Billing System

Report on Project of Management Information System

**Billing System**

**Report on Project of MIS**

**BUS 602**

**Submitted to:**

Dr. Md. Mahbubul Alam Joarder

Professor

Amit Seal Ami

Lecturer

**Submitted by:**

Md. Jubair Ibna Mostafa (BSSE0614)

Rubaiyat Jahan Mumu (BSSE0620)

Ariful Hoque Maruf (BSSE0621)

Syeda Tasnim Fabiha (BSSE0622)

Fazle Mohammed Tawsif (BSSE0628)

Institute of Information Technology

University of Dhaka

**Submission Date:**

17.11.2016

**Letter of Transmittal**

17.11. 2016

Amit Seal Ami

Lecturer

Institute of Information Technology

University of Dhaka

**Subject: Submission of term report on Project of MIS; “Billing System”.**

Dear Sir,

With due respect, we are pleased to submit the final report on “Billing System” that you had asked. In this report, we had to give our best effort albeit there might be some shortcomings. We would be highly obliged if you consider those from excusable point.

Yours sincerely

Md. Jubair Ibna Mostafa (BSSE0614)

Rubaiyat Jahan Mumu (BSSE0620)

Ariful Hoque Maruf (BSSE0621)

Syeda Tasnim Fabiha (BSSE0622)

Fazle Mohammed Tawsif (BSSE0628)

3rd Year, 6th Semester, 6th Batch

Institute of Information Technology

University of Dhaka

Session: 2013-14

**Document Authentication**

The following Project Report has been accepted and approved by following:

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Title** | **Date** | **Signature** |
| Md. Jubair Ibna Mostafa (BSSE0614) |  | 17.11.16 |  |
| Rubaiyat Jahan Mumu (BSSE0620) |  |  |  |
| Ariful Hoque Maruf (BSSE0621) |  |  |  |
| Syeda Tasnim Fabiha (BSSE0622) |  |  |  |
| Fazle Mohammed Tawsif (BSSE0628) |  |  |  |

**Document Approval**

Signature

**Acknowledgement**

By the Grace of ALMIGHTY ALLAH we completed the report on our MIS project **BILLING SYSTEM**.

We would like to thank whole-heartedly our supervisor Amit Seal Ami, Lecturer, Institute of Information Technology, University of Dhaka, for giving us guideline about how can we prepare this report. He helped us a lot by sharing his valuable knowledge with us.

**Abstract**

***Table of Contents***

[Chapter 1: Project Plan of “Billing System” 12](#_Toc467088177)

[1.1 Background of the Project: 12](#_Toc467088178)

[1.2 About the project: 12](#_Toc467088179)

[1.3 Scope of Our Project: 12](#_Toc467088180)

[1.4 Project Scheduling: 12](#_Toc467088181)

[1.5 Conclusion 13](#_Toc467088182)

[Chapter 2: Introduction of SRS 14](#_Toc467088183)

[2.1 Purpose 14](#_Toc467088184)

[2.2 Intended Audience 14](#_Toc467088185)

[2.3 Conclusion 15](#_Toc467088186)

[Chapter 3: Inception 16](#_Toc467088187)

[3.1 Introduction 16](#_Toc467088188)

[3.2 Establishing the ground work 16](#_Toc467088189)

[3.2.1 List of stakeholders 17](#_Toc467088190)

[3.2.2 Recognizing multiple viewpoints 18](#_Toc467088191)

[3.2.3 Working towards collaboration 19](#_Toc467088192)

[3.2.4 Requirement Questionnaire 20](#_Toc467088193)

[3.3 Conclusion 20](#_Toc467088194)

[Chapter 4: Elicitation 22](#_Toc467088195)

[4.1 Introduction 22](#_Toc467088196)

[4.2 Eliciting Requirements 22](#_Toc467088197)

[4.2.1 Collaborative Requirements Gathering 23](#_Toc467088198)

[4.2.2 Quality Function Deployment 23](#_Toc467088199)

[4.2.3 Usage Scenario 25](#_Toc467088200)

[4.2.4 Elicitation Work Product 27](#_Toc467088201)

[4.3 Conclusion 28](#_Toc467088202)

[Chapter 5: Scenario Based Modeling 29](#_Toc467088203)

[5.1 Introduction 29](#_Toc467088204)

[5.2 Definition of Use case 29](#_Toc467088205)

[5.3 Use Case Diagrams 30](#_Toc467088206)

[5.3.1 System Description of Level-0 Use Case Diagram 30](#_Toc467088207)

[5.3.2 System Description from Level-1 Use Case Diagram 31](#_Toc467088208)

[5.3.3 Level-2.1 Use Case Diagram 34](#_Toc467088209)

[5.3.4 System Description from Level-2.2 Use Case Diagram 34](#_Toc467088210)

[5.3.5 System Description from Level-2.3 Use Case Diagram 37](#_Toc467088211)

[5.3.6 System Description from Level-2.4 Use Case Diagram 39](#_Toc467088212)

[5.3.7 System Description from Level-3 Use Case Diagram 41](#_Toc467088213)

[5.4 Activity & Swim Lane Diagrams 43](#_Toc467088214)

[5.5 Conclusion 57](#_Toc467088215)

[Chapter 6: Data Model 58](#_Toc467088216)

[6.1 Data Modeling Concept 58](#_Toc467088217)

[6.2 Data Objects 58](#_Toc467088218)

[6.2.1 Grammatical parsing & analysis 58](#_Toc467088219)

[6.2.2 Identify Data Objects 60](#_Toc467088220)

[6.2.3 Data Object Relation 63](#_Toc467088221)

[6.2.4 E-R Diagram 64](#_Toc467088222)

[6.3.5 Schema Form (Tables) 65](#_Toc467088223)

[6.3 Conclusion 66](#_Toc467088224)

[Chapter 7: Class Based Model 67](#_Toc467088225)

[7.1 Class Based Modeling Concept 67](#_Toc467088226)

[7.2 Identifying Analysis Class 67](#_Toc467088227)

[7.2.1 Class Responsibility Collaboration (CRC) 76](#_Toc467088228)

[7.3 Conclusion 76](#_Toc467088229)

[Chapter 8: Flow Oriented Model 77](#_Toc467088230)

[8.1 Introduction 77](#_Toc467088231)

[8.2 Data Flow Diagram 77](#_Toc467088232)

[8.3 Conclusion 81](#_Toc467088233)

