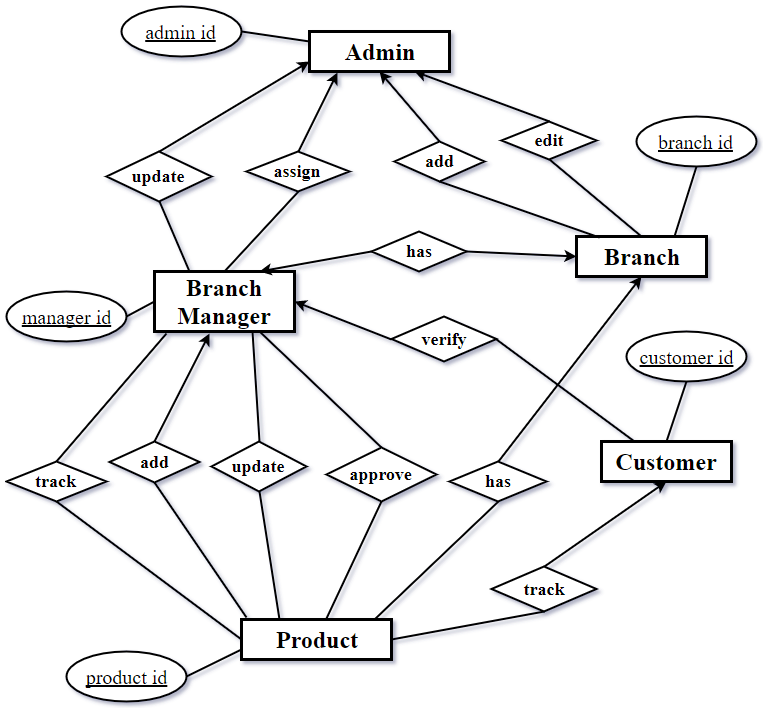
* Admin ID
* Password
* Branch ID
* Manager ID

### 5.2.3 Relationship Between Data Objects



##### Figure 5.1: Relationship between data objects

### 5.2.4 Entity Relationship Diagram



##### Figure 5.2: ER Diagram of Post Office Management System

### 5.2.5 Database Table

|  |  |
| --- | --- |
| **Branch Manager** | |
| Attributes | Type (Size) |
| 1. Branch manager ID 2. User name 3. Password 4. First name 5. Last name 6. Gender 7. Phone number 8. Email 9. Date of birth 10. Present address 11. Permanent address 12. Assigned branch | 1. Integer(10) 2. Varchar2(30) 3. Varchar2(20) 4. Varchar2(50) 5. Varchar2(50) 6. Boolean 7. Integer(15) 8. Varchar2(50) 9. Date 10. Varchar2(300) 11. Varchar2(300) 12. Varchar2(20) |

|  |  |
| --- | --- |
| **Product Status** | |
| Attributes | Type (Size) |
| 1. Status ID 2. Product ID 3. Manager ID | 1. Integer(10) 2. Integer(10) 3. Integer(10) |

|  |  |
| --- | --- |
| **Product** | |
| Attributes | Type (Size) |
| 1. Product ID 2. Product Type 3. Source post office 4. Destination post office 5. Weight 6. Cost 7. Status 8. Money order amount 9. Receiver’s name 10. Receiver’s phone number 11. PIN number | 1. Integer(10) 2. Boolean 3. Varchar2(50) 4. Varchar2(50) 5. Float(20) 6. Integer(10) 7. Boolean 8. Integer(15) 9. Varchar2(50) 10. Integer(15) 11. Integer(20) |

|  |  |
| --- | --- |
| **Branch** | |
| Attributes | Type (Size) |
| 1. Branch ID 2. Branch name 3. Address 4. Post code | 1. Integer(10) 2. Varchar2(30) 3. Varchar2(300) 4. Integer(5) |

|  |  |
| --- | --- |
| **System Admin** | |
| Attributes | Type (Size) |
| 1. Admin ID 2. User name 3. Password | 1. Integer(10) 2. Varchar2(20) 3. Varchar2(20) |

|  |  |
| --- | --- |
| **Customer** | |
| Attributes | Type (Size) |
| 1. Customer ID 2. Name 3. Password 4. Email 5. Address 6. Phone Number | 1. Integer(10) 2. Varchar2(30) 3. Varchar2(20) 4. Varchar2(50) 5. Varchar2(300) 6. Integer(15) |

# **Chapter 6**

# **Class Based Model**

## 6.1 Introduction

Class-based modelling represents the objects that the system will manipulate, the operations that will be applied to the objects, relationships between the objects and the collaborations that occur between the classes that are defined.

## 6.2 General Classification

|  |  |
| --- | --- |
| **Property Name** | **ID (G.C)** |
| External entity | 1 |
| Things | 2 |
| Occurrences | 3 |
| Roles | 4 |
| Organizational units | 5 |
| Places | 6 |
| Structures | 7 |

## 6.3 Selection Criteria

|  |  |
| --- | --- |
| **Property Name** | **ID (S.C)** |
| Retained information | 1 |
| Needed service | 2 |
| Multiple attributes | 3 |
| Common attributes | 4 |
| Common operations | 5 |
| Essential requirements | 6 |

## 

## 6.4 Class Identification

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **NID** | **Noun(Potential Class)** | **Problem/solution space** | **General Classification** | **Special Criterial** |
|  | Post Office Management System | P | --- | --- |
|  | Parcel | P | --- | --- |
|  | Money Order | P | --- | --- |
| 1. \* | Branch Details | S | 7 | 1,3,6 |
|  | Zilla/Upzilla | S | --- | --- |
|  | Address | S | --- | --- |
|  | Post Code | S | --- | --- |
| 1. \* | Branch Manager | S | 4 | 1,3,4,5,6 |
|  | Personal Information | P | --- | --- |
|  | First Name | S | --- | --- |
|  | Last Name | S | --- |  |
|  | Phone Number | S | --- | --- |
|  | Email Address | S | --- | --- |
|  | Date of Birth | S | --- | --- |
|  | Present Address | S | --- | --- |
|  | Permanent Address | S | --- | --- |
|  | Related Branch | P | --- | --- |
| 1. \* | Product | S | 7 | 1,2,3,4,5,6 |
|  | Product Type | S | --- | --- |
|  | Source Post Office | S | --- | --- |
|  | Destination Post Office | S | --- | --- |
|  | Weight | S | --- | --- |
|  | Cost | S | --- | --- |
|  | Status | P | --- | --- |
|  | Amount | S | --- | --- |
|  | Receiver’s Name | S | --- | --- |
|  | PIN Number | S | --- | --- |
| 1. \* | Customer | S | 4 | 2,3,4,6 |
|  | Address | S | --- | --- |
| 1. \* | System Admin | S | 4 | 1,2,4,5 |
|  | Password | S | --- | --- |
|  | Email | P | --- | --- |
|  | Mobile Phone | P | --- | --- |
|  | SMS | P | --- | --- |
|  | Money | P | --- | --- |
|  | Google Map | P | --- | --- |
|  | Location | P | --- | --- |
|  | Money Order Slip | P | --- | --- |

Star (\*) sign indicates that there are selected for class. So, the classes are

1. System Admin
2. Branch Manager
3. Product
4. Customer
5. Branch

## 6.5 Class Card:

|  |  |
| --- | --- |
| **Branch Manager** | |
| Attributes | Methods |
| 1. Branch manager ID 2. User name 3. Password 4. First name 5. Last name 6. Gender 7. Phone number 8. Email 9. Date of birth 10. Present address 11. Permanent address 12. Assigned branch | 1. add\_product() 2. update\_product\_info() 3. approve\_product() 4. verify() 5. track\_product() 6. update\_personal\_info() |
| Responsibilities | Collaborator Class |
| 1. Add parcel or money order 2. Update product information 3. Approve pending product request 4. Verify senders or receivers 5. Track product current location 6. Update personal information | 1. Database 2. Product 3. Customers |

|  |  |
| --- | --- |
| **Product** | |
| Attributes | Methods |
| 1. Product ID 2. Product type 3. Source post office 4. Destination post office 5. Weight 6. Cost 7. Status 8. Money order amount 9. Receiver’s name 10. Receiver’s phone number 11. PIN number | 1. getter/setter() |
| Responsibilities | Collaborator Class |
|  |  |

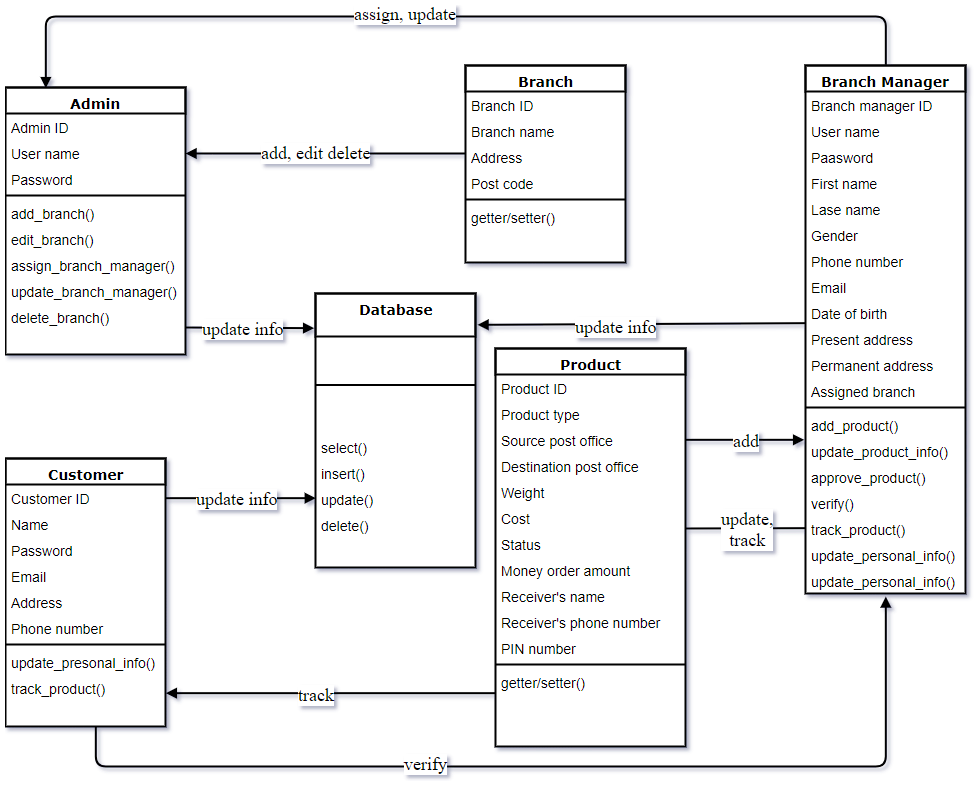
|  |  |
| --- | --- |
| **Branch** | |
| Attributes | Methods |
| 1. Branch ID 2. Branch name 3. Address 4. Post code | 1. getter/setter() |
| Responsibilities | Collaborator Class |
|  |  |

|  |  |
| --- | --- |
| **System Admin** | |
| Attributes | Methods |
| 1. Admin ID 2. User name 3. Password | 1. add\_branch() 2. edit\_branch() 3. assign\_branch\_manager() 4. update\_branch\_manager() 5. delete\_branch() |
| Responsibilities | Collaborator Class |
| 1. Add new post office branch 2. Search and edit specific branch details 3. Assign managers to respective branches 4. Search and update branch managers 5. Delete any branch | 1. Branch 2. Branch manager 3. Database |

|  |  |
| --- | --- |
| **Database** | |
| Attributes | Methods |
|  | 1. select() 2. insert() 3. update() 4. delete() |
| Responsibilities | Collaborator Class |
| 1. Retrieving information 2. Storing information |  |

|  |  |
| --- | --- |
| **Customer** | |
| Attributes | Methods |
| 1. Customer ID 2. Name 3. Password 4. Email 5. Address 6. Phone number | 1. Update\_personal\_information() 2. track\_product() |
| Responsibilities | Collaborator Class |
| 1. Update customer information 2. Tack current location of the product | 1. Database 2. Product |

## 6.6 Class Diagram



##### Figure 6.1: Class Diagram

# **Chapter 7**

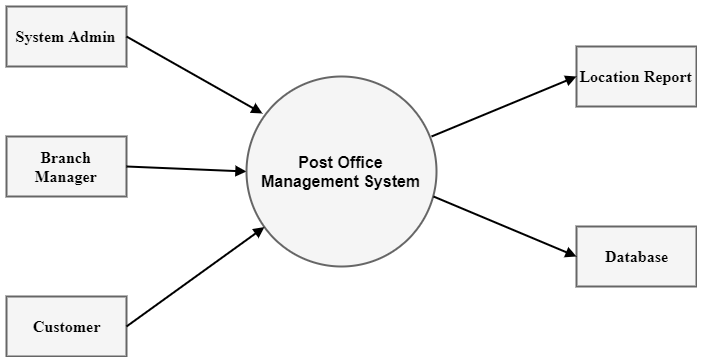
# **Flow-Oriented Model**

## 7.1 Introduction

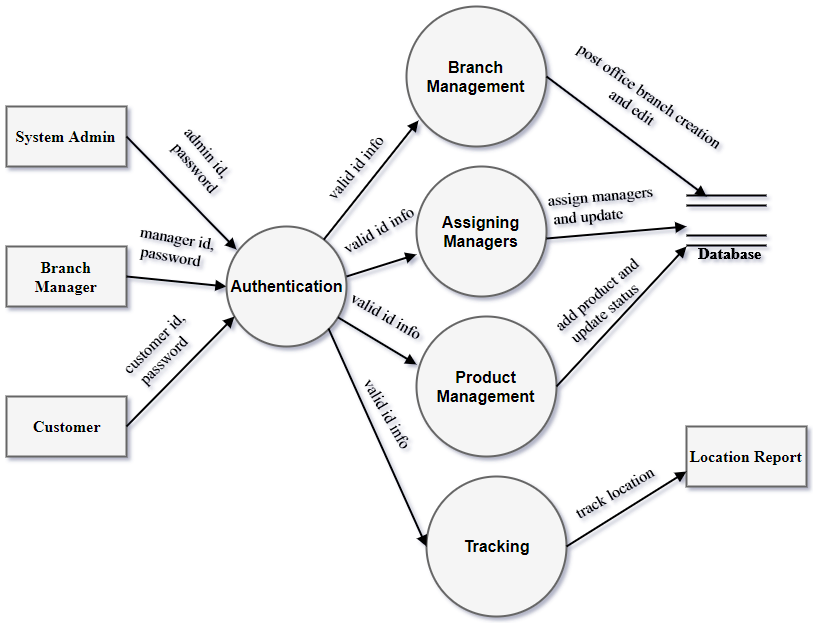
Although data flow-oriented modeling is perceived as an outdated technique by some software engineers, it continues to be one of the most widely used requirements analysis notations in use today.

## 7.2 Data flow diagram

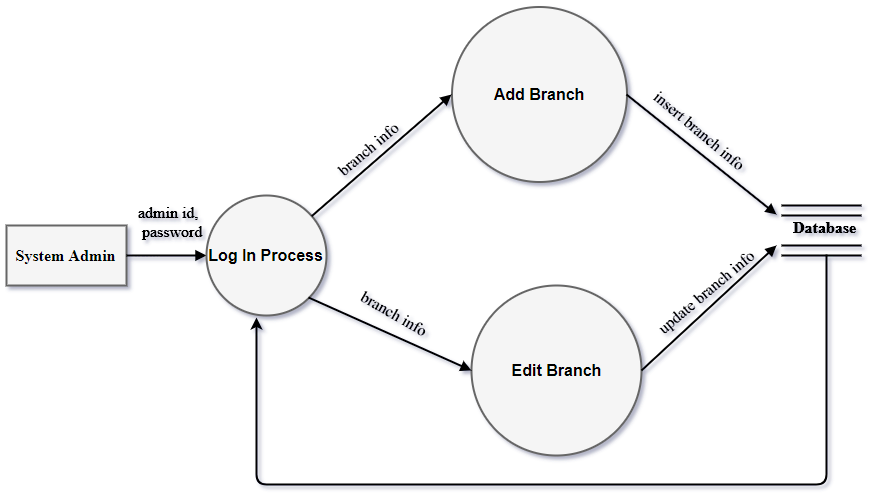
The Data Flow Diagram (DFD) takes an input-process-output view of a system. Data objects flow into the software, are transformed by processing elements and resultant data objects flow out of the software. Data objects are represented by labeled arrows and transformations are represented by circles.



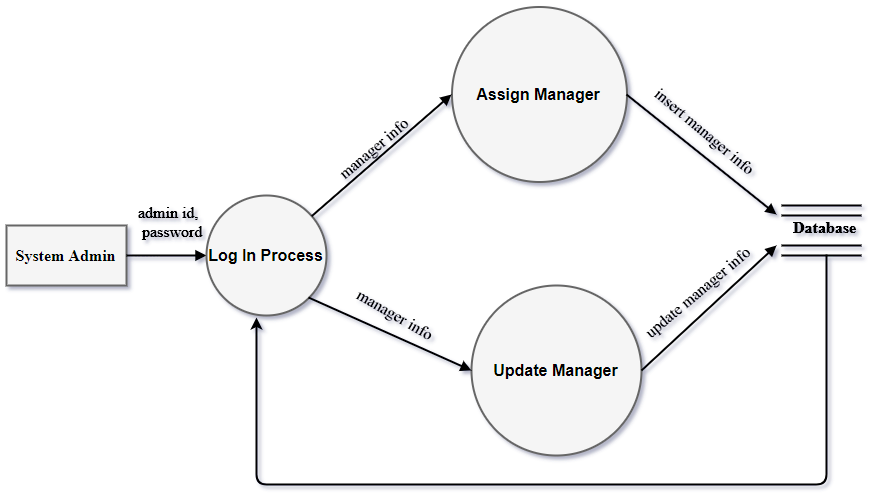
##### Figure 7.1: Level-0 Data Flow Diagram



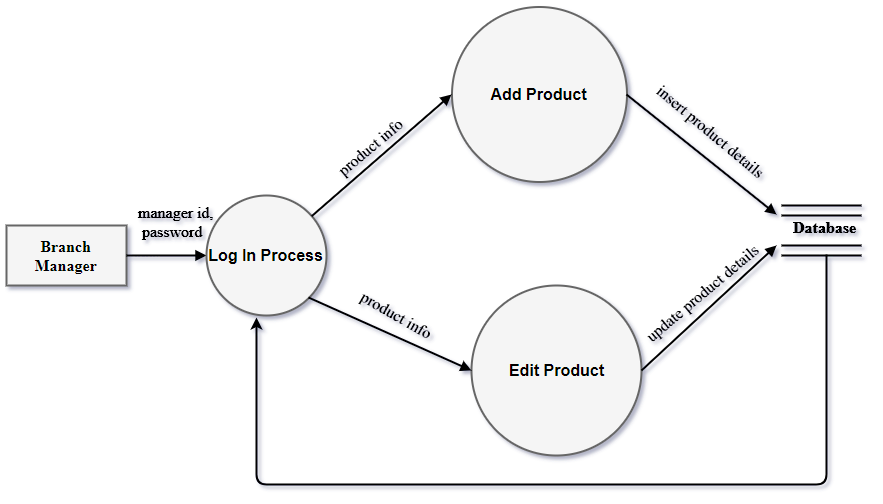
##### Figure 7.2: Level-1 Data Flow Diagram



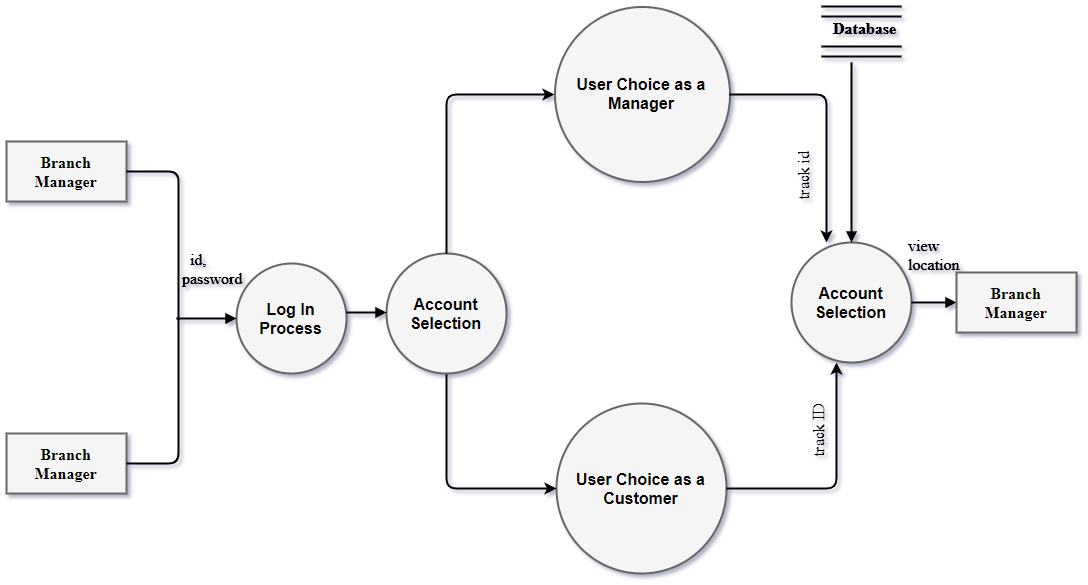
##### Figure 7.3: Level-1.1 Data Flow Diagram



##### Figure 7.4: Level-1.2 Data Flow Diagram



##### Figure 7.5: Level-1.3 Data Flow Diagram



##### Figure 7.6: Level-1.4 Data Flow Diagram

# **Chapter 8**

# **Behavioral Model**

## 8.1 Introduction

Behavior modeling is also referred to as State modeling, State machines and State transition matrix. Behavior modeling is when one thinks of his ideas in terms of states and transitions. This requires both identifying all of the interesting states of being that software or its components are likely to be in. And also, at a high level, abstracting what events are likely to cause software or its components to change between states of being.