[Chapter 9: Behavioral Model 82](#_Toc467088234)

[9.1 State Transition Diagram 82](#_Toc467088235)

[9.2 Sequence Diagram 85](#_Toc467088236)

[9.3 Conclusion 86](#_Toc467088237)

[Chapter 10: User Interface 87](#_Toc467088238)

[10.1 Introduction 87](#_Toc467088239)

[10.2 Basic Interface 87](#_Toc467088240)

[10.3 Conclusion 88](#_Toc467088241)

[Chapter 11: Conclusion 89](#_Toc467088242)

[11.1 Obstacles 89](#_Toc467088243)

[11.2 Achievements 89](#_Toc467088244)

[11.3 Future Plan 90](#_Toc467088245)

[11.4 Last Few Words 90](#_Toc467088246)

[Appendix 91](#_Toc467088247)

***Index of Figures***

[Figure 1: Level-0 Use Case Diagram 30](file:///D:\Projects\billingSystem\project-billing-system\Documents\SRS\BillingSystem_Documentation.docx#_Toc467088258)

[Figure 2: Level-1 Use Case Diagram 32](#_Toc467088259)

[Figure 3: Level-2.1 Use Case Diagram 33](#_Toc467088260)

[Figure 4: Level-2.2 Use Case Diagram 35](file:///D:\Projects\billingSystem\project-billing-system\Documents\SRS\BillingSystem_Documentation.docx#_Toc467088261)

[Figure 5: Level-1.3 Use Case Diagram 37](file:///D:\Projects\billingSystem\project-billing-system\Documents\SRS\BillingSystem_Documentation.docx#_Toc467088262)

[Figure 6: Level-2.4 Use Case Diagram 39](file:///D:\Projects\billingSystem\project-billing-system\Documents\SRS\BillingSystem_Documentation.docx#_Toc467088263)

[Figure 7: Level-3 Use Case Diagram 41](file:///D:\Projects\billingSystem\project-billing-system\Documents\SRS\BillingSystem_Documentation.docx#_Toc467088264)

[Figure 13: STD for User Class 82](file:///D:\Projects\billingSystem\project-billing-system\Documents\SRS\BillingSystem_Documentation.docx#_Toc467088265)

[Figure 18: User Interface of Billing System (Home) 86](file:///D:\Projects\billingSystem\project-billing-system\Documents\SRS\BillingSystem_Documentation.docx#_Toc467088266)

[Figure 19: User Interface of Billing System (Set Goal) 87](file:///D:\Projects\billingSystem\project-billing-system\Documents\SRS\BillingSystem_Documentation.docx#_Toc467088267)

# Chapter 1: Project Plan of “Billing System”

This chapter covers the project proposal and feasibility of the proposal along with background study, product and business perspective, the scopes and some preliminary idea of our web application.

## 1.1 Background of the Project:

The implementation and deployment of the project Billing System is for the purpose of management of the billing procedure of Institute of information Technology. Billing system is an information system. Management of information system is the purpose of the course, Management of Information System(MIS). Implementing an automated billing system will increase the productivity of employees while decreasing the maintenance cost, time and make life easier. So, the necessity of an automated billing system is mandatory for IIT.

## 1.2 About the project:

## 1.3 Scope of Our Project:

## 1.4 Project Scheduling:

|  |  |  |
| --- | --- | --- |
| **Start Date** | **End Date** | **Project State and Objectives** |
| , 2016 | , 2016 | Project Proposal, meeting with supervisor about our idea |
| , 2016 | , 2016 | Planning , thinking about project story , features and  Learning Technology |
| **Start Date** | **End Date** | **Project State and Objectives** |
| , 2016 | 2016 | Start constructing SRS document & project report, choose tools, environment and learning technology |
| 2016 | 2016 | Complete SRS document & project report, Start design and implementation |
| 2016 | ---------------------- | Final report submission |
| ,2016 | , 2016 | Start full phase implementation |
| , 2016 | ------------------------ | Project Submission |

## 1.5 Conclusion

# Chapter 2: Introduction of SRS

This chapter is a part of our software requirement specification for the project “Billing System”. In this chapter we focus on the intended audience for this project.

## Purpose

This document briefly describes the software requirement analysis of **Billing System**. 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 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.

## 2.2 Intended Audience

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

* The customer will use this SRS to verify that the developer team  
  has created a product that is acceptable to the customer.
* The project managers of the developer team will use this SRS to plan  
  milestones and a delivery date, and ensure that the developing team  
  is on track during development of the system.
* 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.
* 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.
* 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.

## 2.3 Conclusion

This analysis of the audience helped us to focus on the users who will be using our system. This overall document will help each and every person related to this project to have a better idea about the project.

# Chapter 3: Inception

In this chapter, the Inception part of the software requirement analysis of our project will be discussed briefly.

## 3.1 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. At project inception, we establish a basic understanding of the problem, the people who want a solution, the nature of the solution that is desired, and the effectiveness of preliminary communication and collaborations between the other stakeholders and the software team.

## 3.2 Establishing the ground work

To establish the groundwork we have worked with the following factors related to the inception phases:

* List of stakeholders
* Recognizing multiple viewpoints
* Working towards collaboration
* Requirements questionnaire

### 3.2.1 List of stakeholders

Stakeholder refers to any person or group who will be affected by the system directly or indirectly. Stakeholders include end-users who interact with the system and everyone else in an organization that may be affected by its installation. At inception, a list of people who will contribute input as requirements are elicited (Chapter 3) is created. The initial list will grow as stakeholders are contacted because every stakeholder will be asked: “Whom else do you think I should talk to?”

To identify the stakeholders we consulted with the teachers and student who are cyclists or runner asked those following questions:

* What will be the main purpose of using the web application?
* Who will be using the project outcomes?
* Where we will find the resources we need to get the project done?
* Whose work will my project affect? (During the project and also once the project is completed).

Concluding thoughts on Stakeholders thoughts we identified that the common stakeholders of this system are all the faculties and the worker related to the existing manual billing system.

**Faculties of IIT:**

**Director of IIT:**

**Coordinator of IIT:**

**Other people working in billing process in IIT:**

**Developers:** Developers will build the system and work on further development of the project. They will be responsible for any type of interruption and they will solve it.