## 8.2 Identifying Events

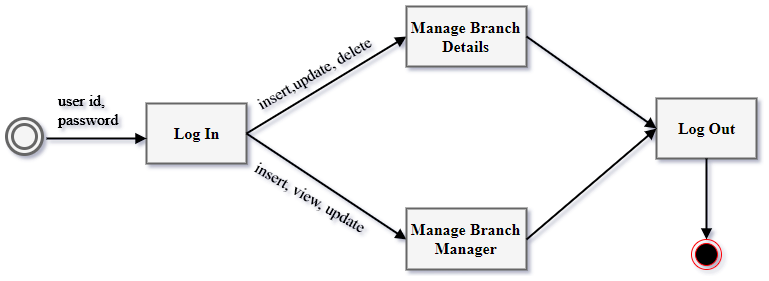
Here we have identified events from the usage scenario and listed their corresponding initiators and collaborators:

|  |  |  |
| --- | --- | --- |
| Events | Initiator | Collaborator |
| Add Branch | System Admin | Branch |
| Assign Managers | System Admin | Branch Manager |
| Edit Branch | System Admin | Branch |
| Add Product | Branch Manager | Product |
| Update Product | Branch Manager | Product |
| Verify Customer | Branch Manager |  |
| Track Product | Manager, Customer | Product |
| Update Manager | System Admin | Branch manager |

## 8.3 State Transition Diagram

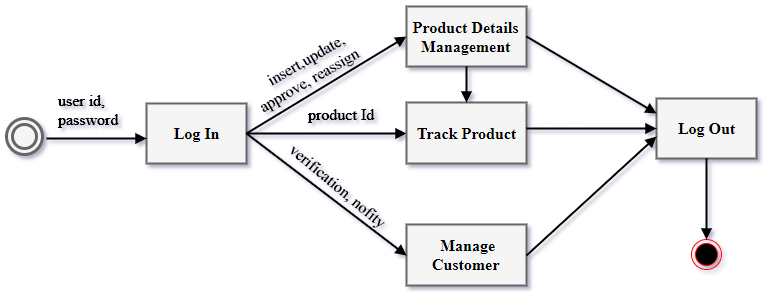
State transition diagram represents active states for each class where the arrows indicate events that cause transition from one state to another. Here, state transition for each class of Blood Management Automated System is given below:

**System Admin:**

****

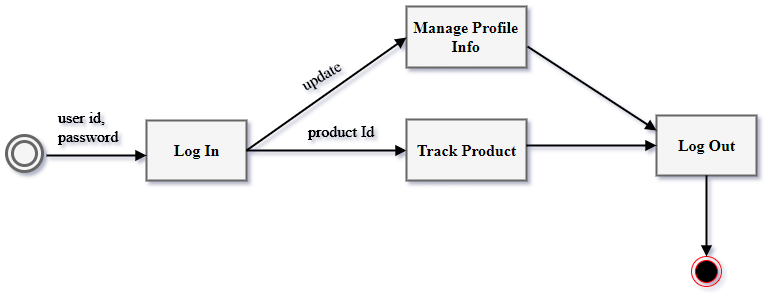
##### Figure 8.1: State Transition Diagram for System Admin

**Branch Manager:**

****

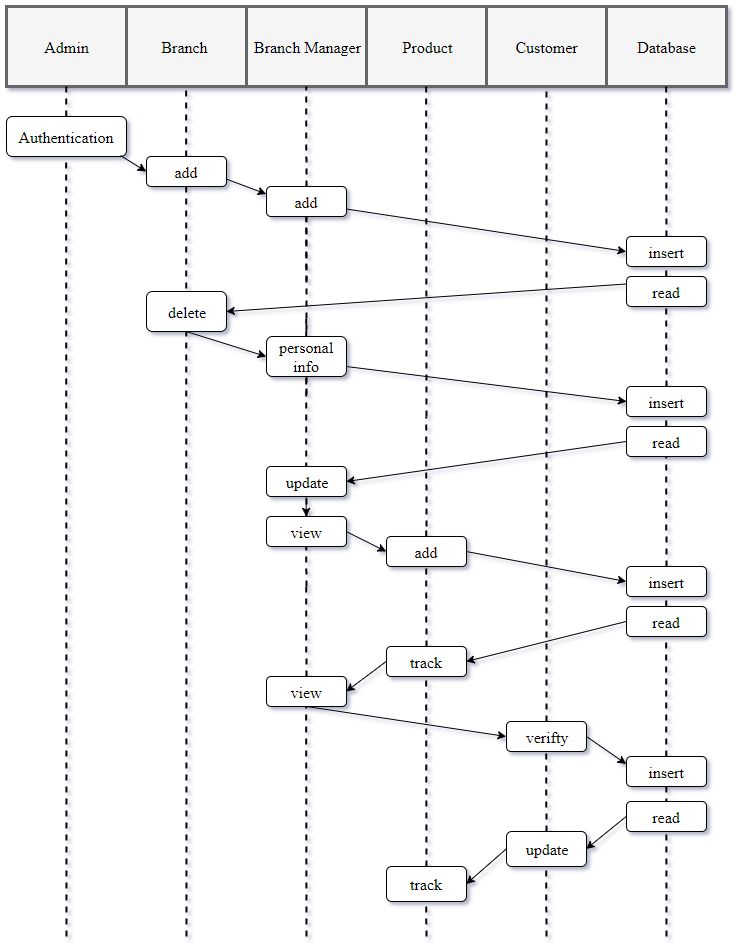
##### Figure 8.2: State Transition Diagram for Branch Manager

**Customer:**

****

##### Figure 8.3: State Transition Diagram for Customer

## 8.4 Sequence Diagram



##### Figure 8.4: Sequence Diagram

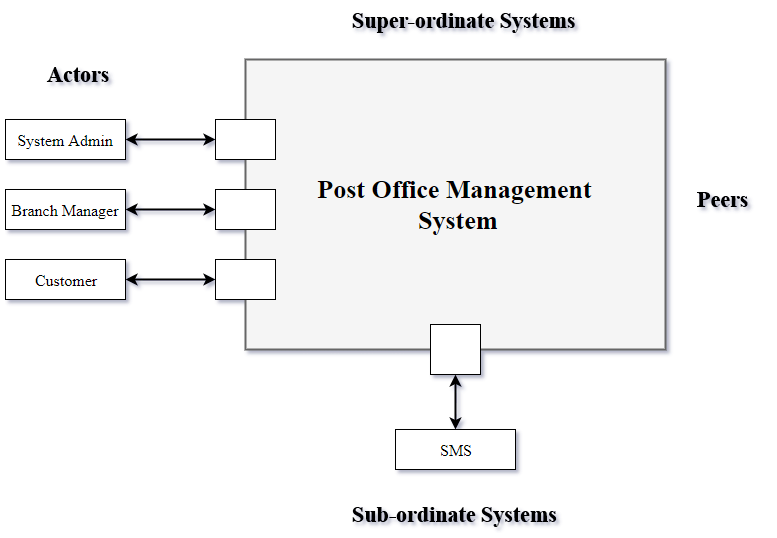
# **Chapter 9**

# **Software Design**

## 9.1 Architectural Design

Software architecture refers to the high level structures of a software system, the discipline of creating such structures, and the documentation of these structures. Software architectural design has been followed in Post Office Management System. At the beginning of the architectural design, the context diagram of the System is defined. Then the archetype of the project is described. And after the finding the archetypes, the components and the classes are defined.

### 9.1.1 Representing the system in context



##### Figure 9.1: Architectural Context Diagram

### 9.1.2 Define Archetypes

1. Authentication
2. Branch Management
3. Product Management
4. Tracking System

### 9.1.3 Refining Archetypes into Components

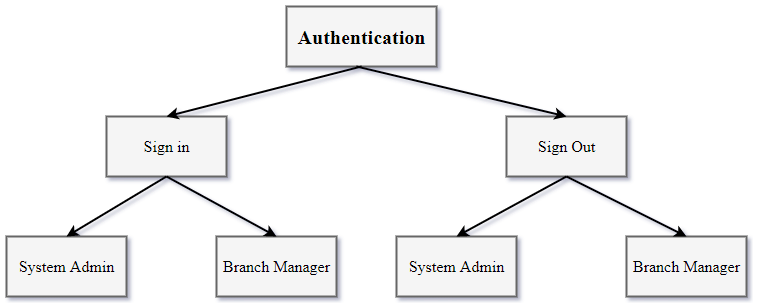
**Components:**

1. Sign in, Sign Out, DAL
2. Add Branch, Edit Branch, Assign Branch Manager, Update manager, DAL
3. Add Product, Edit Product, Approve Request, Track Product, DAL
4. Tract Current Location

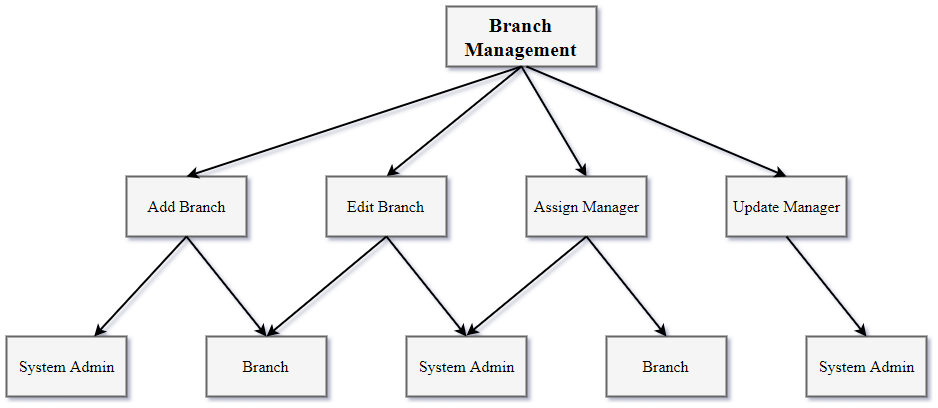
**Classes:**

1. System Admin, Branch Manager
2. System Admin, Branch
3. Branch Manager, Product
4. Branch Manager, Customer

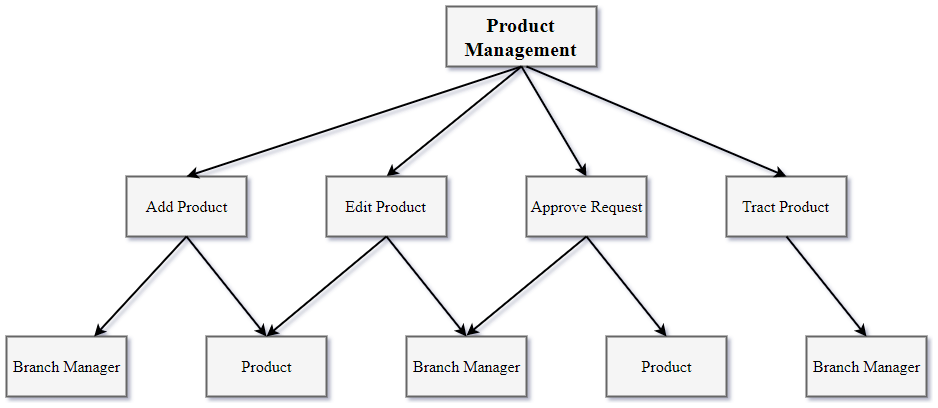
### 9.1.4 Describing Instantiation of the System



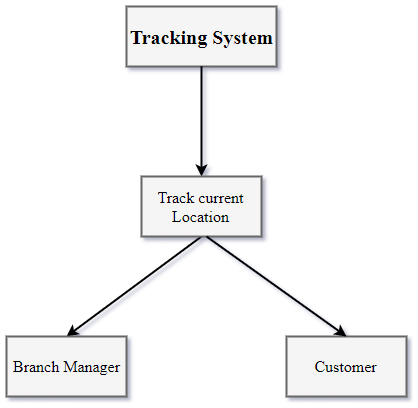
##### Figure 9.2: Refining ‘Authentication’ archetype into components and classes



##### Figure 9.3: Refining ‘Branch Management’ archetype into components and classes



##### Figure 9.4: Refining ‘Product Management’ archetype into components and classes



##### Figure 9.5: Refining ‘Tracking System’ archetype into components and classes

## 9.2 Component Level Design

### 9.2.1 Identifying all design classes that correspond to the problem domain

The Classes are:

1. System Admin
2. Branch Manager
3. Product
4. Branch

### 9.2.2 Identify all design classes that correspond to the infrastructure domain

1. Database

### 9.2.3 Elaborate all design classes that are not acquired as reusable components

**1. Analysis Class 2. Design Component**

|  |
| --- |
| Branch Manager |
| Branch manager ID  User name  Password  First name  Last name  Gender  Phone number  Email  Date of birth  Present address  Permanent address  Assigned branch |
| add\_product()  update\_product\_info()  approve\_product()  verify()  track\_product()  update\_personal\_info() |

**Branch Manager**

Processing Product

Information Management

**3. Elaborated Design Class**

|  |
| --- |
| Branch Manager |
| Branch manager ID  User name  Password  First name  Last name  Gender  Phone number  Email  Date of birth  Present address  Permanent address  Assigned branch |
| add\_product()  update\_product\_info()  approve\_product()  verify()  track\_product()  update\_personal\_info() |

|  |
| --- |
| <<interface>>  Product Processing |
| add\_product()  update\_product\_info()  approve\_product()  track\_product() |

|  |
| --- |
| <<interface>>  Information Management |
| update\_personal\_info() |

**1. Analysis Class 2. Design Component**

|  |
| --- |
| System Admin |
| Admin ID  User name  Password |
| add\_branch()  edit\_branch()  assign\_branch\_manager()  update\_branch\_manager()  delete\_branch() |

**System Admin**

Branch Management

Manager Recruitment

**3. Elaborated Design Class**

|  |
| --- |
| System Admin |
| Admin ID  User name  Password |
| add\_branch()  edit\_branch()  assign\_branch\_manager()  update\_branch\_manager()  delete\_branch() |

|  |
| --- |
| <<interface>>  Branch Management |
| add\_branch()  edit\_branch()  delete\_branch() |

|  |
| --- |
| <<interface>>  Manager Recruitment |
| assign\_branch\_manager()  update\_branch\_manager() |

**1. Analysis Class 2. Design Component**

|  |
| --- |
| Customer |
| Customer ID  Name  Password  Email  Address  Phone number |
| track\_product() |

**Customer**

Tracking Process

**3. Elaborated Design Class**

|  |
| --- |
| Customer |
| Customer ID  Name  Password  Email  Address  Phone number |
| track\_product() |

|  |
| --- |
| <<interface>>  Tracking Process |
| track\_product() |

#### 9.2.3.1 Collaboration Details

System Admin

Assign Branch Manager

Update Manager

Add branch

Edit Branch

Delete Branch

Branch Manager

Branch

Add Product

Edit Product

Track Product

Verify

Product

Customer

Track Product

#### 9.2.3.2 Appropriate Interfaces

I have considered that the elaborated classes do not need to be refactored any more. Therefore, no interface is requisite for this design.

#### 9.2.3.3 Elaborate Attributes

Elaborate attributes and define data types and data structures required to implement them:

|  |
| --- |
| Branch Manager |
| First name: String=not null {All characters from A-Z}  Last name: String=not null {All characters from A-Z}  Username: String = not null {All characters including A-Z, 0-9, special characters}  Password: String = not null {All characters including A-Z, 0-9, special characters}  Date of birth: Date = not null {d/m/y format}  Present address: String = not null {All characters including A-Z, 0-9, special characters}  Permanent address: String = not null {All characters including A-Z, 0-9, special characters}  Email \_address: String = not null {format - abc@abc}  Phone Number: Integer= not null {within 20 letters} |

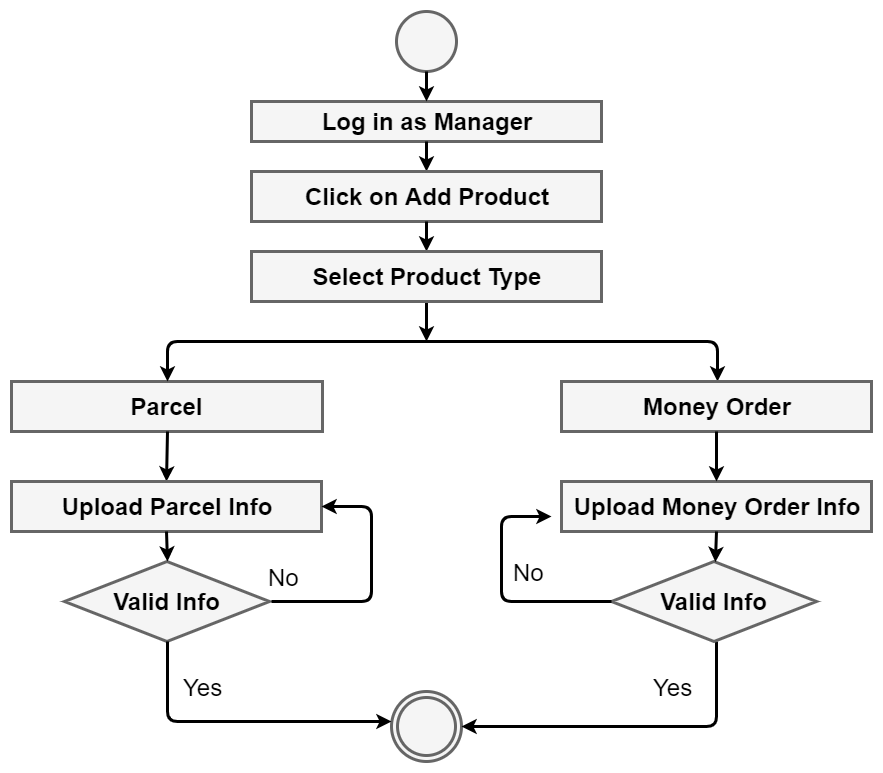
|  |
| --- |
| System Admin |
| User name: String = not null {All characters including A-Z, 0-9, special characters}  Password: String = not null {All characters including A-Z, 0-9, special characters} |

|  |
| --- |
| Product |
| Product type: String=not null {All characters from A-Z}  Source post office: String = not null {All characters including A-Z, 0-9, special characters}  Destination post office: String = not null {All characters including A-Z, 0-9, special characters}  Weight: Integer = not null{within 20 letters}  Cost: Integer = not null{within 20 letters}  Money order amount: {within 50 letters}  Receiver’s name: String=not null {All characters from A-Z}  Receiver’s phone number: Integer= not null {within 20 letters}  PIN number: Integer= not null {within 10 letters} |

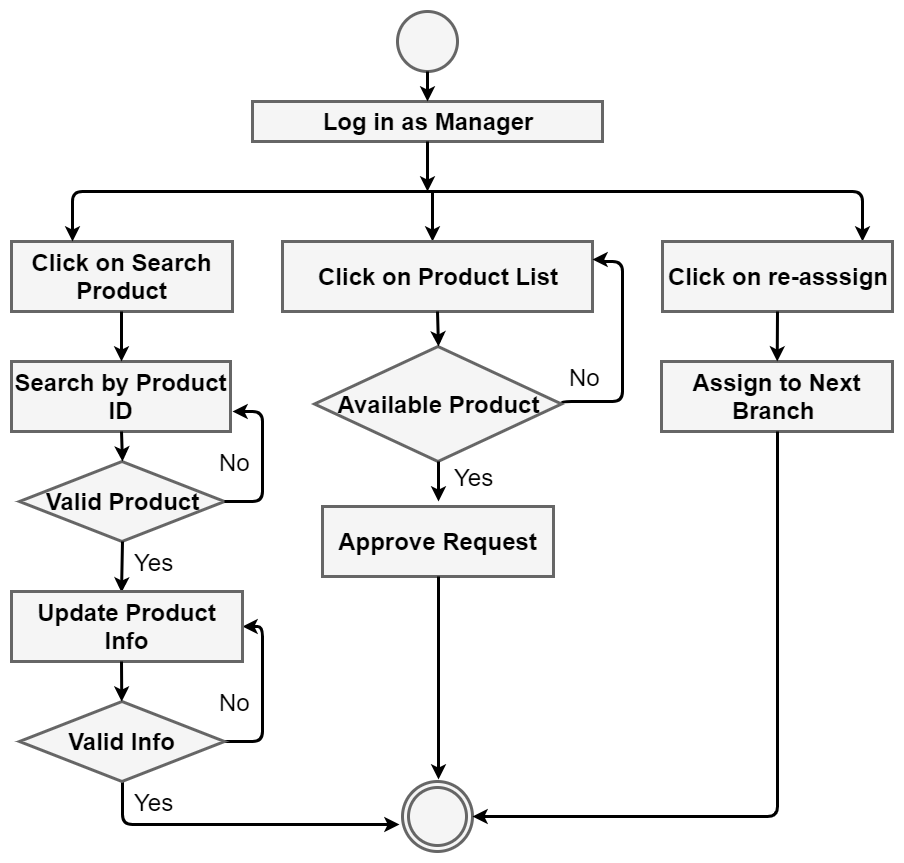
|  |
| --- |
| Customer |
| Name: String=not null {All characters from A-Z}  Password: String = not null {All characters including A-Z, 0-9, special characters}  Email address: String = not null {format - abc@abc}  Address: String = not null {All characters including A-Z, 0-9, special characters}  Phone number: Integer= not null {within 20 letters} |

#### 9.2.3.4 Describe Processing Flow

Describe processing flow within each operation in detail by means of pseudo code or UML activity diagrams.

****

##### Figure 9.6: Processing flow for “Add Product” method

****

##### Figure 9.7: Processing flow for “Product Management” method

### 9.2.4 Persistent Data

Describing persistent data sources and identifying the classes required to manage them

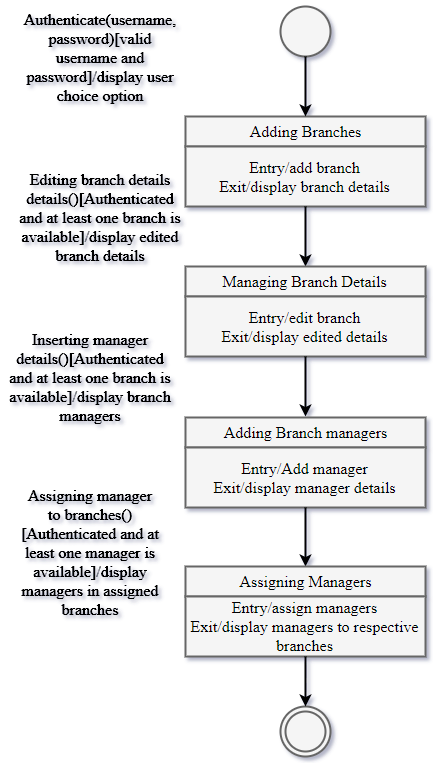
**Persistent data Source:**

Microsoft SQL Server 2014 Express is a powerful and reliable free data management system that delivers a rich and reliable data store for lightweight Web Sites and desktop applications.

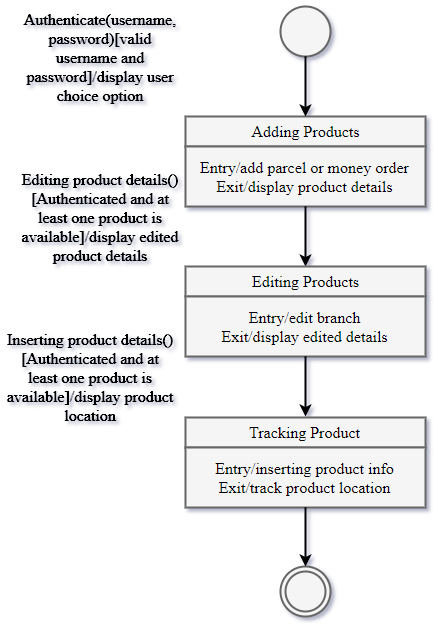
**Classes required to manage:**

Database

### 9.2.5 Develop and elaborate behavioral representations for a class or component



##### Figure 9.8: State Chart fragment for System Admin class



##### Figure 9.9: State Chart fragment for Branch manager class

### 9.2.6 Elaborating deployment diagrams

Processor

Elaborating deployment diagrams to provide additional implementation detail

Broadband or wireless connection

**Web Server**

**Web browser**

**Post Office Management System**

User PC

##### Figure 9.10: Elaborative deployment diagram of POMS

## 9.3 User Interface Design

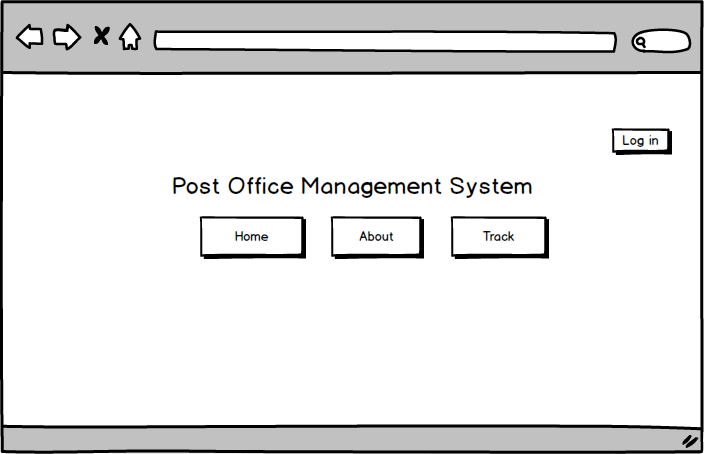
### **9.3.1 User Analysis:**

User Interface Design is basically for the users who will be using this  
system. We have identified some criteria by analyzing our intended  
users.

* Intended User: Post Office System Admin, Respective Branch Managers and Customers
* Gender: Male/ Female
* Level of Education: Any level for Customer (Capable of using internet), Post office Employers
* Age group: 18 or more

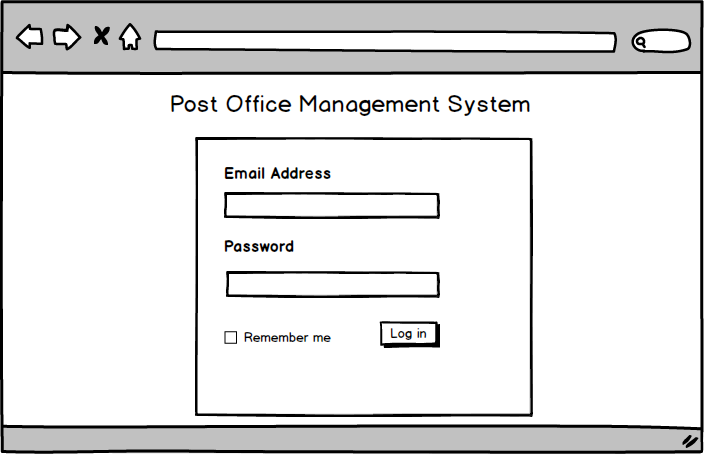
### **9.3.2 User Interface (MOC):**

When a user runs the application, the following window will appear.

****

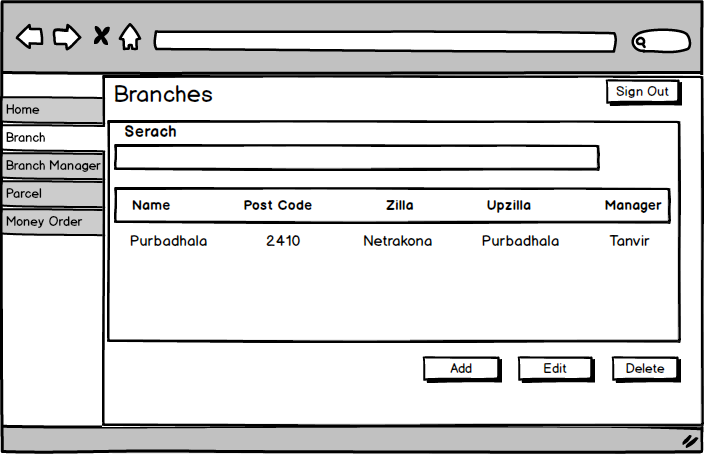
##### Figure 9.11: User interface for “Start” page

After Clicking log in button the following window will appear

****

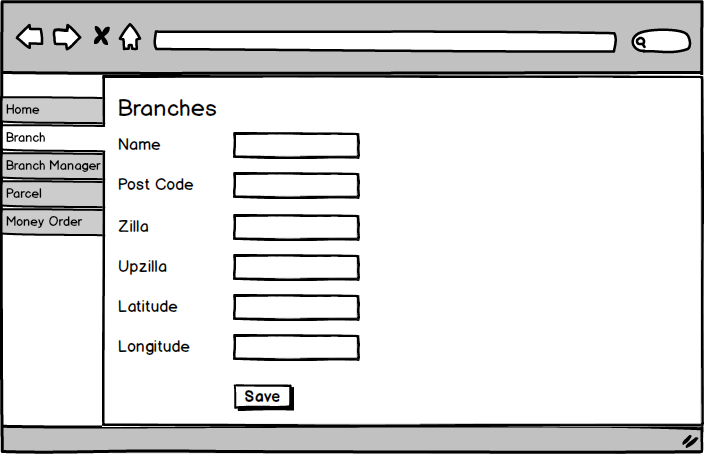
###### Figure 9.12: User interface for “Log in” page

After successful log in as a System Admin, he/she can see this page

****

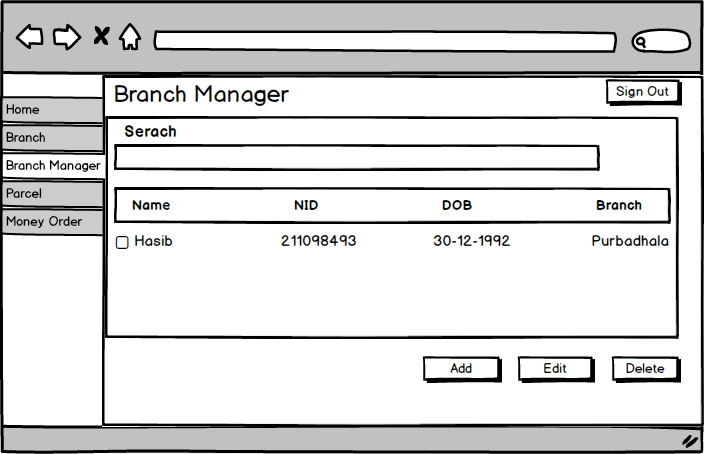
##### Figure 9.13: User interface for “View branch” page

System Admin can add, update, delete, search and see the list of branches from this page.

****

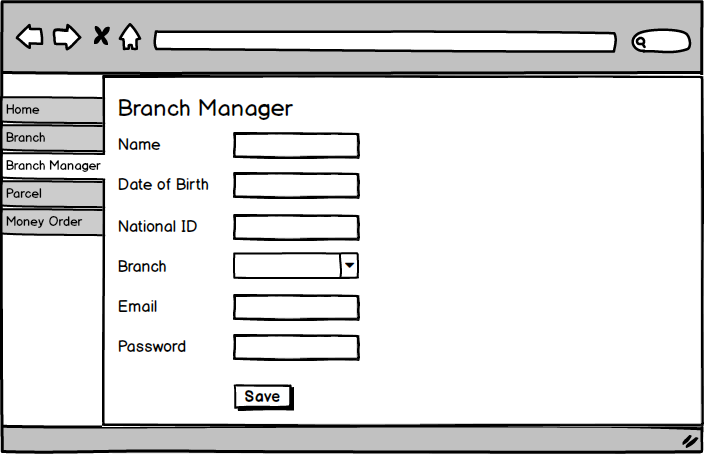
##### Figure 9.14: User interface for “Add branch” page

System Admin can also assign branch managers to respective branches.



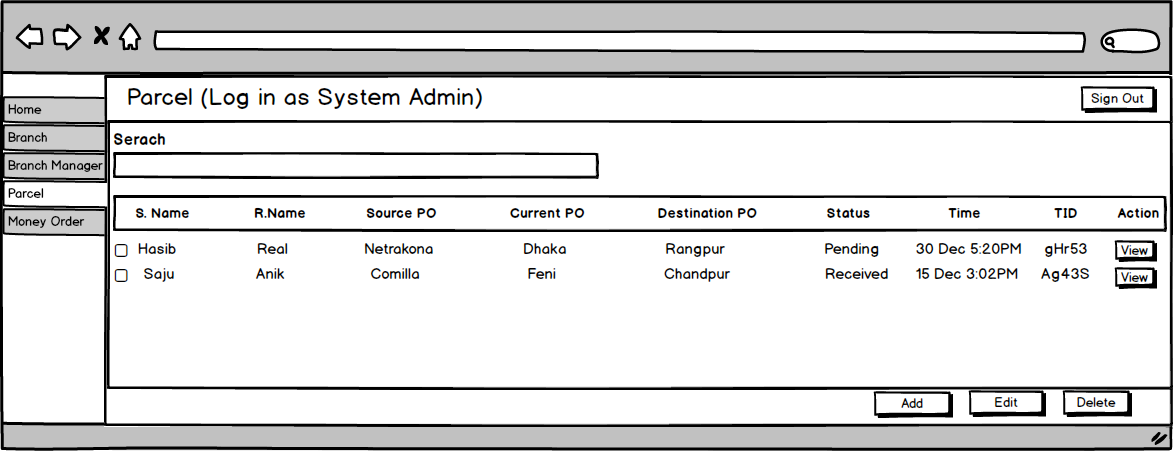
##### Figure 9.15: User interface for “Branch manager” page

System admin can also add, update, delete, search and view the list of branch managers.

****

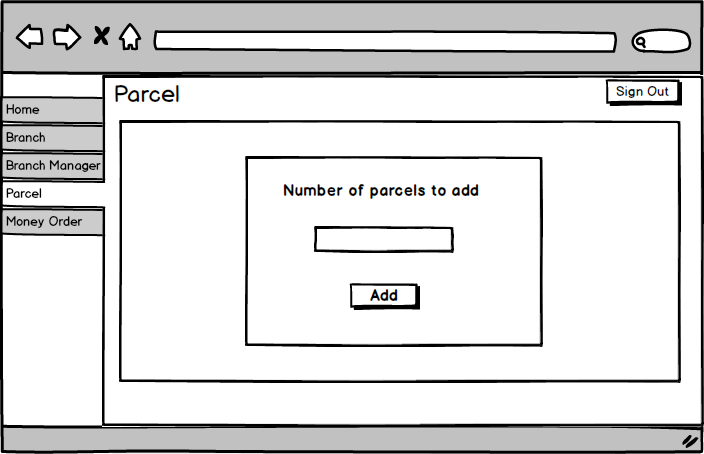
##### Figure 9.16: User interface for “Add branch manager” page

System Admin can manage product also.

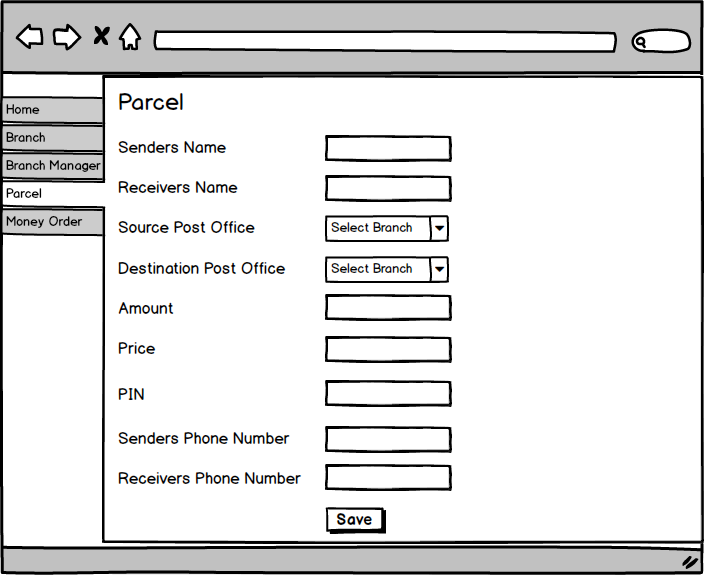
****

##### Figure 9.17: User interface for “Product (parcel/money order)” page

Add, update, search, delete, view are the operations of this page for product.