### 3.2.2 Recognizing multiple viewpoints

Though all types of stakeholders we identified will use the application on same purposes, each of them has slightly different view of the system. So we have to identify the multiple views of requirements. Assumptions are given below:

**Viewpoint of Faculties:**

* User friendly efficient system
* System must generate calculation about the speed of his/her riding
* The generated value from the system should be error free
* System must show the location on map
* Strongly authenticated system
* System must be able to allow other athletes to communicate with one another
* Easily operable system

**Viewpoint of Coordinator:**

* User friendly system
* System must generate the average speed of his/her daily running
* Calculation must be specific and exact
* Map representation is a must
* Monthly performance should be generated
* System must be strongly authenticated
* Activities or progress of other runners must be shareable with other users
* Easily operable system

**Viewpoint of Director:**

* User friendly system
* System must be strongly authenticated
* Calculation of the amount of burn calorie must be included
* It will be easy to operate the system

**Viewpoint of Other Workers:**

**Viewpoint of Developers:**

* All the requirements will be well defined
* No major requirement change will come in the development phase
* All the stakeholders will be available for further information and consult

### 3.2.3 Working towards collaboration

Every stakeholders has their own requirements from their own point of view. We followed following steps to merge all the requirements. We-

* Identified the common and conflicting requirements
* Categorized the requirements
* Take priority points for each requirements from stakeholders
* Make final decision about the requirements

**Common requirements:** We found some requirements are all the same from different user

* User friendly and efficient system
* Easy to operate
* Authentication
* Making a community among all other user
* Communicate with other user through activity
* Error free and exact calculation

**Conflicting requirements:** We found some requirements conflicting each other. We had to trade-off between the requirements.

* Availability of all requirements within the budget
* No ambiguous requirement
* Easy access
* Strong authentication and high security
* No harmful effects on existing technology

**Final requirements:** We finalized following requirements for the system by categorizing and prioritizing the requirements.

* Error free system (Maximum 5% error may be considerable)
* Allow the users login and logout
* User friendly and efficient system
* Secure way to communicate with other user
* Automated entry of student information
* Central database contain all user information and other analysis information derived from the GPS data

### 3.2.4 Requirement Questionnaire

We set our first set of context-free questions focuses on the customer and other stakeholders, overall project goals and benefits. The questions are mentioned above. These questions helped us to identify all stakeholders, measurable benefit of the successful implementation and possible alternatives to custom software development. 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. The final set of question focused on the effectiveness of the communication activity itself.

## 3.3 Conclusion

Inception phase helped us to establish basic understanding about the project **Billing System**, identify the people who will be benefited using this system, define the nature of the project and establish a preliminary communication with our stakeholders.

In our project, we have established a basic understanding of the problem, the nature of the solution that is desired and the effectiveness of preliminary communication and collaboration between the stake-holders and the software team. More studies and communication will help both side (developer and client) to understand the future prospect of the project. Our team believes that the full functioning document will help us to define that future prospect.

# Chapter 4: Elicitation

After discussing on inception part, we need to keep focus on the elicitation part. So this chapter specifies the elicitation part.

## 4.1 Introduction

Requirements elicitation is a part of requirement engineering that is the  
practice of gathering requirements from the users, customers, and other  
stakeholders. We have faced many problems like understanding the  
problems, problems of making questions for the stakeholders, problems of  
less communication with the stakeholders for time limitation, problems of  
volatility. Though it is not too easy to gather requirements within a very short  
time, we have surpassed these problems in an organized and systematic  
manner.

## 4.2 Eliciting Requirements

We have seen Question and Answer (Q&A) approach in the previous chapter, where the inception phase of requirement engineering has been described. The main task of this phase is to combine the elements of problem solving, elaboration, negotiation and specification. The collaborative working approach of the stakeholders is required to elicit the requirements. We have finished the following tasks for eliciting requirements-

* Collaborative Requirements Gathering
* Quality Function Deployment
* Usage Scenarios
* Elicitation work products

### 4.2.1 Collaborative Requirements Gathering

Actually, we met with many stakeholders in the inception phase such as cyclists, runners, our classmates and teacher as the common people and also the developers. These meetings created an indecisive state for us to elicit the requirements. To solve this problem we have met with the stakeholders (who are acting a vital rule in the whole process) again to elicit the requirements. A slightly different scenario from these approaches has been found.

Following activities have been completed to accomplish this task.

* The meetings were conducted with the cyclists, runner and common people; they were questioned about their requirements and expectations from the web application we are developing.
* They were asked about the problems they are facing without using any relevant application regarding sports activity.
* At last we selected our final requirement list from the meetings.

### 4.2.2 Quality Function Deployment

Quality Function Deployment (QFD) is a technique that translates the needs of the customer into technical requirements for software. Ultimately the goal of QFD is to translate subjective quality criteria into objective ones that can be quantified and measured and which can then be used to design and manufacture the product. It is a methodology that concentrates on maximizing customer satisfaction from the software engineering process. So we have followed this methodology to identify the requirements for the project. The requirements, which are given below, are identified successfully by the QFD.

#### 4.2.2.1 Normal Requirements

Normal requirements are generally the objectives and goals that are stated for a product or system during meetings with the customer. The presence of  
these requirements fulfills customers' satisfaction. These are the normal  
requirements for our project.

* + Allow user to do registration
  + Allow valid user to sign in and sign up
  + Check user validity
  + Features to have an idea about the user’s activity
  + Feature that will calculate the amount of burnt calorie
  + Feature that will calculate the speed and distance
  + Allow users to generate graph from the simulation results
  + Map representation
  + Efficient and user friendly
  + The user interface of the system would be easy
  + Allow user to set their goal
  + Allow user to create group with other users for sharing the activity
  + Allow user to leave from a group
  + Allow user to search other using their user name
  + Allow user who created a group to add other user as the member of the group
  + Allow users to share their activity and thoughts
  + Allow user to add their personal information along with their picture to the application
  + Security issue
  + Error free activity

#### 4.2.2.2 Expected Requirements

These requirements are intrinsic to the product or system and may be so elementary that the customer does not explicitly state them. Their absence will be a cause for significant dissatisfaction. Below the expected requirements for our project are briefly described.

* The application shall be easily maintainable.
* The application will be stable.
* The application will be open for future extension and modification.
* All the records and scores will be saved to database for future references
* The application shall be user friendly for all type of users.
* The user interface shall make use of input such as drop downs,  
  check boxes and radio buttons as much as possible to avoid invalid and  
  incorrect input.

#### 4.2.2.3 Exciting Requirements

These requirements are for features that go beyond the customer's expectations and prove to be very satisfying when present. Following are some exciting requirements of out project.