****

##### Figure 9.18: User interface for “Number of product” window

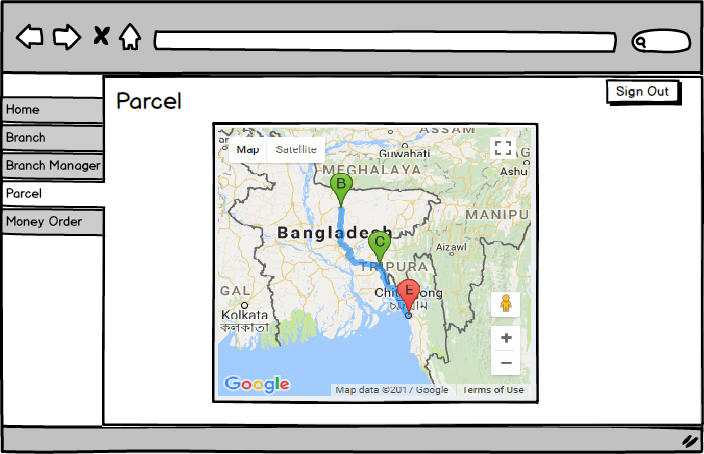
****

##### Figure 9.19: User interface for “Add Product” window

System admin and branch manager can view the track of product and see the location in google map

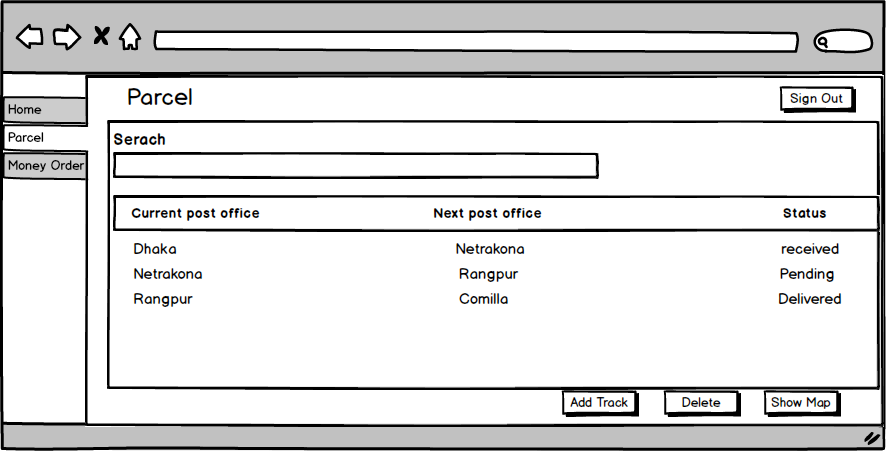
****

##### Figure 9.20: User interface for “Track of product” window

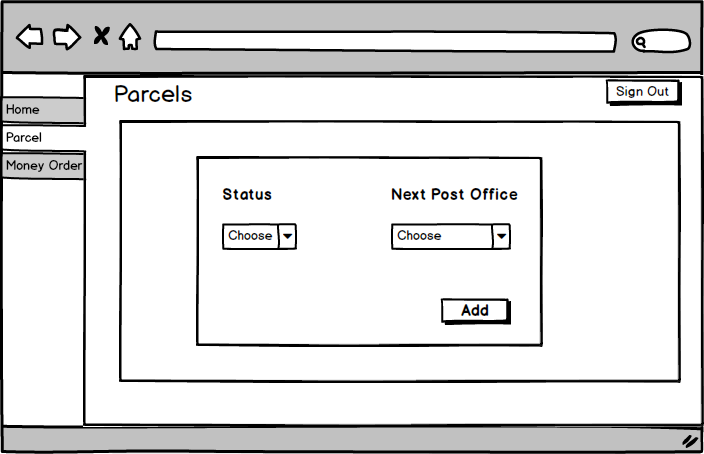
****

##### Figure 9.21: User interface for “Google Map” window

After successful log in as branch manager, he/she perform same tasks like add, view, edit, delete product but another extra task he/she can perform is tracking the product by assigning it the next branch.

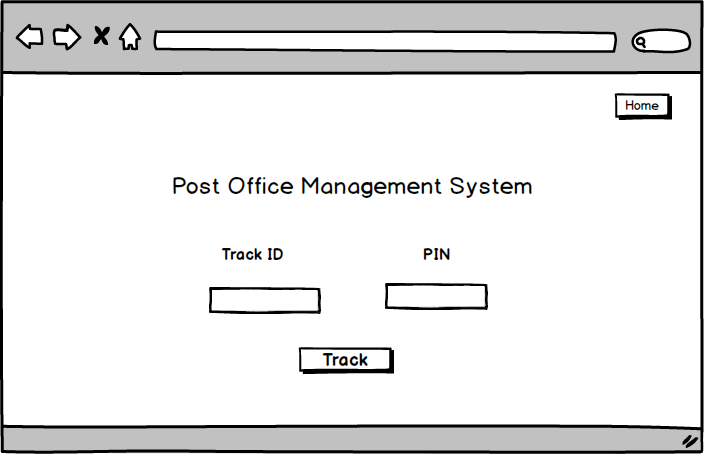


##### Figure 9.22: User interface for “Product” page

****

##### Figure 9.23: User interface for “Add track” window

The act by the customer is to track the product with the given secret PIN number and the Track ID.



##### Figure 9.24: User interface for “Track product” page

# **Chapter 10**

# **Testing**

|  |  |  |
| --- | --- | --- |
| **Version** | **Date** | **Document name** |
| 1.0 | 20-12-2017 | Test plan outline |
|  |  |  |

## 

## 10.1 Introduction

This test plan describes the testing approach and overall framework that will drive the testing of the Post Office Management System project. The document introduces the test strategy, test execution strategy and test management process. The test plan document is created during the planning phase of the project. Intended audience is the testing team members, project supervisor, developer and stakeholders.

## 10.2 Test plan identifier

This is the most important plan of my project. It should identify the project as well as the documents related to test phase level. Changing requirements at many phases is a common factor and for this purpose my test plan may have several versions. However, Testing will be performed on the basis of work progress and the deadline.

#### 10.2.1 Summary of items and features to be tested

Web application version is to be tested for School of Programming. In this application, it has some features that to be tested, these are: authentication, branch management, product management and tracking system.

#### 10.2.2 Requirement and history of items

The requirement that will be tested are collected from the software requirement and specification (SRS) and design primarily.

#### 10.2.3 High-level description of testing goals

We use testing to detect bug, fix bud and to make the system more secure than before. When a software product is released, there can be a lot of bugs which are not detected and solved during the development phase. These bugs are found in testing phase and solved after testing phase. So, testing is required for developing a better software product.

#### 10.2.4 Reference Document

* Software Requirement Specification
* Architectural design
* Component design
* User Interface design

## 10.3 Scope of Testing

In Post Office Management System project, I will only cover black box testing only in testing phase. I have planned to perform functional testing along with feature test to ensure correct implementation of software requirement specifications. In this project I am not covering white box testing, so no tools will be used for testing.

## 10.4 Test items

* Web application of Post Office Management System project

## 10.5 Features to be tested

* Log in
* View branch
* Adding branch
* Edit Branch
* View branch manager
* Assigning managers to branch
* Update branch managers
* View parcel
* Add parcel
* Edit parcel
* Add money order
* Edit money order
* Tracking Product

## 10.6 Features not to be tested

* All the features will be tested in this project.

## 10.7 Approach

* Black box testing will be covered in this system
* Functionality of the system will be tested
* Features will be tested in Google Chrome and Mozilla Firefox browser.
* Integration testing will be performed

## 10.8 Item pass/fail criteria

Specifying the criteria that I will use to determine whether each test item of my project has passed or failed during testing. The planning criteria gives the framework for how the system will be evaluated and under what circumstances it will be released.

## 10.9 Suspension criteria and Resumption requirements

Suspension criteria will be used when it is needed to suspend all or a portion of the testing activities when the testing has no value and the build is not working properly which is overall a wasting of resources. On the other hand, resumption criteria specify when testing can be resumed after it has been suspended. These will be applied in such situations:

**Suspension when**:

* A defect is introduced that doesn’t allow any further testing.
* Unavailability of external dependent systems during execution.

**Resumption when:**

* When the defect is fixed.
* When the external dependent systems become available again.

## 10.10 Test Deliverables

Following deliverables will be provided after testing at the end of the project:

* Test plan document
* Test cases

## 10.11 Environmental needs

* As we need data for testing, the test data which will be provided is an environmental need. Without the provided data, we cannot perform any phase of testing.
* I will conduct an integration test after testing the individual features.

## 10.12 Staffing and training needs

I will take some short training on how to perform functional and integration testing as I am not very proficient in functional and integration testing.

## 10.13 Responsibilities

The Responsibilities for testing different modules is given below:

|  |  |  |  |
| --- | --- | --- | --- |
| **Serial number** | **Module** | **Responsible** | **Roll** |
| 1. | Authentication | Tanvir Ahmed Siddiki | BSSE-0639 |
| 2. | Branch management | Tanvir Ahmed Siddiki | BSSE-0639 |
| 3. | Product management | Tanvir Ahmed Siddiki | BSSE-0639 |
| 4. | Tracking system | Tanvir Ahmed Siddiki | BSSE-0639 |

## 10.14 Risks and contingencies

* If development takes much time to finish than estimated, then testing will be late. Consequently, there is a risk of not meeting the deadline of the software.
* In every software project, there is a chance of changing or modification of the requirements. If this happens, I may need to redesign or modify my plan and test cases.

If the requirement will change further, the following steps will be taken:

* Since I have to complete my product in due time, I will reschedule my working period and increase my working time to complete testing.
* The number of test cases may be reduced.
* Number of acceptable defects may be increased. These defects will be fixed in further releases.

## 10.15 Testing cost:

Since this is an academic project, so the cost of testing is ignored

## 10.16 Approval

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Role** | **Approver/Reviewer** | **Approval date** |
| Emon Kumar Dey | Project Supervisor |  |  |

## 10.17 Test Cases

**Title:** Log in

**Short Description:** Test Log in system

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test cases** | **Action** | **Pre-condition** | **Expected result** | **Pass/Fail** |
| 1. | Verify that all labels are spelled and aligned correctly. | Application runs correctly | All labels should be spelled and aligned correctly. | Pass |
| 2. | Verify with empty e-mail address, empty password and click submit button | Users are not signed in | The system should display “Fill out the fields” | Pass |
| 3. | Verify with valid e-mail address and empty password  and click submit | Users are not signed in | The system should display “Fill out the fields” | Pass |
| 4. | Verify with valid e-mail and wrong password  and click submit | Users are not signed in | The system should display “Invalid user name & password” | Pass |
| 5. | Verify with empty e-mail and valid password  and click submit | Users are not signed in | The system should display “Fill out the fields” | Pass |
| 6. | Verify with wrong e-mail and valid password  and click submit | Users are not signed in | The system should display “Invalid user name & password” | Pass |
| 7. | Verify with wrong e-mail and wrong password  and click submit | Users are not signed in | The system will display “Invalid user name & password” | Pass |
| 8. | Verify with correct e-mail and correct password and click submit | Users are not signed in | Redirect authenticated users to respective home page | Pass |

**Tile:** View Branch

**Short Description:** Test all branch details in branch page

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test cases** | **Action** | **Pre-condition** | **Expected result** | **Pass/Fail** |
| 1. | Verify that all labels are spelled and aligned correctly. | Signed in as System Admin | All labels should be spelled and aligned correctly. | Pass |
| 2. | Verify that clicking on "Add Branch" button can take system admin to add branch page | Signed in as System Admin | The system should take System Admin to add branch page | Pass |
| 3. | Verify that clicking on "Edit Branch” button can take System Admin to edit branch page | Signed in as System Admin | The system should take System Admin to edit branch page | Pass |
| 4. | Verify that Check boxes are working correctly when selecting branches | Signed in as System Admin | The system must select branches with checkbox | Pass |
| 5. | Verify that "Delete" button can delete corresponding branch from the list | Signed in as System Admin | The system must delete selected branch by clicking delete branch | Pass |
| 6. | Verify that clicking “Next” button can take the user to the next page of branch list | Signed in as System Admin | The system should take system admin to the next page of branches by clicking next button | Pass |
| 7. | Verify that clicking “Previous” button can take the user to the previous page of branch list | Signed in as System Admin | The system should take system admin to the previous page of branches by clicking previous button | Pass |

**Tile:** Add Branch

**Short Description:** Test Add branch details page

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test cases** | **Action** | **Pre-condition** | **Expected result** | **Pass/Fail** |
| 1. | Verify that all labels are spelled and aligned correctly. | Signed in as system admin | All labels should be spelled and aligned correctly. | Pass |
| 2. | Verify with empty name, post code, zilla, upzilla, latitude, longitude and click save button | Signed in as system admin | The system should display “Fill out the fields” | Pass |
| 3. | Verify with providing inputs on all the fields except anyone of the fields including name, post code, zilla, upzilla, latitude, longitude and click save button | Signed in as system admin | The system should display “Fill out the fields” | Pass |
| 4. | Verify that providing character input in Post code and all the other inputs are valid and click save button | Signed in as system admin | The system should display “Invalid input” | Pass |
| 5. | Verify that providing character input in latitude and longitude, and all the other inputs are valid and click save button | Signed in as system admin | The system should display “Invalid input” | Pass |
| 6. | Verify with valid name, post code, zilla, upzilla, latitude, longitude and click save button | Signed in as System Admin | The system should save the branch | Pass |

**Tile:** Edit Branch

**Short Description:** Test edit branch details page

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test cases** | **Action** | **Pre-condition** | **Expected result** | **Pass/Fail** |
| 1. | Verify that all labels are spelled and aligned correctly. | Signed in as system admin | All labels should be spelled and aligned correctly. | Pass |
| 2. | Verify that all the previous inputs are shown in particular categories | Signed in as system admin | The system should show previous inputs in particular categories | Pass |
| 2. | Verify with empty name, post code, zilla, upzilla, latitude, longitude and click save button | Signed in as system admin | The system should display “Fill out the fields” | Pass |
| 3. | Verify with providing inputs on all the fields except anyone of the fields including name, post code, zilla, upzilla, latitude, longitude and click save button | Signed in as system admin | The system should display “Fill out the fields” | Pass |
| 4. | Verify that providing character input in Post code and all the other inputs are valid and click save button | Signed in as system admin | The system should display “Invalid input” | Pass |
| 5. | Verify that providing character input in latitude and longitude, and all the other inputs are valid and click save button | Signed in as system admin | The system should display “Invalid input” | Pass |
| 6. | Verify with valid name, post code, zilla, upzilla, latitude, longitude and click save button | Signed in as System Admin | The system should successfully edit the branch | Pass |
| 7. | Verify that without clicking save button no changes are adapted | Signed in as System Admin | The system should keep the previous data without clicking save button | Pass |

**Tile:** View Branch Managers

**Short Description:** Test all branch manager’s details in branch manager page

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test cases** | **Action** | **Pre-condition** | **Expected result** | **Pass/Fail** |
| 1. | Verify that all labels are spelled and aligned correctly. | Signed in as System Admin | All labels should be spelled and aligned correctly. | Pass |
| 2. | Verify that clicking on "Add Branch Manager" button can take system admin to add branch page | Signed in as System Admin | The system should take System Admin to add branch manager page | Pass |
| 3. | Verify that clicking on "Edit Branch manager” button can take System Admin to edit branch page | Signed in as System Admin | The system should take System Admin to edit branch manager page | Pass |
| 4. | Verify that Check boxes are working correctly when selecting branch managers | Signed in as System Admin | The system must select branch managers with checkbox | Pass |
| 5. | Verify that "Delete" button can delete corresponding branch managers from the list | Signed in as System Admin | The system must delete selected branch managers by clicking delete branch managers | Pass |
| 6. | Verify that clicking “Next” button can take the system admin to the next page of branch manager list | Signed in as System Admin | The system should take system admin to the next page of branch managers by clicking next button | Pass |
| 7. | Verify that clicking “Previous” button can take the system admin to the previous page of branch manager list | Signed in as System Admin | The system should take system admin to the previous page of branch managers by clicking previous button | Pass |

**Tile:** Add Branch Manager

**Short Description:** Test Add branch manager details page

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test cases** | **Action** | **Pre-condition** | **Expected result** | **Pass/Fail** |
| 1. | Verify that all labels are spelled and aligned correctly. | Signed in as system admin | All labels should be spelled and aligned correctly. | Pass |
| 2. | Verify with empty name, date of birth, national id, branch, email, password and click save button | Signed in as system admin | The system should display “Fill out the fields” | Pass |
| 3. | Verify with providing valid inputs on all the fields except anyone of the fields including name, post code, zilla, upzilla, latitude, longitude and click save button | Signed in as system admin | The system should display “Fill out the fields” | Pass |
| 4. | Verify that providing input for date of birth except “yy-mm-dd” format and click save button | Signed in as system admin | The system should display “Invalid input” | Pass |
| 5. | Verify that providing input for national id except integer number and click save button | Signed in as system admin | The system should display “Invalid input” | Pass |
| 6. | Verify that input for branch are from created branches | Signed in as System Admin | The system should take input only from provided branches | Pass |
| 7. | Verify that providing input for email except abc@abc.abc and click save button | Signed in as System Admin | The system should display “Invalid input” | Pass |
| 8. | Verify that providing input for password is less than six characters and click save button | Signed in as System Admin | The system should display “Invalid input” | Pass |

**Tile:** Edit Branch Manager

**Short Description:** Test edit branch manager details page

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test cases** | **Action** | **Pre-condition** | **Expected result** | **Pass/Fail** |
| 1. | Verify that all labels are spelled and aligned correctly. | Signed in as system admin | All labels should be spelled and aligned correctly. | Pass |
| 2. | Verify that all the previous inputs are shown in particular categories | Signed in as system admin | The system should show previous inputs in particular categories | Pass |
| 3. | Verify with empty name, date of birth, national id, branch, email, password and click save button | Signed in as system admin | The system should display “Fill out the fields” | Pass |
| 4. | Verify with providing valid inputs on all the fields except anyone of the fields including name, post code, zilla, upzilla, latitude, longitude and click save button | Signed in as system admin | The system should display “Fill out the fields” | Pass |
| 5. | Verify that providing input for date of birth except “yy-mm-dd” format and click save button | Signed in as system admin | The system should display “Invalid input” | Pass |
| 6. | Verify that providing input for national id except integer number and click save button | Signed in as system admin | The system should display “Invalid input” | Pass |
| 7. | Verify that input for branch are from created branches | Signed in as System Admin | The system should take input only from provided branches | Pass |
| 8. | Verify that providing input for email except abc@abc.abc and click save button | Signed in as System Admin | The system should display “Invalid input” | Pass |
| 9. | Verify that providing input for password is less than six characters and click save button | Signed in as System Admin | The system should display “Invalid input” | Pass |
| 10. | Verify that without clicking “reset password” the previous password remain unchanged | Signed in as System Admin | Without clicking “reset password” the previous password must remain unchanged | Pass |
| 11. | Verify that without clicking save button no changes are adapted | Signed in as System Admin | The system should keep the previous data | Pass |