* Users can save graphs as images.
* Different session will be provided for different users.
* Connect user account with Facebook or other social media
* Users can gain weekly/monthly/yearly badge considering their activities
* Users can gain badge after fulfilment of their goal they selected to achieve
* Ranking system among the group members
* User can get their elevation information from the system
* User’s posts can be edited

### 4.2.3 Usage Scenario

The billing system has four kinds of users: an admin, a director, coordinators and normal users. The admin has the supreme authority over the whole system. He can promote any user to a coordinator. Also anytime he can demote a coordinator to a normal user.

Every user needs to be authenticated to enter into the system .Each user must provide following information during registration.

1. User's full Name

2. User's Email Address

3. User's designation

5. Password

There are some categories on which bills are made. If a user wants to make a bill, he enters the name of the bill. Then he selects the category/categories of the corresponding bill and enters the amount for each category. If the category is not listed, the user can request for adding new category. The admin adds new categories based on requests from the users.

Then the user creates a document of the bill and sends this document to the corresponding coordinator of the bill. The coordinator makes a validation check on the bill and then approves or rejects the bill. If the bill is approved or rejected, the user is notified about it. If a bill is rejected, the user can edit the billing document and send it back to the coordinator again for approval. If the bill is approved, the coordinator sends it to the director for final approval. When a bill is sent for final approval, the user is notified about it again. The director approves it and the system notifies the user about the approval. After final approval, user can generate a pdf of the bill and print it out.

**Billing fields:**

* Admission
* Examination
* Extra-curriculum activity
* Games
* Cultural programs
* Picnic
* Script Checking
* Overtime
* Food
* Training Programs
* Regular course
* Short course
* Maintenance
* Buying new things
* Repair

### 4.2.4 Elicitation Work Product

At first we have to know that the output of the Elicitation task may vary because of the dependency on size of the system or product to be built. Here, the elicitation work product includes:

* Making a statement of our requirements for the project
* Making a bounded statement of scope for our system.
* Making a list of customers, users, and other stakeholders who participated in the requirements elicitation.
* Making a list of requirements that are organized by function and domain constraints that apply to each.
* A set of usage scenarios that provide insight into the use of the system.
* Description of the system’s technical environment.

## 4.3 Conclusion

Elicitation process gives us a clear view of the requirements of the stakeholders and develops our understanding of the whole project. It also enables us to deliver a product that will satisfy all the stakeholders. This phase also helps us to identify the requirements, negotiate different approaches and specify a preliminary set of solution requirements in an atmosphere that is conducive to the accomplishment of the goal.

# Chapter 5: Scenario Based Modeling

This chapter describes the scenario based model for the project Billing System.

## 5.1 Introduction

Although the success of a computer-based system or product is measured in many ways, user satisfaction resides at the top of the list. If we understand how end users (and other actors) want to interact with a system, our software team will be better able to properly characterize requirements and build meaningful analysis and design models. Hence, requirements modeling with begins with the creation of scenarios in the form of use cases, activity diagrams, and swim lane diagrams.

## 5.2 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.

## 5.3 Use Case Diagrams

Use case diagram is the non-technical view of overall system. The system is described from the user’s point of view. As this is the first model, it serves as input for creation of other modeling elements.

### 5.3.1 System Description of Level-0 Use Case Diagram

After analyzing the user story we found six 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.

We identified that our all our actors of the system falls into the same category. They will all be our users, who will be using our system to fulfill their billing procedures.

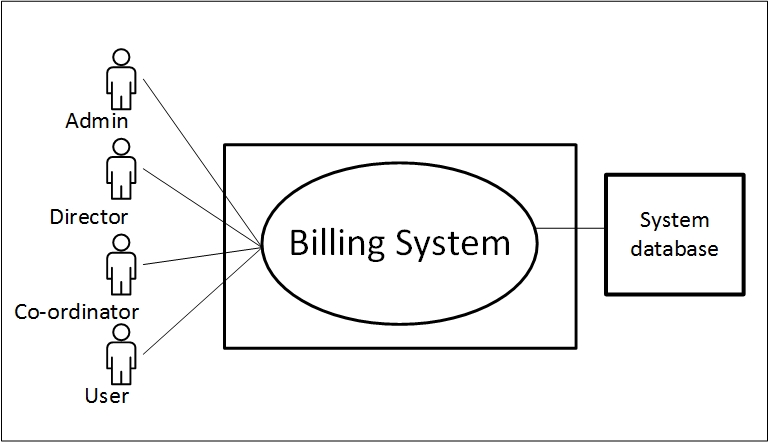
#### 5.3.1.1 Level-0 Use Case Diagram

In this level of use case diagram describes the overall system and the actors interacting with the system. Here in our project we have only one category user dealing with the system.

Figure 1: Level-0 Use Case Diagram

### 5.3.2 System Description from Level-1 Use Case Diagram

The actors our system have to play different actions and system will reply according to these actions –

Action 1: Enters signup.

Reply 1: Please fill up the required information.

Action 2: Enters the information

Reply 2: Registration successful.

Action 3: Enters username and password.

Reply 3: Sign in successful.

Action 4: Enters **Create Bill** button

Reply 4: Create bill interface will appear

Action 5: Enters **Bill description**s.

Reply 5: Prompts for further Bill information.

Action 6: Enter Add-User button to create coordinators.

Reply 6: Coordinator added.

Action 7: Enter approved button for Bill.

Reply 7: Bills are sent for next verification

Action 8: Enters to view the available features.

Reply 8: Features appeared in the page.