**Tile:** View Parcel

**Short Description:** Test all parcel details in parcel page

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test cases** | **Action** | **Pre-condition** | **Expected result** | **Pass/Fail** |
| 1. | Verify that all labels are spelled and aligned correctly. | Signed in as System Admin or Branch Manager | All labels should be spelled and aligned correctly. | Pass |
| 2. | Verify that clicking on "Add Parcel" button can take the user to add parcel page | Signed in as System Admin or Branch Manager | The system should take System Admin to add parcel page | Pass |
| 3. | Verify that clicking on "Edit parcel” button can take user to edit parcel page | Signed in as System Admin or Branch Manager | The system should take System Admin to edit parcel page | Pass |
| 4. | Verify that Check boxes are working correctly when selecting parcels | Signed in as System Admin or Branch Manager | The system must select parcel correctly with checkbox | Pass |
| 5. | Verify that "Delete" button can delete corresponding parcels from the list | Signed in as System Admin or Branch Manager | The system must delete selected parcels by clicking delete branch managers | Pass |
| 6. | Verify that clicking “Next” button can take the user to the next page of parcel list | Signed in as System Admin or Branch Manager | The system should take system admin to the next page of parcels by clicking next button | Pass |
| 7. | Verify that clicking “Previous” button can take the user to the previous page of parcel list | Signed in as System Admin or Branch Manager | The system should take system admin to the previous page of parcels by clicking previous button | Pass |
| 8. | Verify that clicking on “Arrow” system admin can see the track of parcel and the “Show Map” option | Signed in as System Admin | clicking on “Arrow” system admin must see the track of parcel and the “Show Map” option | Pass |
| 9. | Verify that clicking on “Arrow” branch manager can see the track of parcel and the “Show Map”, “Add Track”, “Delete” option | Signed in as Branch Manager | clicking on “Arrow” branch manager should see the track of parcel and the “Show Map”, “Add Track”, “Delete” option | Pass |
| 10. | Verify that clicking on “Add Track” branch manager can set the status of the parcel and select next post office form created branch list | Signed in as Branch Manager | Clicking on “Add Track” branch manager should set the status of the parcel and select next post office form created branch list | Pass |
| 11. | Verify that clicking on “Show in Map” button can correctly shows the location of product in google map with corresponding tracks | Signed in as System Admin or Branch Manager | Clicking on “Show in Map” button must correctly show the location of product in google map with corresponding tracks | Pass |

**Tile:** Add Parcel

**Short Description:** Test add parcel page

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test cases** | **Action** | **Pre-condition** | **Expected result** | **Pass/Fail** |
| 1. | Verify that all labels are spelled and aligned correctly. | Signed in as system admin or Branch Manager | All labels should be spelled and aligned correctly. | Pass |
| 2. | Verify with empty sender’s name, receivers name, receivers address, source post office, destination post office, weight, price, PIN, senders and receivers mobile number and click save button | Signed in as system admin or Branch Manager | The system should display “Fill out the fields” | Pass |
| 3. | Verify with providing inputs on all the fields except anyone of the fields including sender’s name, receivers name, receivers address, source post office, destination post office, weight, price, PIN, senders and receivers mobile number and click save button | Signed in as system admin or Branch Manager | The system should display “Fill out the fields” | Pass |
| 4. | Verify that input for source and destination post office can be selected from created branches of post office | Signed in as system admin or Branch Manager | Input for source and destination post office must be from created branches of post office | Pass |
| 5. | Verify that providing input for senders and receivers phone number except eleven digit integer and click save button | Signed in as System Admin or Branch Manager | The system should display “Invalid input” | Pass |

**Tile:** Edit Parcel

**Short Description:** Test edit parcel page

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test cases** | **Action** | **Pre-condition** | **Expected result** | **Pass/Fail** |
| 1. | Verify that all labels are spelled and aligned correctly. | Signed in as system admin or Branch Manager | All labels should be spelled and aligned correctly. | Pass |
| 2. | Verify that all the previous inputs are shown in particular categories | Signed in as system admin or Branch Manager | The system should show previous inputs in particular categories | Pass |
| 3. | Verify with empty sender’s name, receivers name, receivers address, source post office, destination post office, weight, price, PIN, senders and receivers mobile number and click save button | Signed in as system admin or Branch Manager | The system should display “Fill out the fields” | Pass |
| 4. | Verify with providing inputs on all the fields except anyone of the fields including sender’s name, receivers name, receivers address, source post office, destination post office, weight, price, PIN, senders and receivers mobile number and click save button | Signed in as system admin or Branch Manager | The system should display “Fill out the fields” | Pass |
| 5. | Verify that input for source and destination post office can be selected from created branches of post office | Signed in as system admin or Branch Manager | Input for source and destination post office must be from created branches of post office | Pass |
| 6. | Verify that providing input for senders and receivers phone number except eleven digit integer and click save button | Signed in as System Admin or Branch Manager | The system should display “Invalid input” | Pass |
| 7. | Verify that without clicking save button no changes are adapted | Signed in as System Admin or Branch Manager | The system should keep the previous data without clicking save button | Pass |

**Tile:** View Money Order

**Short Description:** Test all money order details in money order page

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test cases** | **Action** | **Pre-condition** | **Expected result** | **Pass/Fail** |
| 1. | Verify that all labels are spelled and aligned correctly. | Signed in as System Admin or Branch Manager | All labels should be spelled and aligned correctly. | Pass |
| 2. | Verify that clicking on "Add money order " button can take user to add money order page | Signed in as System Admin or Branch Manager | The system should take System Admin to add money order page | Pass |
| 3. | Verify that clicking on "Edit money order” button can take user to edit money order page | Signed in as System Admin or Branch Manager | The system should take System Admin to edit money order page | Pass |
| 4. | Verify that Check boxes are working correctly when selecting money order | Signed in as System Admin or Branch Manager | The system must select money order correctly with checkbox | Pass |
| 5. | Verify that "Delete" button can delete corresponding money order from the list | Signed in as System Admin or Branch Manager | The system must delete selected money order by clicking delete money order | Pass |
| 6. | Verify that clicking “Next” button can take the user to the next page of money order list | Signed in as System Admin or Branch Manager | The system should take system admin to the next page of money order by clicking next button | Pass |
| 7. | Verify that clicking “Previous” button can take the user to the previous page of money order list | Signed in as System Admin or Branch Manager | The system should take system admin to the previous page of money order by clicking previous button | Pass |
| 8. | Verify that clicking on “Arrow” system admin can see the track of money order and the “Show Map” option | Signed in as System Admin | clicking on “Arrow” system admin must see the track of money order and the “Show Map” option | Pass |
| 9. | Verify that clicking on “Arrow” branch manager can see the track of money order and the “Show Map”, “Add Track”, “Delete” option | Signed in as Branch Manager | clicking on “Arrow” branch manager should see the track of parcel and the “Show Map”, “Add Track”, “Delete” option | Pass |
| 10. | Verify that clicking on “Add Track” branch manager can set the status of the money order and select next post office form created branch list | Signed in as Branch Manager | Clicking on “Add Track” branch manager should set the status of the parcel and select next post office form created branch list | Pass |
| 11. | Verify that clicking on “Show in Map” button can correctly shows the location of money order in google map with corresponding tracks | Signed in as System Admin or Branch Manager | Clicking on “Show in Map” button must correctly show the location of money order in google map with corresponding tracks | Pass |

**Tile:** Add Money order

**Short Description:** Test money order page

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test cases** | **Action** | **Pre-condition** | **Expected result** | **Pass/Fail** |
| 1. | Verify that all labels are spelled and aligned correctly. | Signed in as system admin or Branch Manager | All labels should be spelled and aligned correctly. | Pass |
| 2. | Verify with empty sender’s name, receivers name, receivers address, source post office, destination post office, amount, cost, PIN, senders and receivers mobile number and click save button | Signed in as system admin or Branch Manager | The system should display “Fill out the fields” | Pass |
| 3. | Verify with providing inputs on all the fields except anyone of the fields including sender’s name, receivers name, receivers address, source post office, destination post office, amount, cost, PIN, senders and receivers mobile number and click save button | Signed in as system admin or Branch Manager | The system should display “Fill out the fields” | Pass |
| 4. | Verify that input for source and destination post office can be selected from created branches of post office | Signed in as system admin or Branch Manager | Input for source and destination post office must be from created branches of post office | Pass |
| 5. | Verify that providing input for senders and receivers phone number except eleven digit integer and click save button | Signed in as System Admin or Branch Manager | The system should display “Invalid input” | Pass |

**Tile:** Edit Money Order

**Short Description:** Test edit money order page

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test cases** | **Action** | **Pre-condition** | **Expected result** | **Pass/Fail** |
| 1. | Verify that all labels are spelled and aligned correctly. | Signed in as system admin or Branch Manager | All labels should be spelled and aligned correctly. | Pass |
| 2. | Verify that all the previous inputs are shown in particular categories | Signed in as system admin or Branch Manager | The system should show previous inputs in particular categories | Pass |
| 3. | Verify with empty sender’s name, receivers name, receivers address, source post office, destination post office, amount, cost, PIN, senders and receivers mobile number and click save button | Signed in as system admin or Branch Manager | The system should display “Fill out the fields” | Pass |
| 4. | Verify with providing inputs on all the fields except anyone of the fields including sender’s name, receivers name, receivers address, source post office, destination post office, amount, cost, PIN, senders and receivers mobile number and click save button | Signed in as system admin or Branch Manager | The system should display “Fill out the fields” | Pass |
| 5. | Verify that input for source and destination post office can be selected from created branches of post office | Signed in as system admin or Branch Manager | Input for source and destination post office must be from created branches of post office | Pass |
| 6. | Verify that providing input for senders and receivers phone number except eleven digit integer and click save button | Signed in as System Admin or Branch Manager | The system should display “Invalid input” | Pass |
| 7. | Verify that without clicking save button no changes are adapted | Signed in as System Admin or Branch Manager | The system should keep the previous data without clicking save button | Pass |

**Tile:** Tracking Product

**Short Description:** Test tracking system for product

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test cases** | **Action** | **Pre-condition** | **Expected result** | **Pass/Fail** |
| 1. | Verify that all labels are spelled and aligned correctly. | Need to have track id and pin number | All labels should be spelled and aligned correctly. | Pass |
| 2. | Verify with empty track id and pin number and click track button | Need to have track id and pin number | The system should display “Fill out the fields” | Pass |
| 3. | Verify with empty track id and valid pin number and click track button | Need to have track id and pin number | The system should display “Fill out the fields” | Pass |
| 4. | Verify with valid track id and empty pin number and click track button | Need to have track id and pin number | The system should display “Fill out the fields” | Pass |
| 5. | Verify with valid track id and valid pin number and click track button | Need to have track id and pin number | The system should show the user google map location of the product with respective track points | Pass |

# **Chapter 11**

# **User Manual**

## 12.1 Introduction

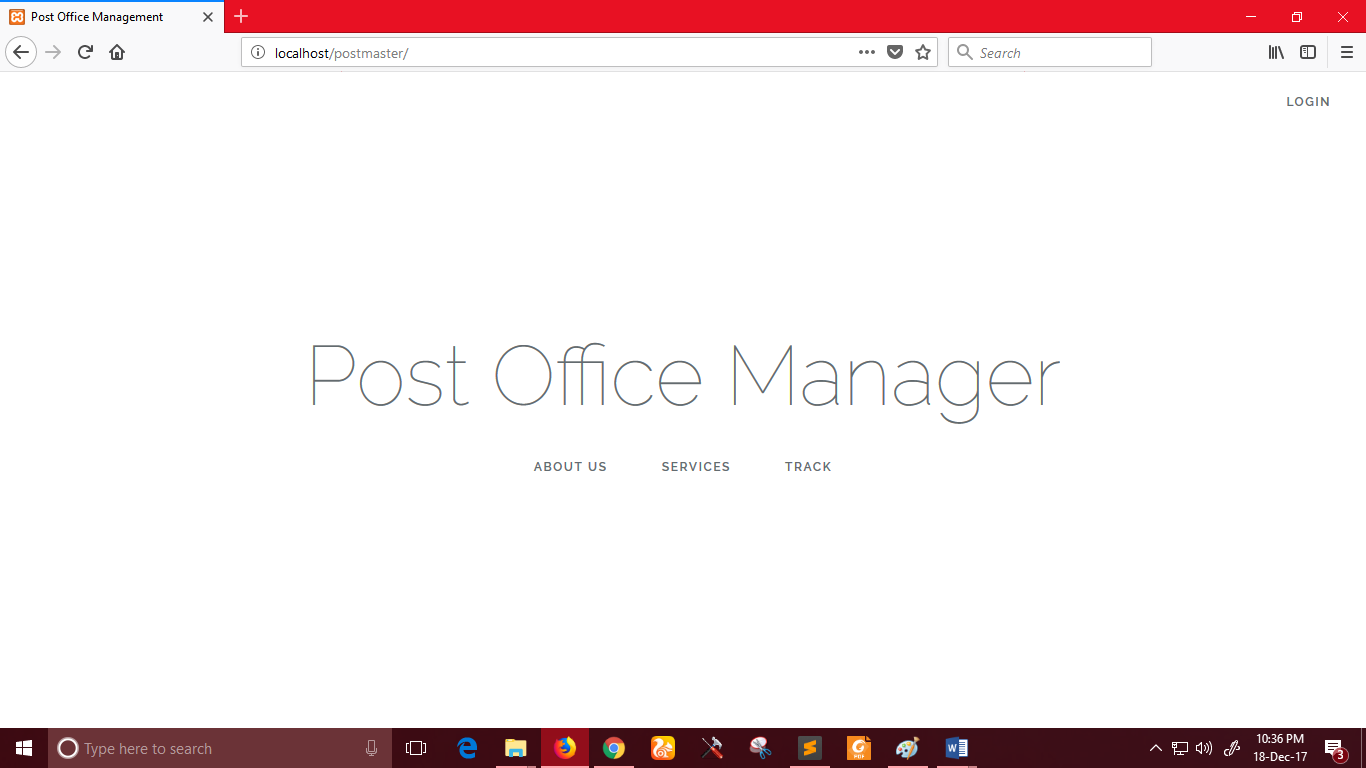
A user guide also commonly known as a user manual, is a technical communication document intended to give assistance to people using a particular system. It is usually written by a technical writer. Although user guides are written by programmers, product or project managers, or other technical staff particularly in smaller companies. User guides are most commonly associated with electronic goods, computer hardware and software. Most user guides contain both a written guide and the associated images. In the case of computer applications, it is customary to include screenshots of the human-machine interface.

## 12.2 User’s Guide

It is a document of technical communication that is used to give a detailed knowledge to the people who is intended to use the system. Here in user manual I have given the screenshots of the project and particular description of it.

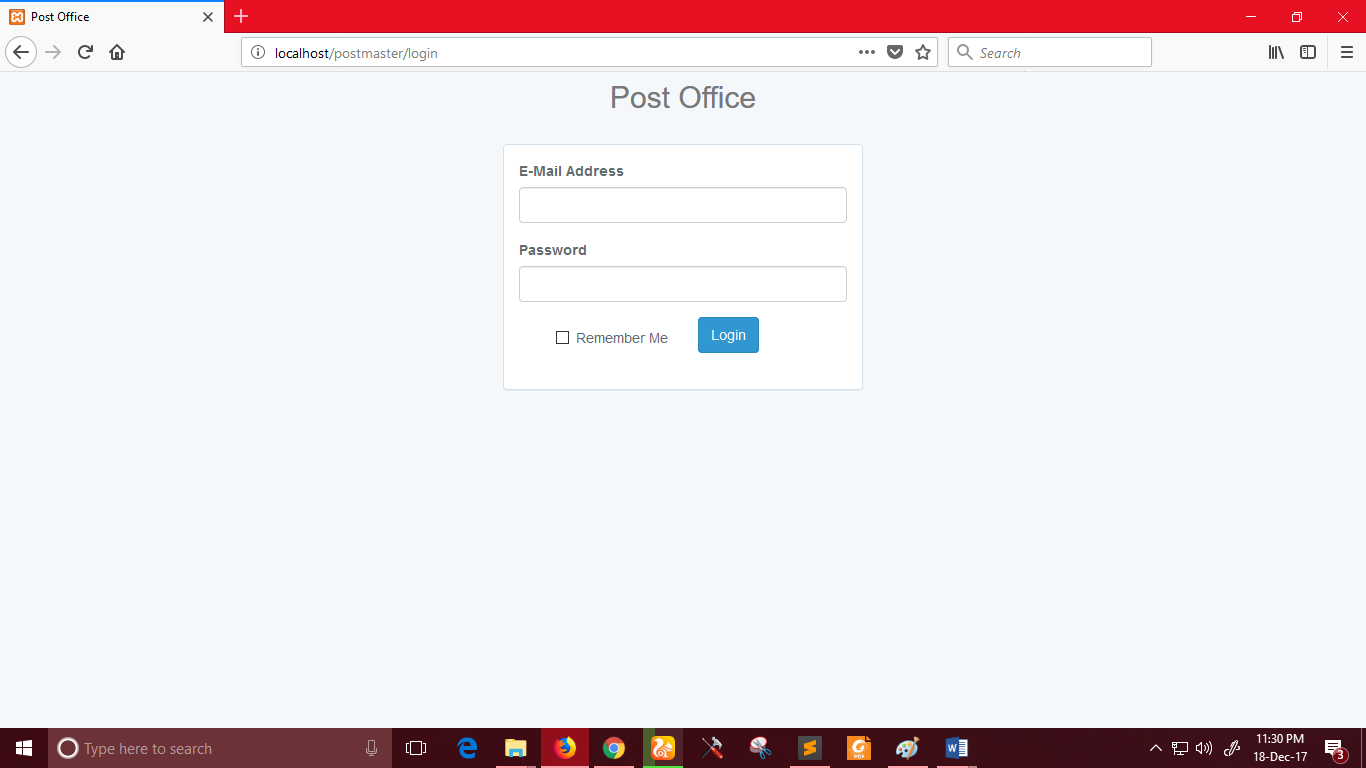
**Start Page:**

When e user runs the application the following window will appear. Here, the log in option, details about post office and the tracking options are available



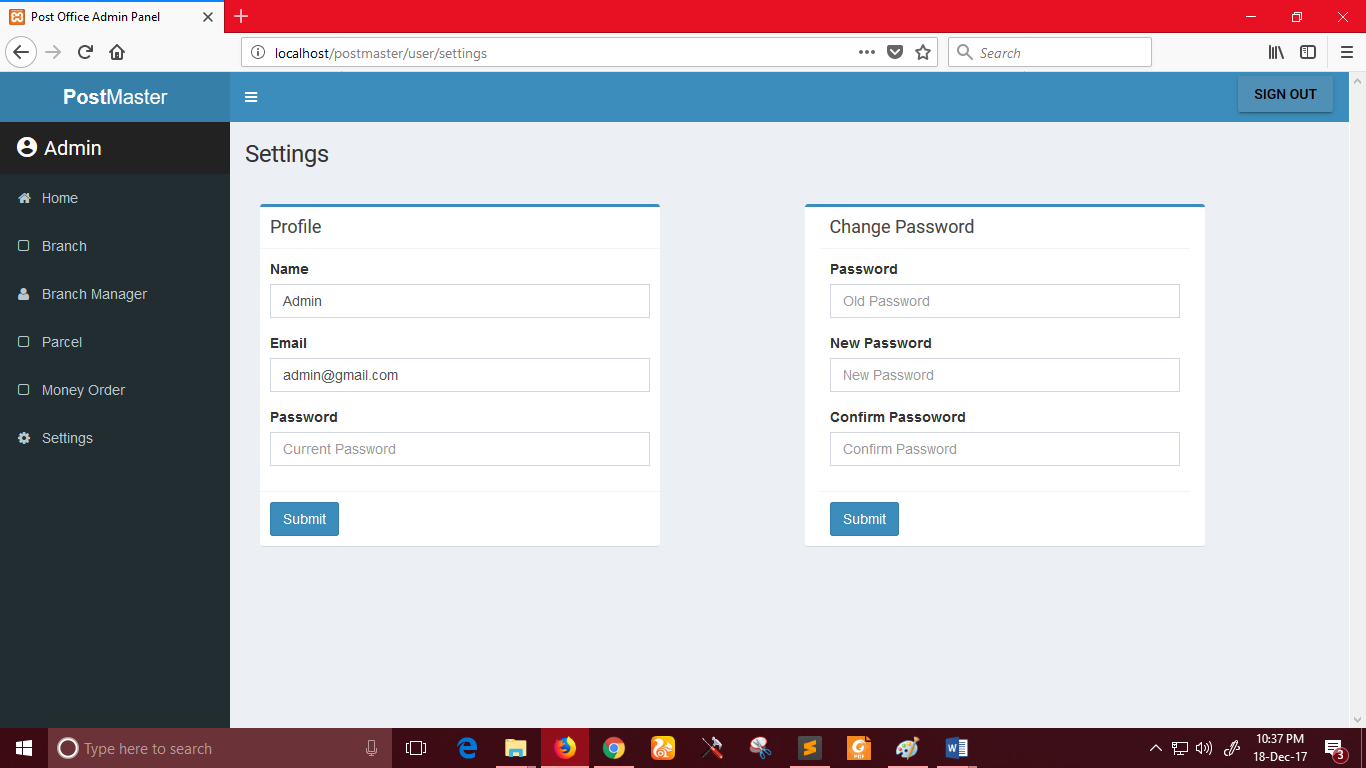
**Log in:**

After clicking log in button, the following window will appear and the system admin or branch manager can be logged in to the system with valid email and password.



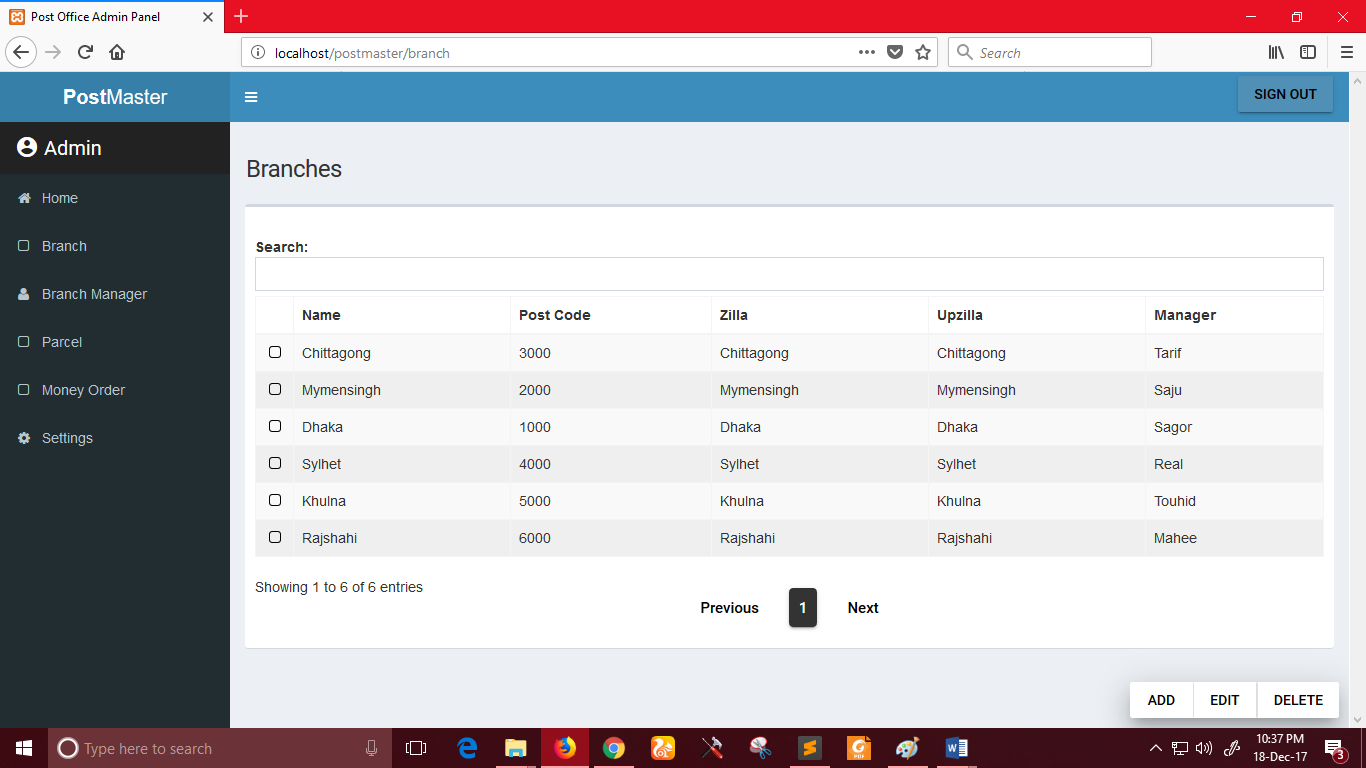
**System admin profile:**

After successful log in as system admin, he/she can view and update his/her profile.

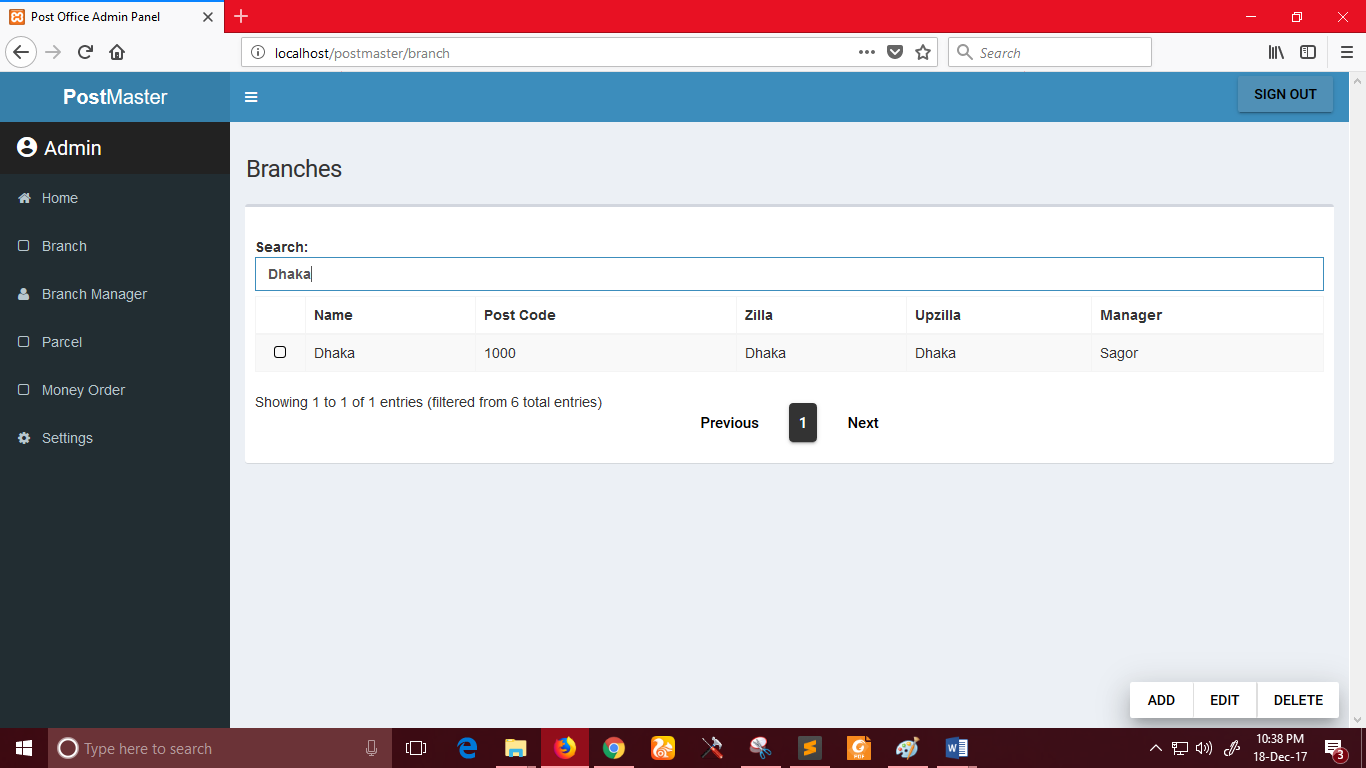


**Branch Page:**

System admin can see the list of all branch

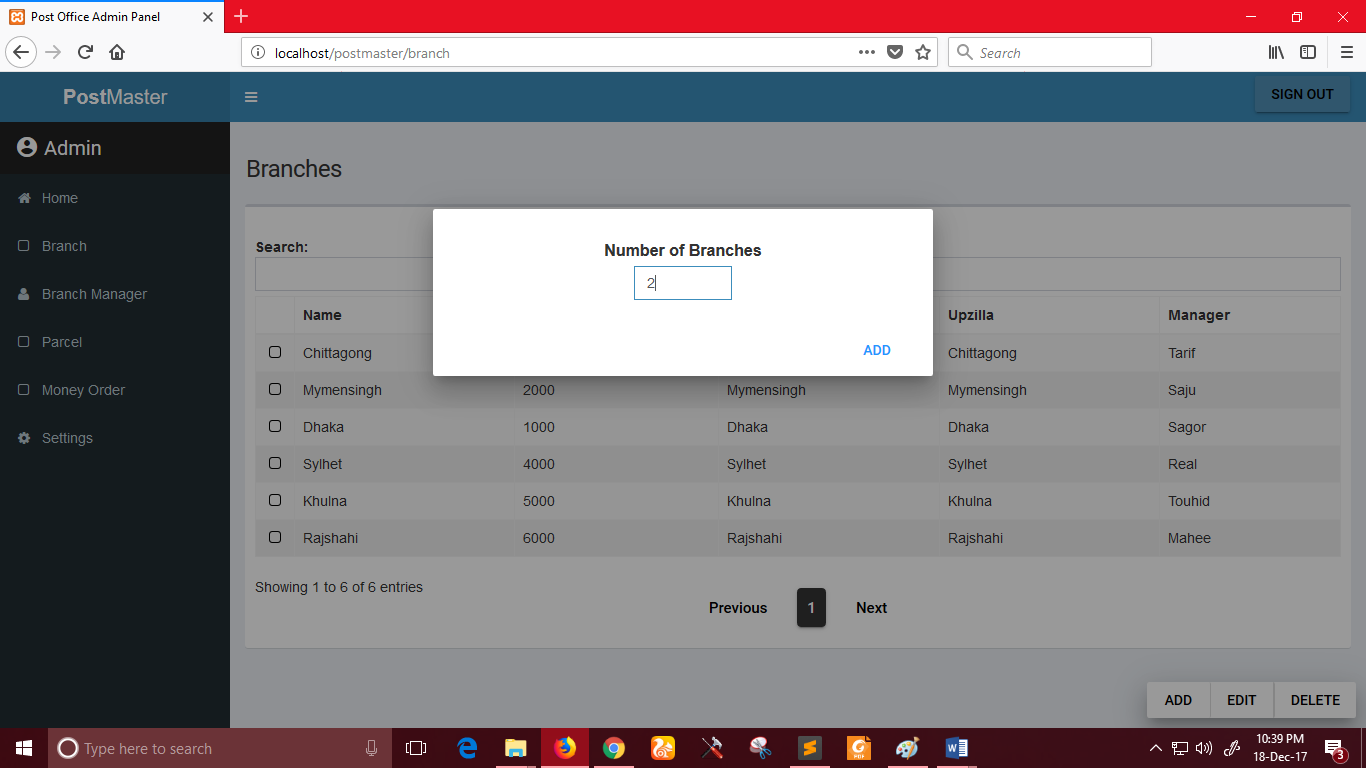


He/she can also search and specified branch with any key related with it

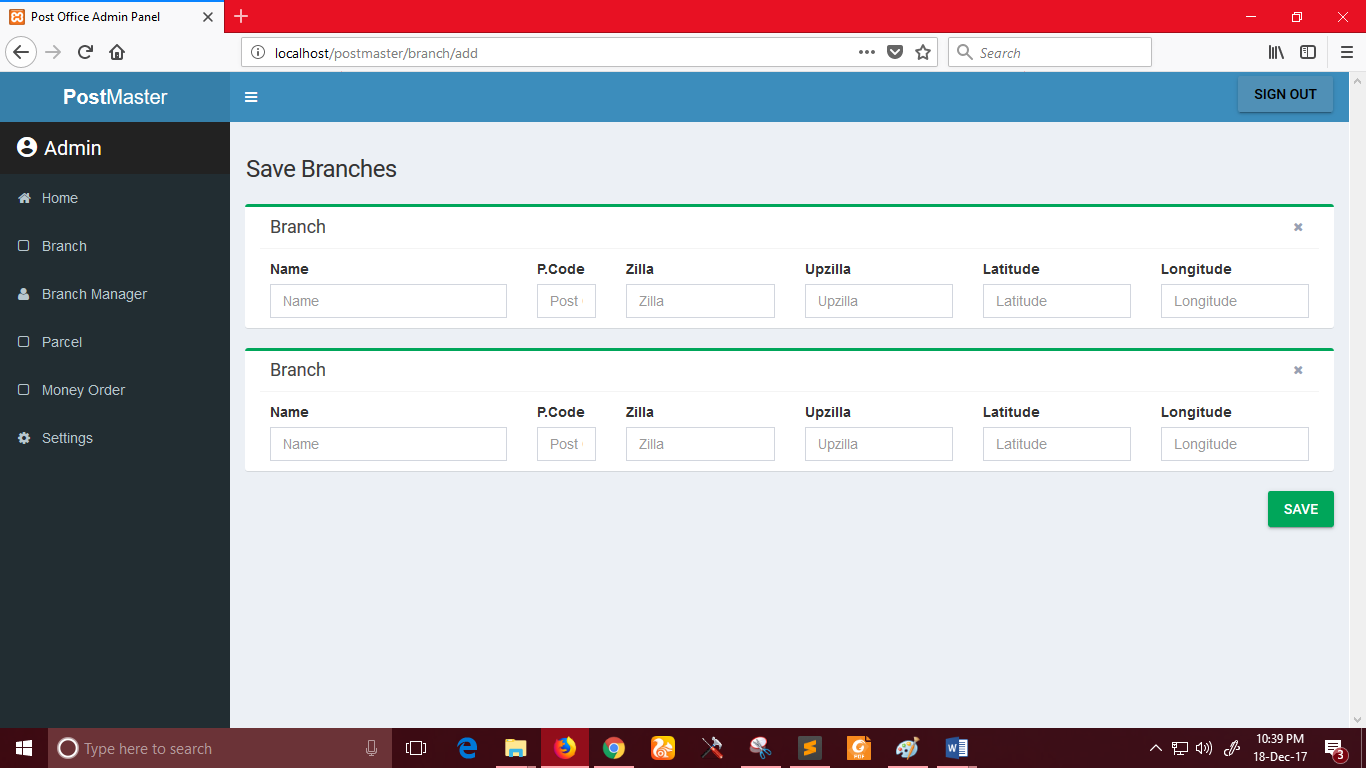


Add Branch:

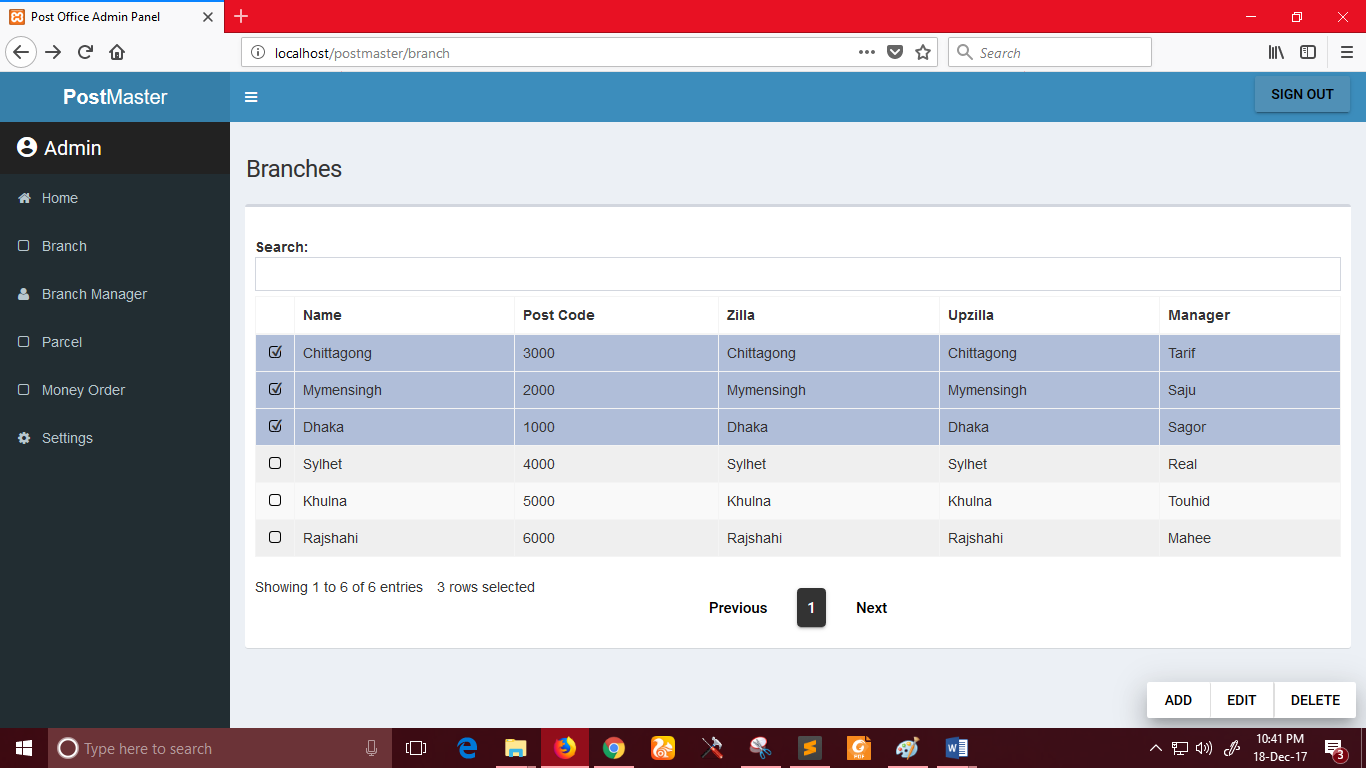
After clicking “Add” system admin can see the following window. He can create multiple branch at once.