Exception: No Exceptions

#### 5.3.2.1 Level-1 Use Case Diagram

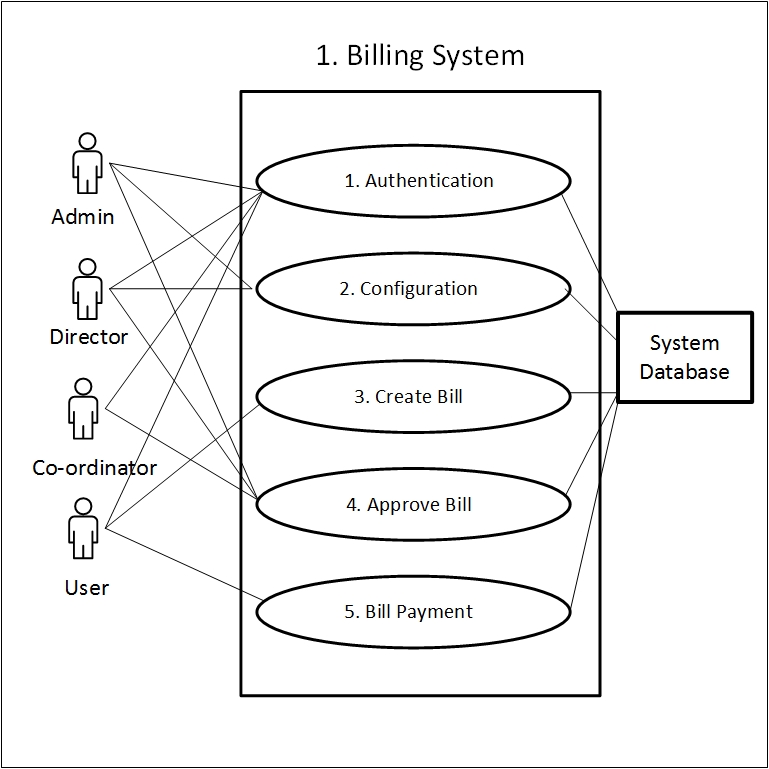
Level-1 use case where total system is divided into its subsystems which elaborately described in section 5.3.2.

Figure 2: Level-1 Use Case Diagram

### 5.3.3 Level-1.1 Use Case Diagram

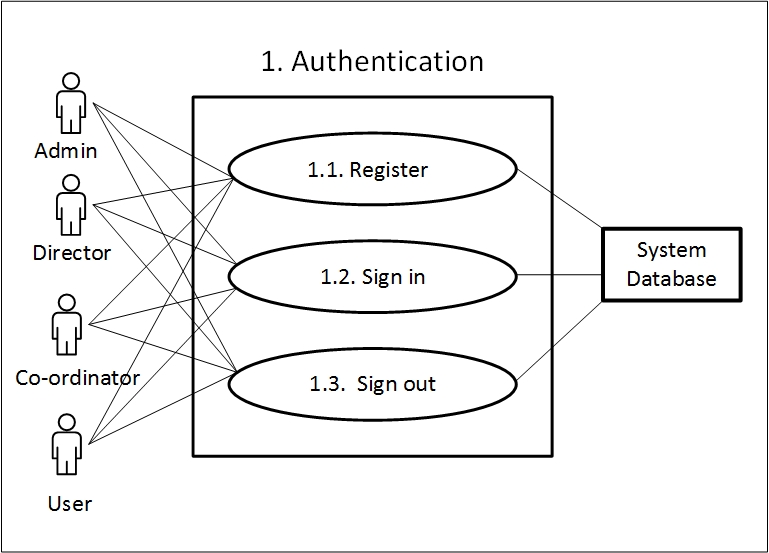
 Subsystems of subsystem 1.1 of level-1 use case diagram. System description described in the section 5.3.2.

Figure 3: Level-2.1 Use Case Diagram

### 5.3.4 System Description from Level-2.2 Use Case Diagram

Here the sub-subsystems of level-1 subsystem 2.1 are described according to users’ point of view.

Once a user clicked on **Edit Info** button following actions he/she need to conduct and the system will response on user’s corresponding action.

Action 1: Enters First Name

Enters email.

Enter password

Clicks on **Save Changes.**

Reply 1: Changes successfully saved.

Action 2: Enters **Change Password.**

Reply 2: Prompts for enter previous password.

Action 3: Enters previous password.

Reply 3: Prompts for new password.

Action 4: Enters new password.

Re-enters new password.

Enters **Save Changes.**

Reply 4: Password successfully changed.

Action 5: Enters sign out.

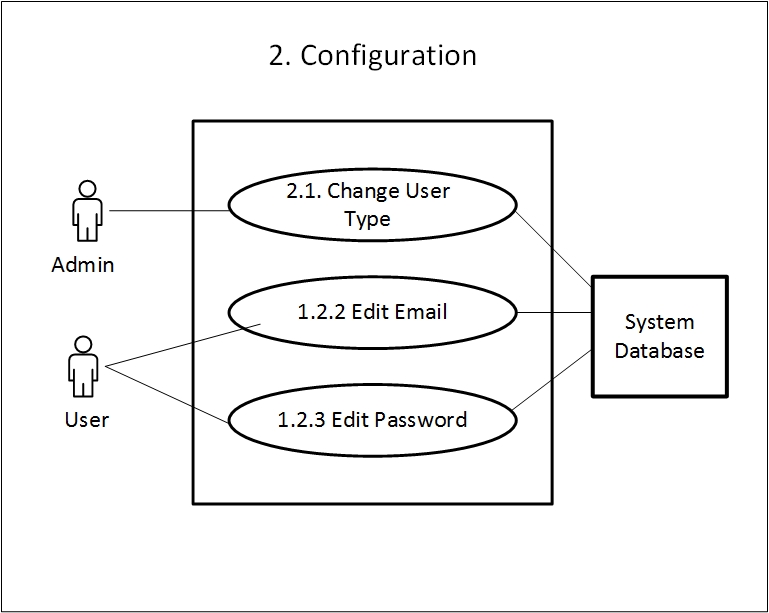
Reply 3: Logged out user.

Exception: No Exceptions

#### 5.3.4.1 Level-2.2 Use Case Diagram

The subsystem 2.2 is divided into three sub-subsystems which are shown in following level-1.2 Use Case Diagram.

Figure 4: Level-2.2 Use Case Diagram



### 5.3.5 System Description from Level-2.3 Use Case Diagram

Here the sub-subsystems of level-1 subsystem 1.3 are described.

Once a user clicked on **Create Bill** button the user will be able to give a name of that group and complete the group creation. Following are actions he/she need to conduct for further group creation process and the system will response on user’s corresponding action.

Action 1: Enters **Create Bill.**

Reply 1: Prompt for Bill user.

Action 1: Enter Bill name.

Reply 2:

Action 3: Bill name is saved

Reply 3: Appearance of the user profile.

Action 4: Enter bill category**.**

Reply 4: Bill category added

Action 5: Enter bill amount

Reply 5: Enter Bill amount;

Exception: Actor is not the group admin.

Applicable when: Actor is a normal user.