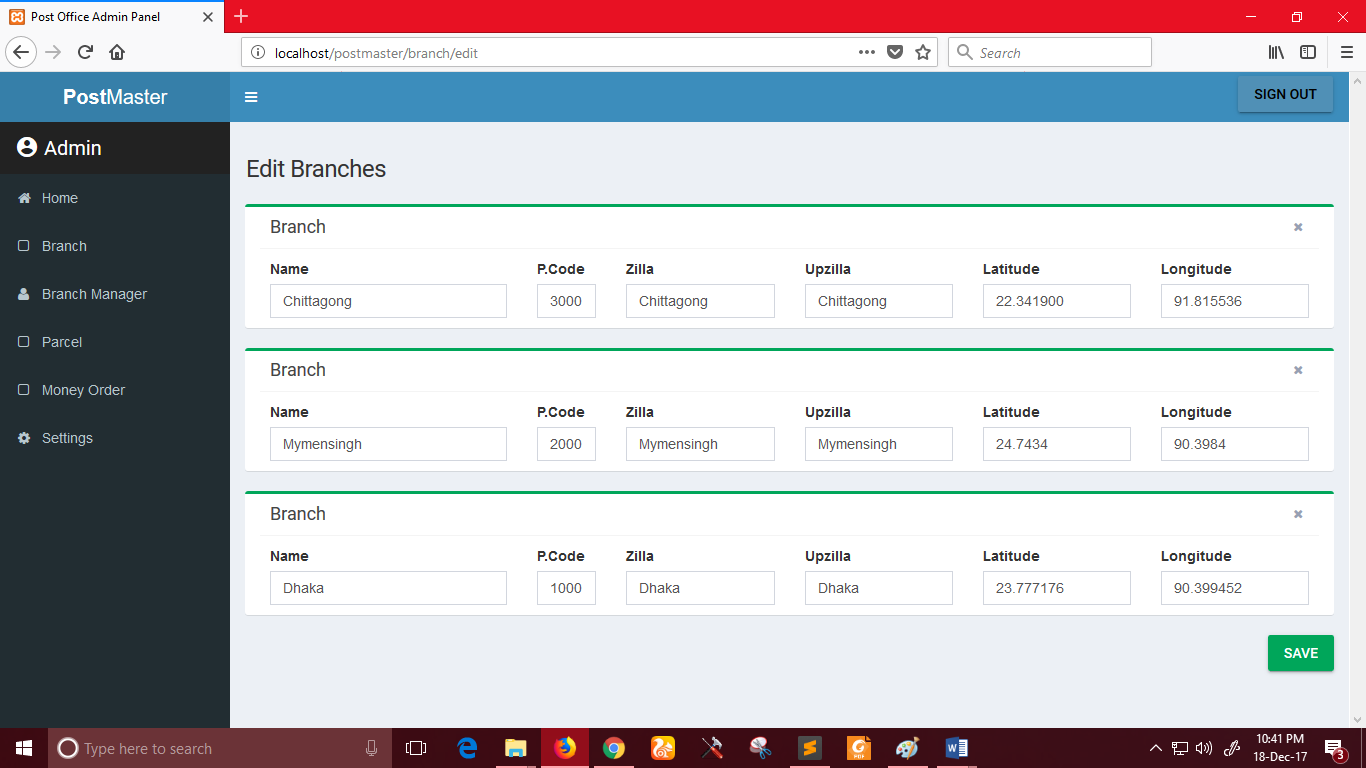


Syetem admin should fill up the branch name, post code, respective zilla, upzilla, latitude, longitude of a branch to create.



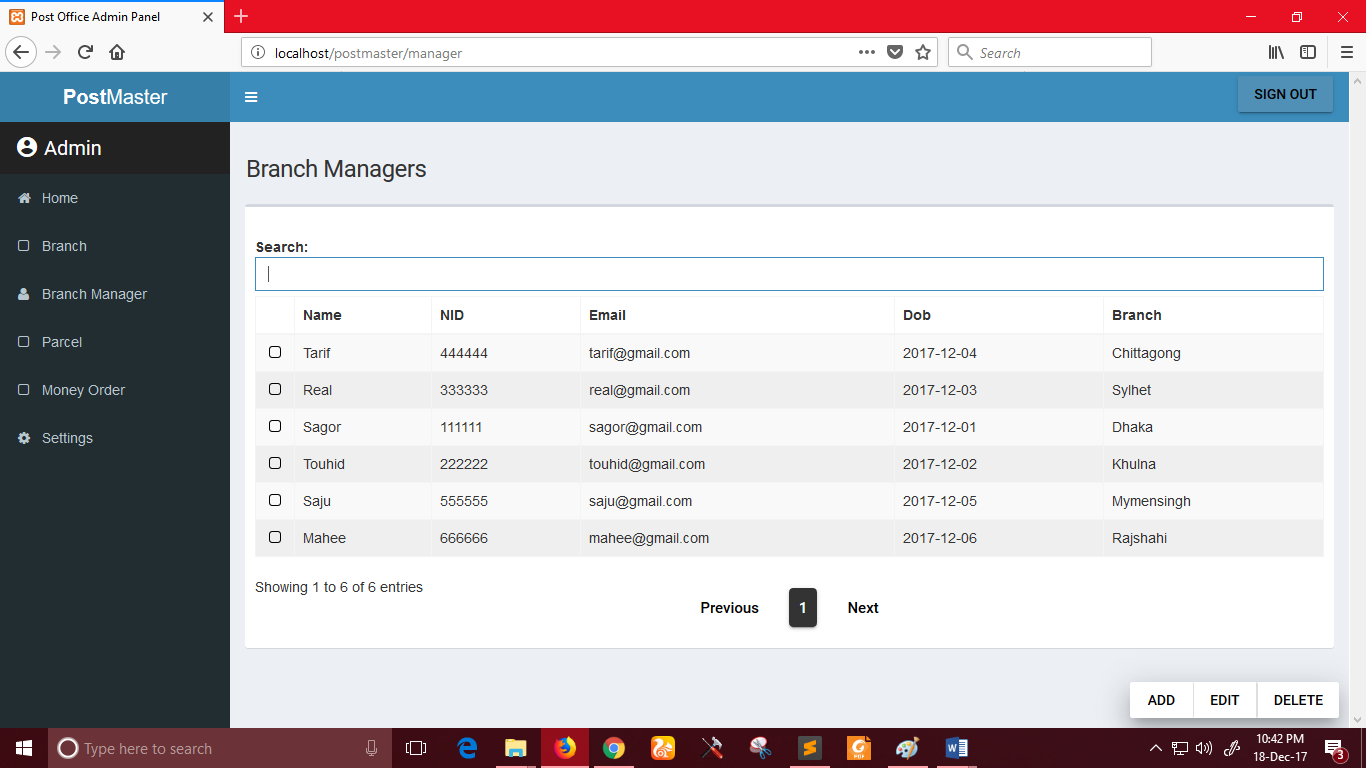
He/she can also edit or delete branch by selecting the branches as following



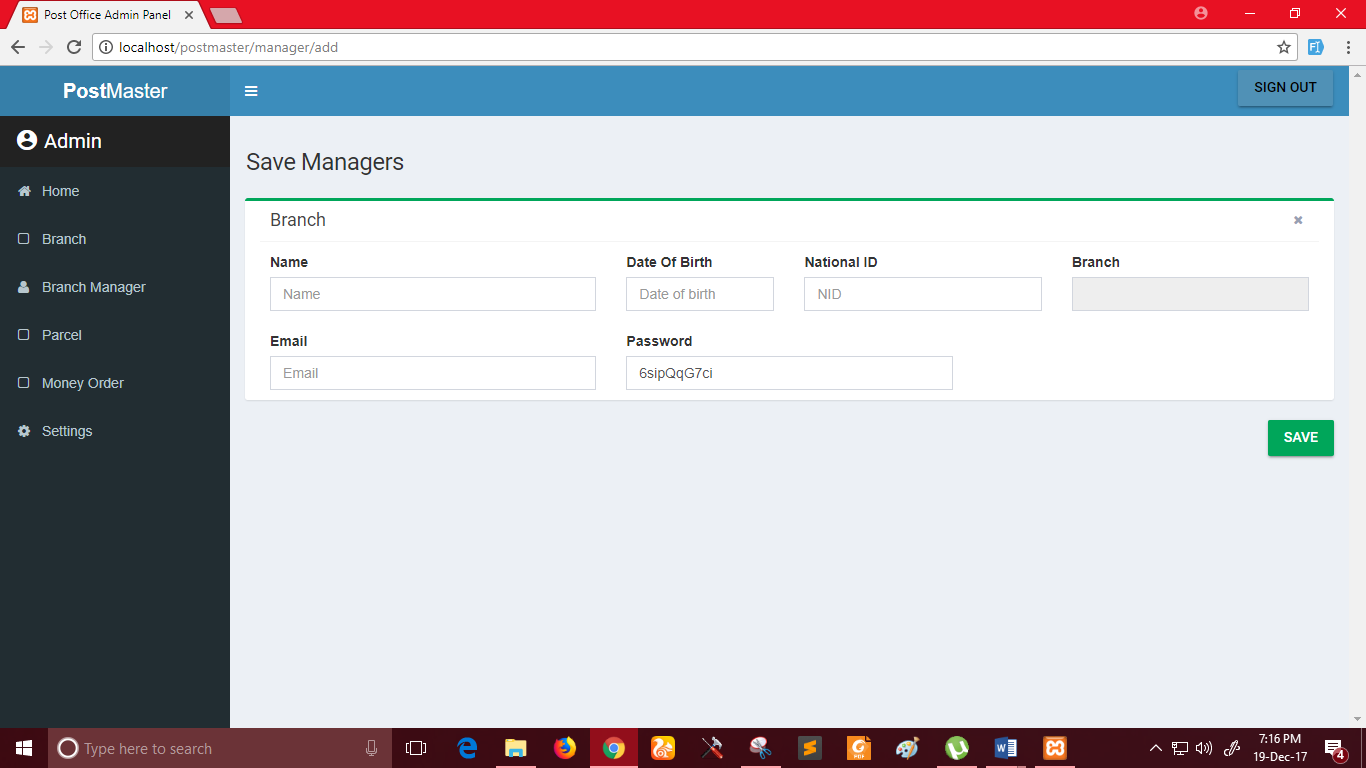


**Branch Manager page:**

System admin can create and assign branch managers to respective branches.

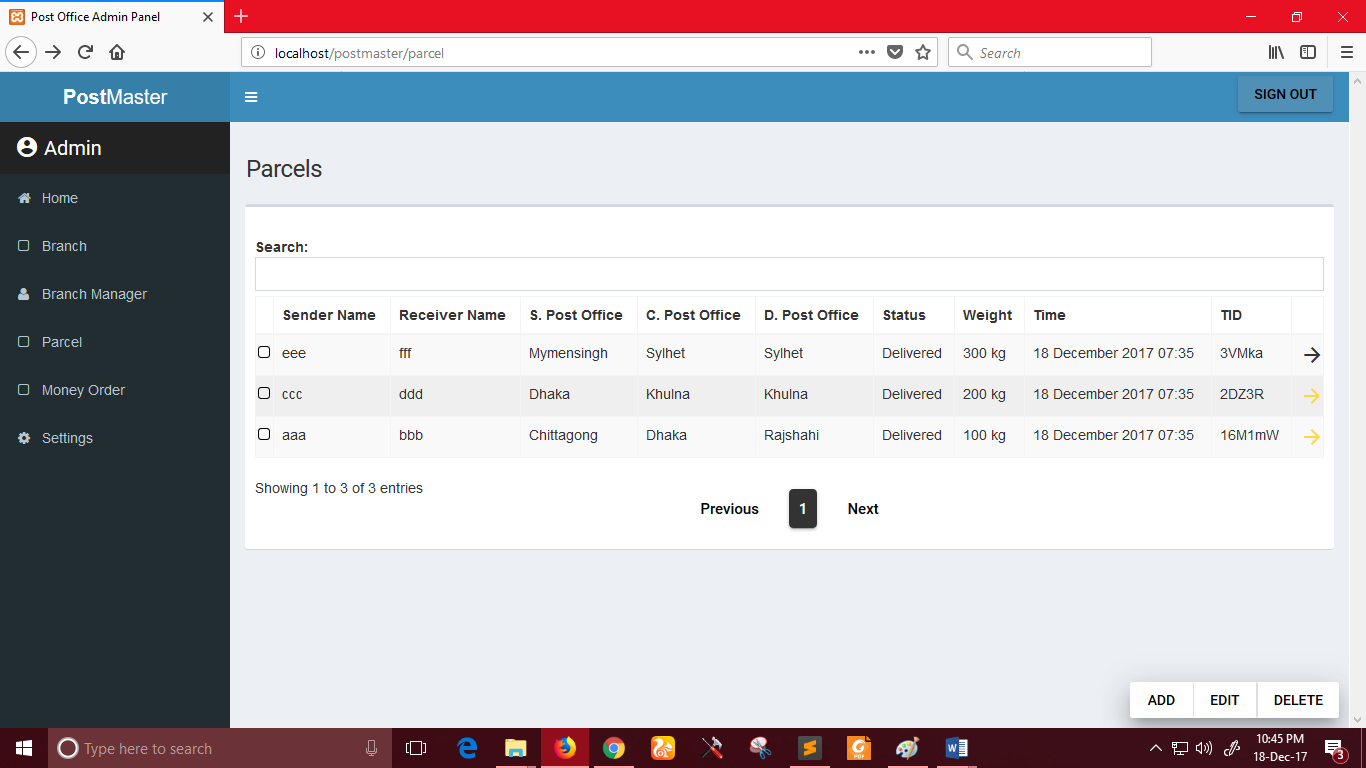


Here he/she should add the details of a branch manager and assign it to a respective branch.

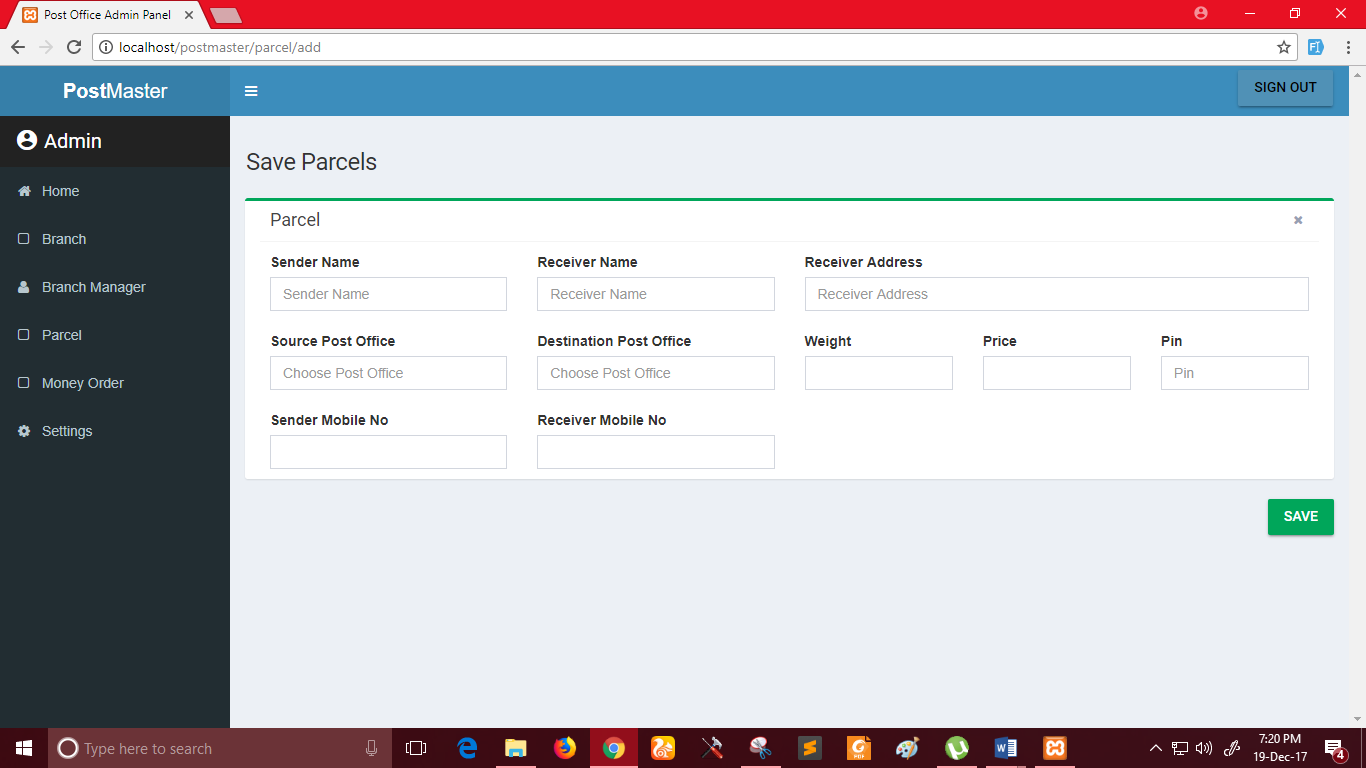


**Parcel page:**

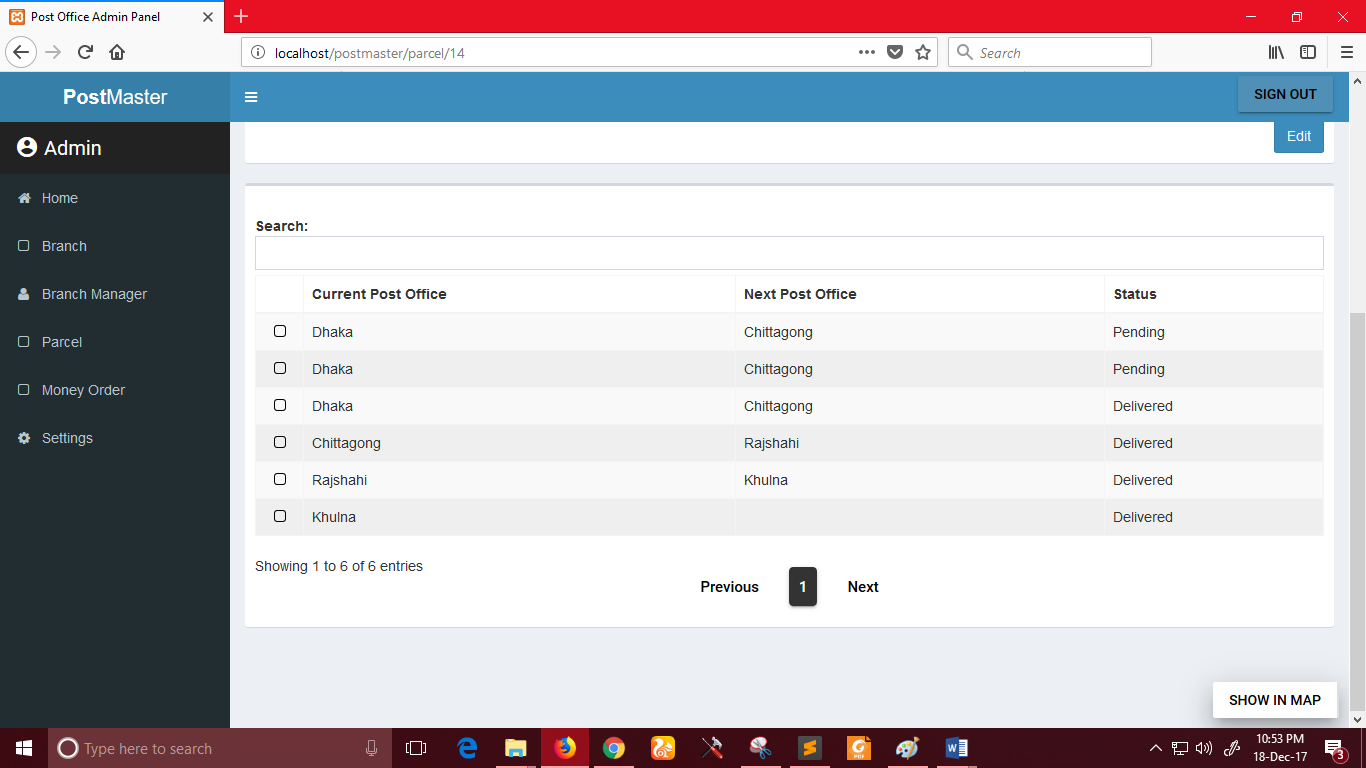
System admin can also have the list of parcels and money orders and can add parcels or money orders to the system.