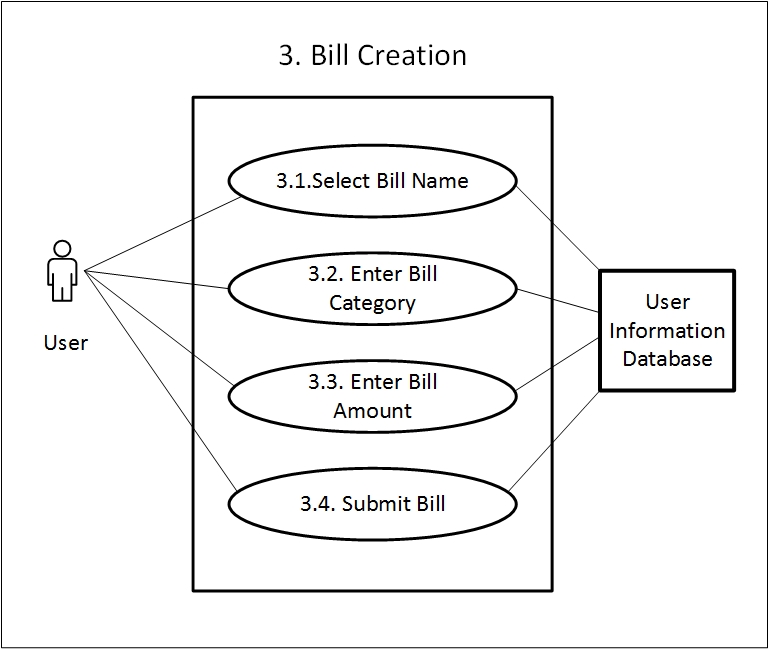
Action 6: Click create link button

Reply 7: Bill is created

#### 5.3.5.1 Level-2.3 Use Case Diagram

The subsystem 1.3 Make Group is divided into four sub-subsystems which are shown in following level-1.3 Use Case Diagram.

Figure 5: Level-1.3 Use Case Diagram



### 5.3.6 System Description from Level-2.4 Use Case Diagram

Once a user clicked on **bill create** button following are actions are needed to conduct for further post submission process and the system will response on user’s corresponding action.

Action 1: Enters **Bill create** Button

Reply 1: Take to the create bill page

Action 2: Enters Bill name, Category and Bill amount and clicks on **submit.**

Reply 2: Bill is created

Action 3: Request for approval from Admins

Reply. “Action 4: Clicks on **Submit.**

Reply 4: Edited post successfully submitted.

Action 5: Clicks on **Delete Post.**

Reply 5: Post deletion successful.

Exception: No Exceptions

#### 5.3.6.1 Level-2.4 Use Case Diagram

The subsystem 1.4 Posting is divided into four sub-subsystems which are shown in following level-1.4 Use Case Diagram.

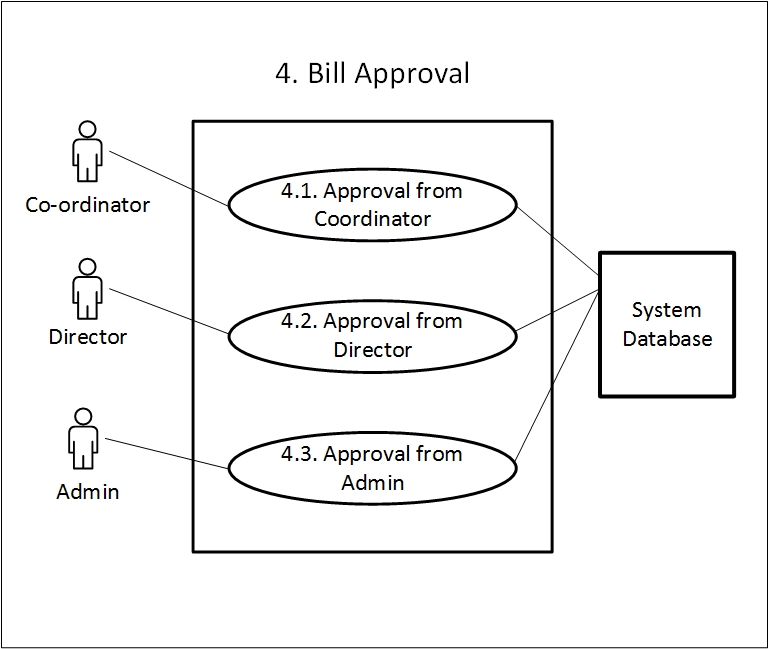


Figure 6: Level-2.4 Use Case Diagram

### 5.3.7 System Description from Level-3 Use Case Diagram

The Data Analysis subsystem is divided into two major sub systems. Following are the actions and replies regarding this system.

Action 1: Enters **Features** Button.

Reply 1: Appearance of all features.

Action 2: Clicks on **Set Goal.**

Reply 2: Prompt for goal description.

Action 3: Enters **Done**.

Reply 3: Goal is set.

Action 4: Clicks on **get location.**

Reply 4: Location will be showed.

Action 5: Clicks on **Get graph.**

Reply 5: Graph will be generated.

Action 6: Click on **Get Speed**.

Reply 7: Speed will be generated.

Exception: No Exceptions

#### 5.3.7.1 Level-3 Use Case Diagram

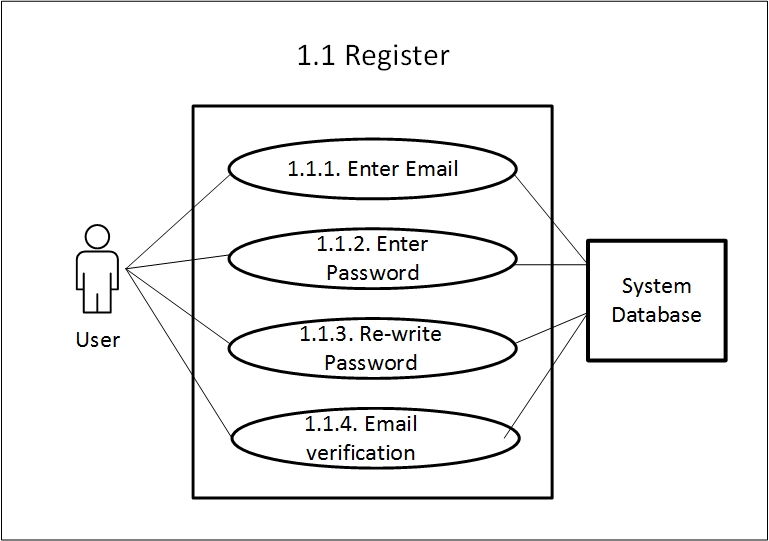
The subsystem Data Analysis is divided into two sub-subsystems. The system description of this subsystem are described in previous section.