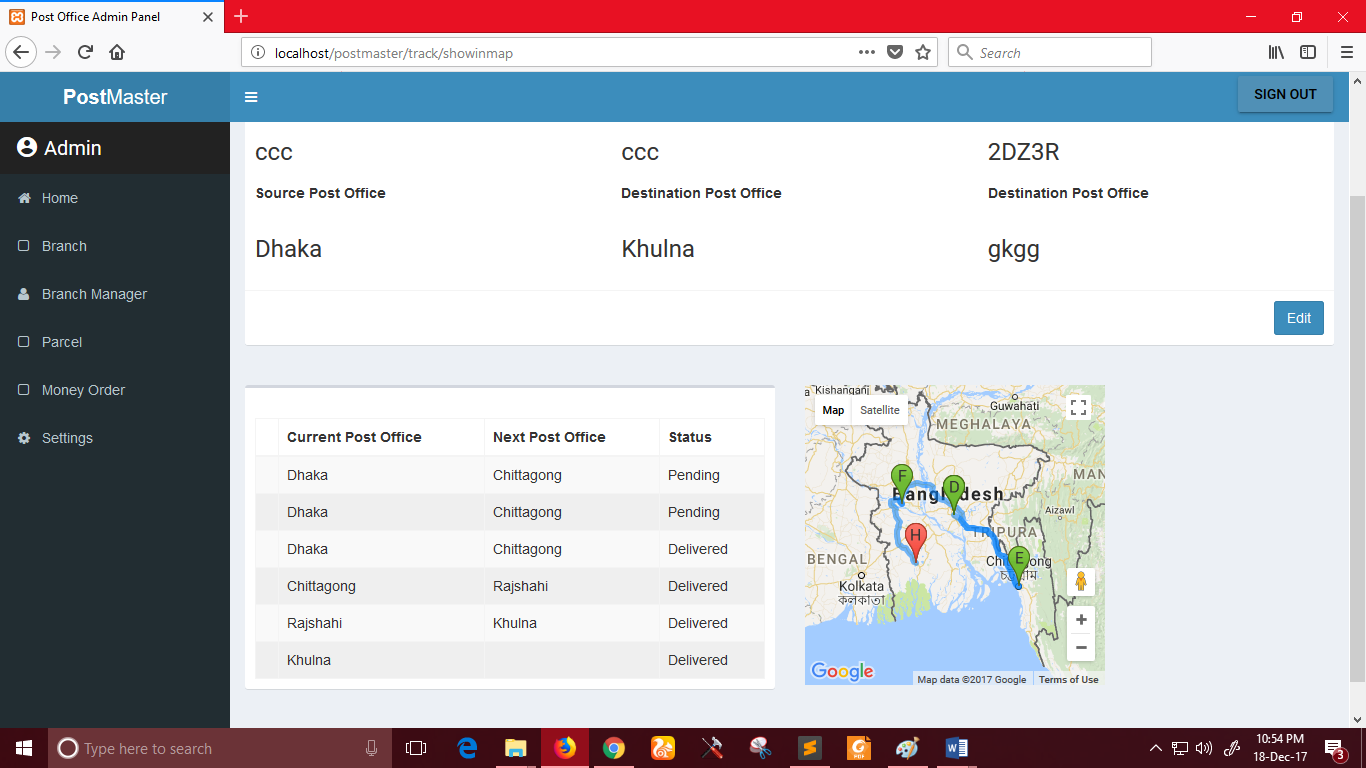
Here the details of product should be inserted.



After clicking any parcel/money order he/she can see the track of that product as current and next post office of it.



After clicking “Show in map” option he/she can see the track and location of the product in google map.

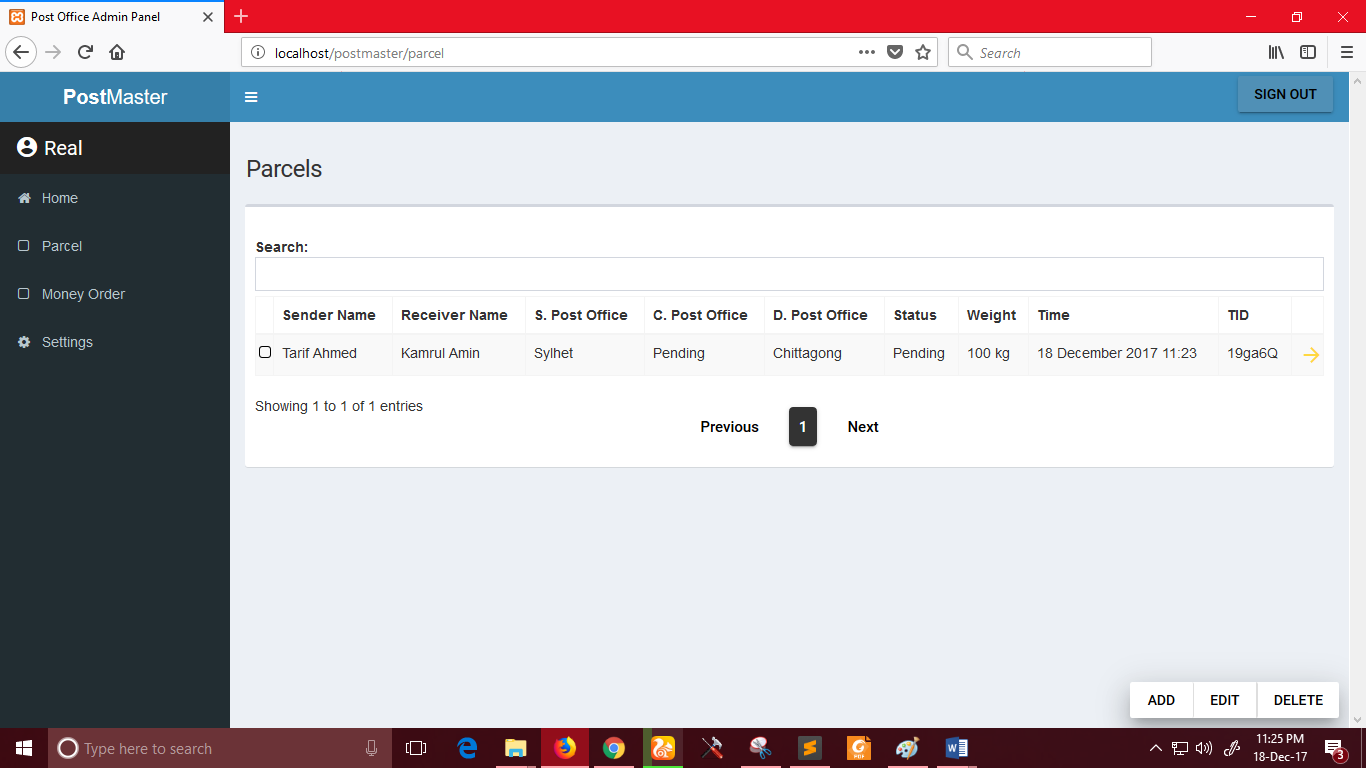


Money order processing is as same as the parcels.



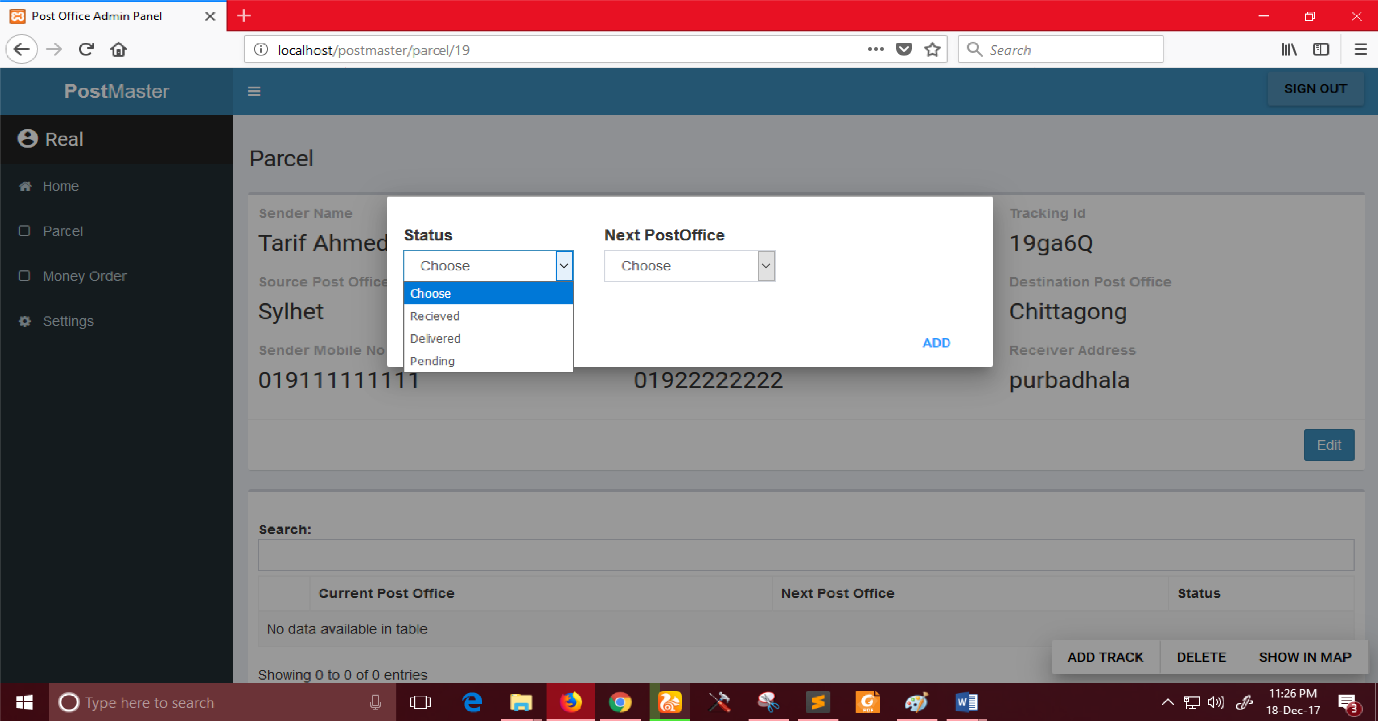
**Branch Managers:**

After successful log in as a branch manager he/she can see the parcel and money order list assigned to his/her branch. He/she can also add new parcels and money orders in his branch. Also can edit and delete as system admin.



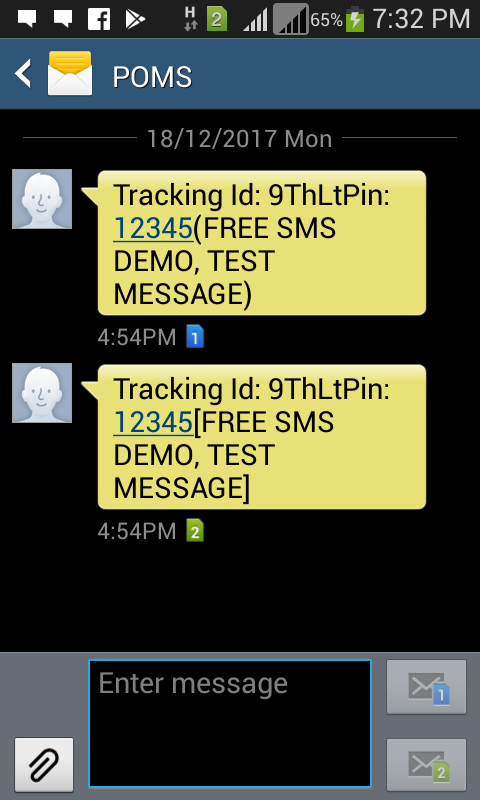
**Add track:**

The main responsibility of a branch manager is to track a product to its next branch and repeating until it reaches its destination branch. Here as following every branch manager must have to track the product assigned to his/her branch and select the current status of it.

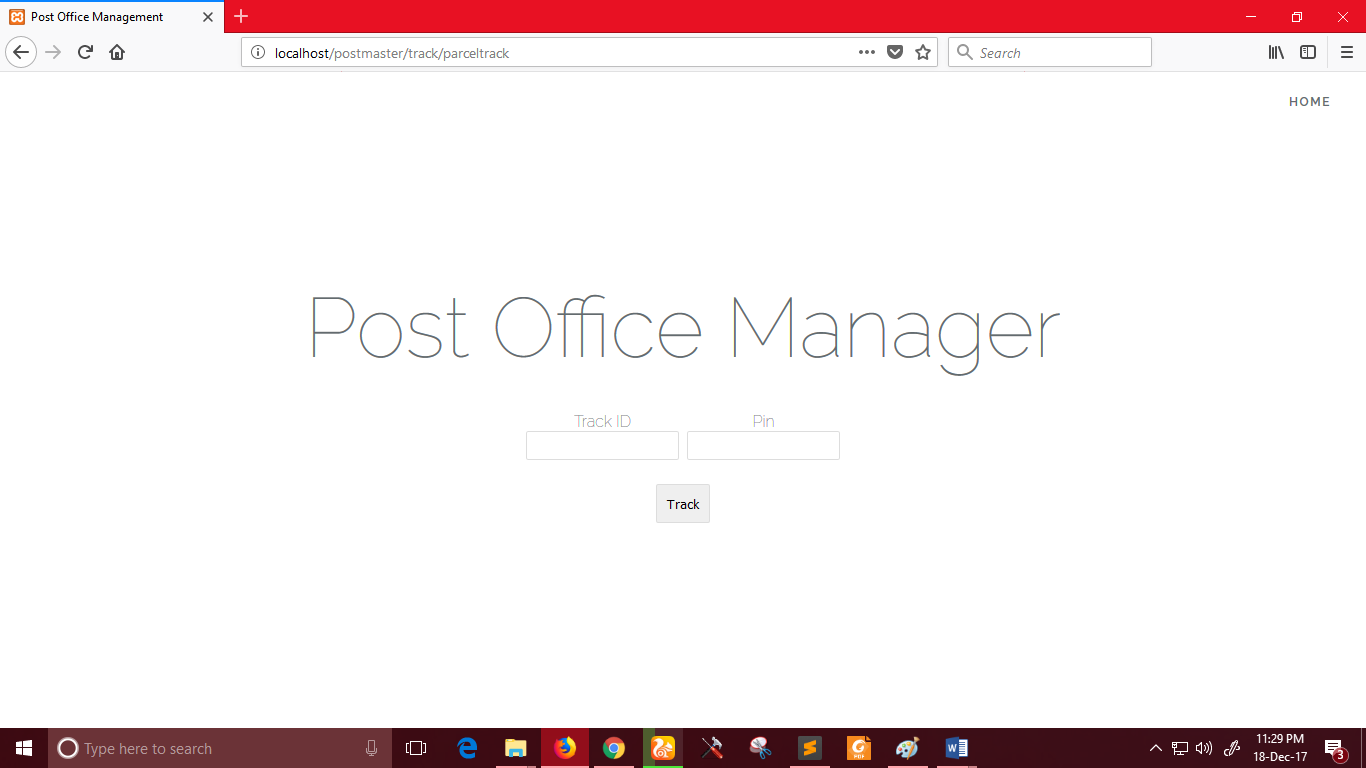


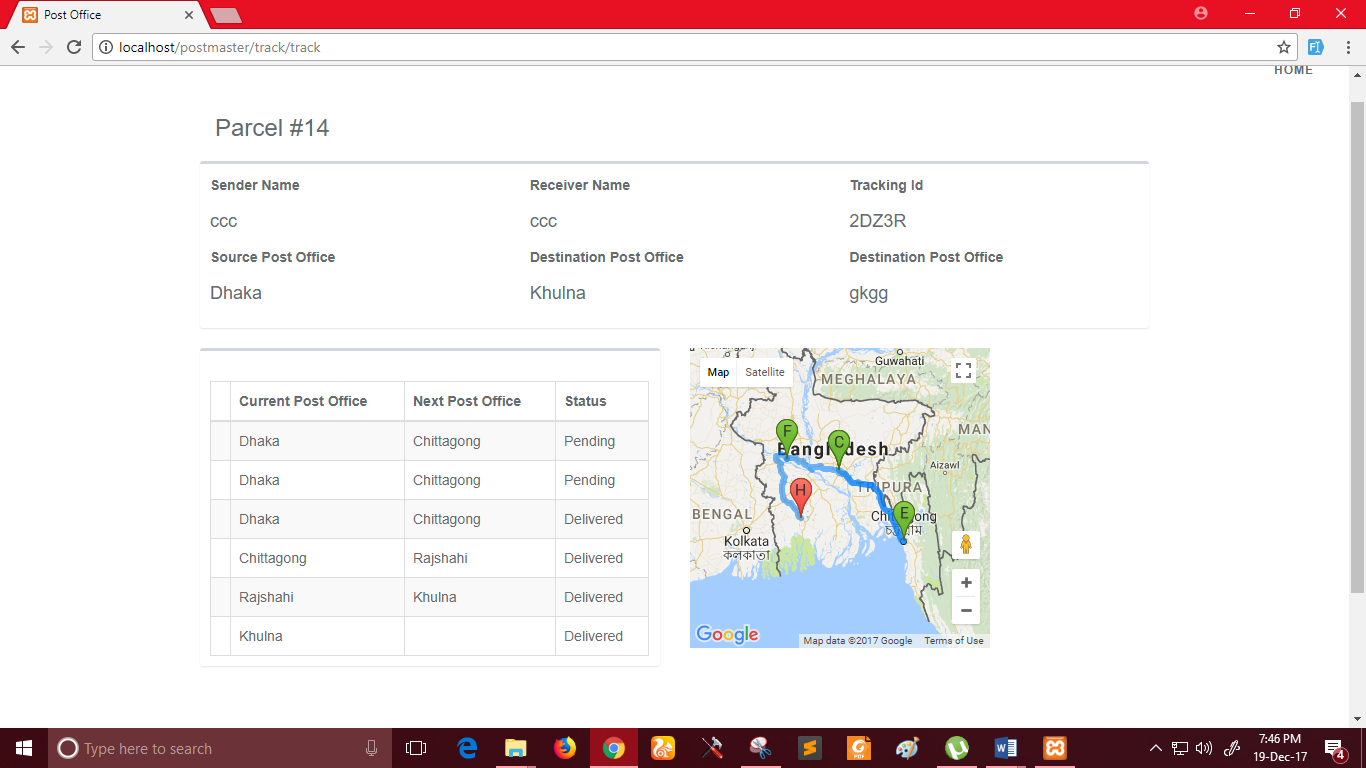
**Tracking Product (Customer):**

When a customer sends a product to its destination, then the phone number of customer and receiver of the product are included to the system. An auto generated track id and PIN given by the manager is send through mobile phone to the senders and receivers phone number.



With the given track id and pin number customer can track the current location of his/ her product





# **Chapter 12**

# **Conclusion**

After a lot of hard working, at last I am able to complete the final SRS, Design and Testing report on Post Office Management System Software. Here I have tried my best to identify necessary requirements from my stakeholders and based on the requirements, I have illustrated different models with diagrams which will help the developers, software designers and other people associated with it to understand about the system and to do their tasks in a better way. Students as well as teachers may use this document as academic learning resource. I hope that the readers will get benefit from the document.

There are some limitations of the project. It is not an overall system for post office because it also provides some services which are not implemented here.

The future plan with is project is to add the rest of the features of total post office system which will make the whole system automated. Where financial services, postal orders in foreign countries, EMS are also will be implemented to system which will make the post office service better and reliable. My future plan is to find out rest of the requirements and implement them in my post office management system project.

# **Chapter 13**

# **Reference**

1. Pressman, Roger S. Software Engineering: A Practitioner's Approach (7th ed.). Boston, Mass: McGraw-Hill. ISBN 0-07-285318-2
2. Database System Concepts, 5th Ed. ©Silberschatz, Korth and Sudarshan
3. Sommerville, I. Software Engineering, 7th ed. Harlow, UK: Addison Wesley, 2006