Figure 7: Level-3 Use Case Diagram

## 5.4 Activity & Swim Lane Diagrams

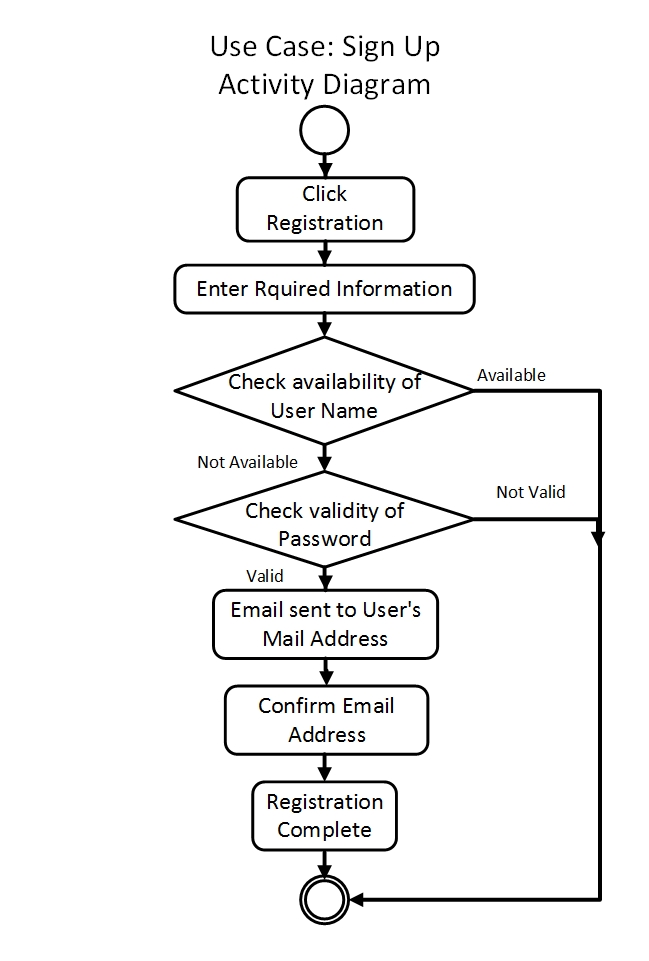
**Activity diagram** shows the technical view of the system for every use case  
from which we can understand how the system actually works and how the actors interact with the system. Here is the activity diagram for our Sign Up use case.

Figure 8: Activity for Sign Up

**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.

Following is the swim lane of use case Sign Up.

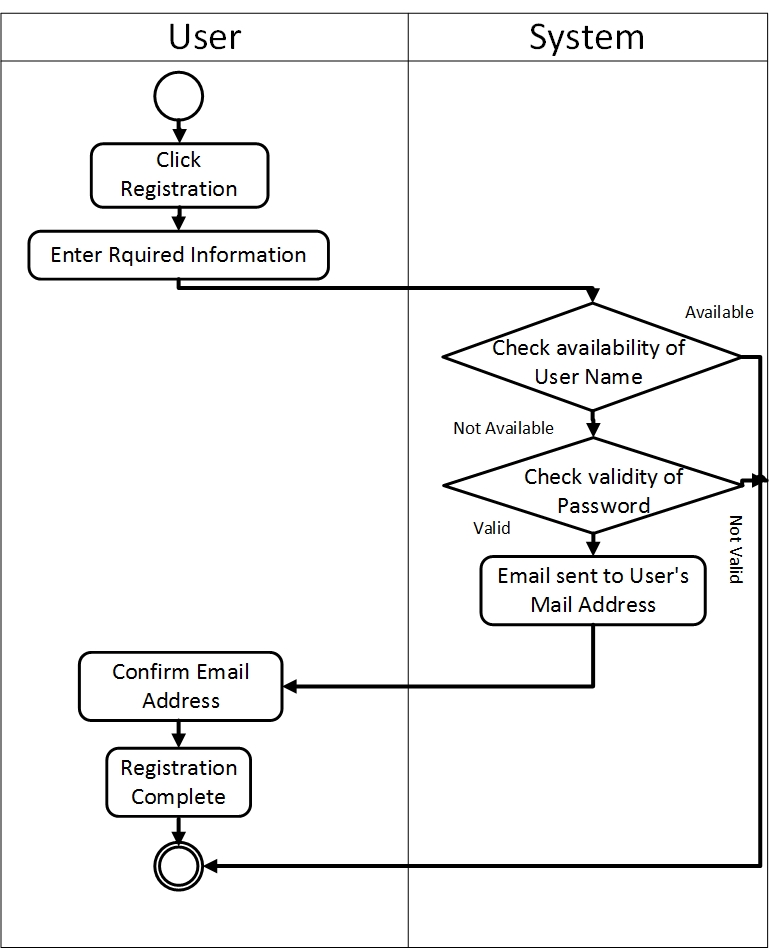


Figure 9: Swim lane for Sign Up

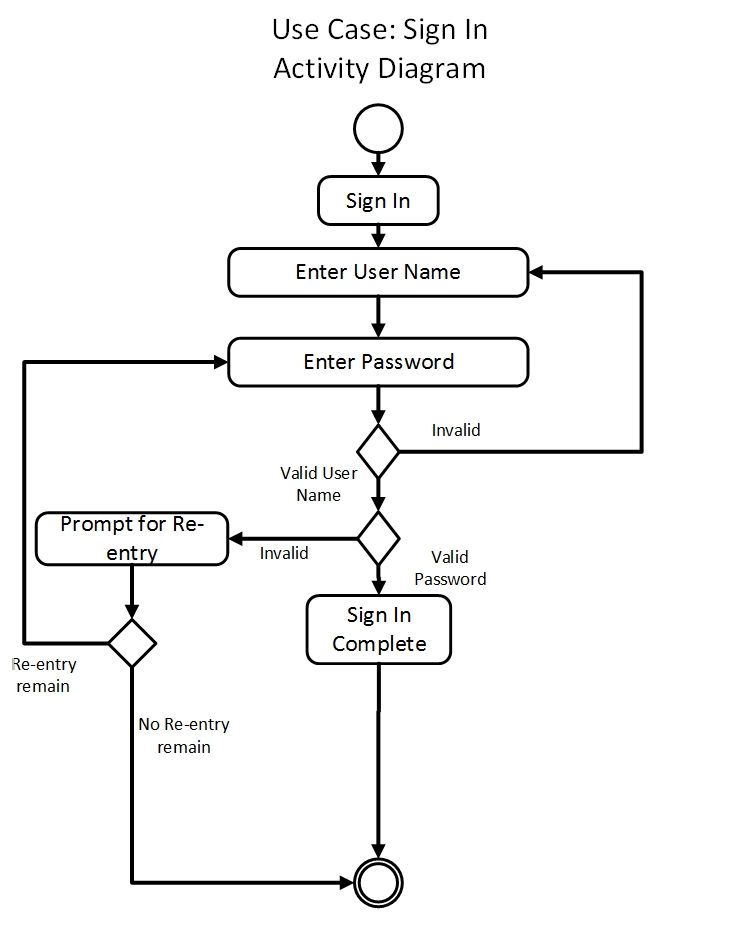
Following is the activity diagram of use case Sign In.

Figure 10: Activity for Sign In

Following is the swim lane diagram of use case Sign In.

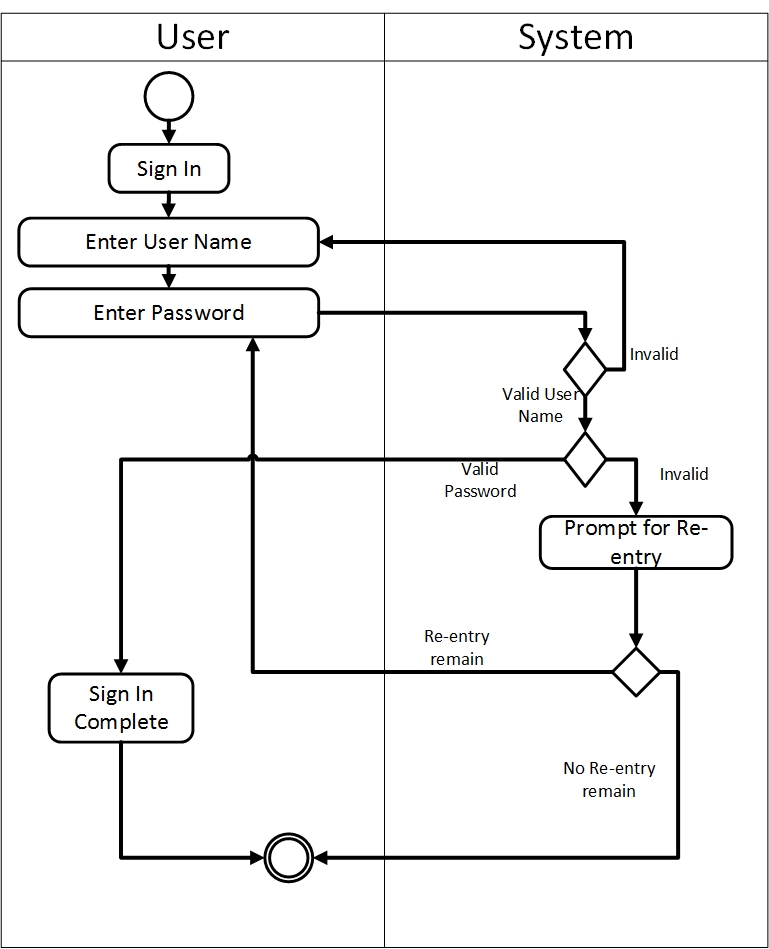


Figure 11: Swim lane diagram for Sign In

Following is the activity diagram of use case Sign out.

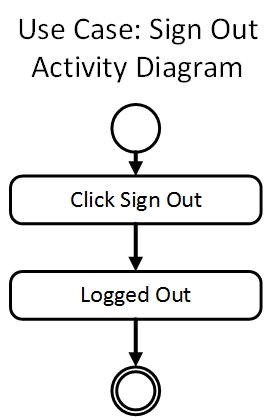


Figure 12: Activity for Sign Out

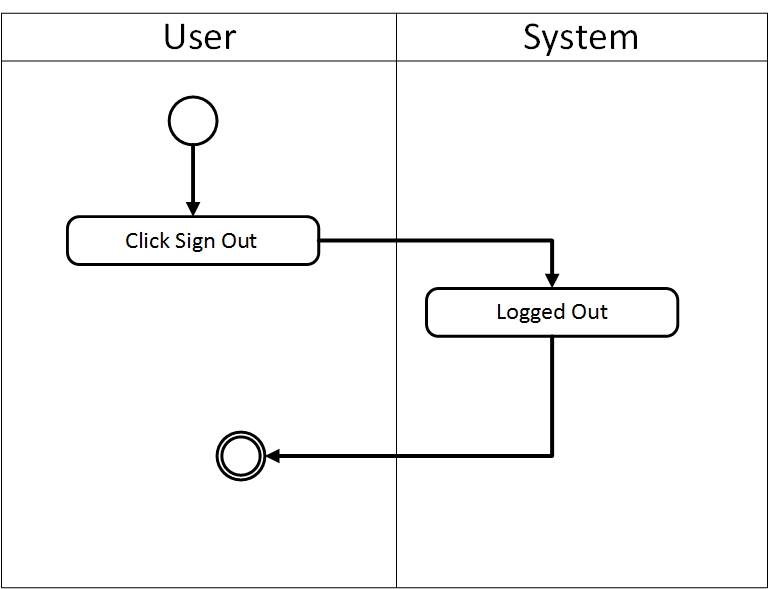
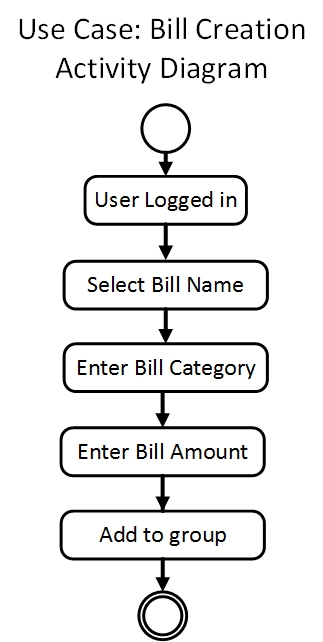
Following is the swim lane diagram of use case Sign out

Figure 13: Swim lane for Sign Out

Following is the activity diagram of use case Bill Creation.

Figure 14: Activity for Bill Creation



Following is the swim lane of use case Edit Info.

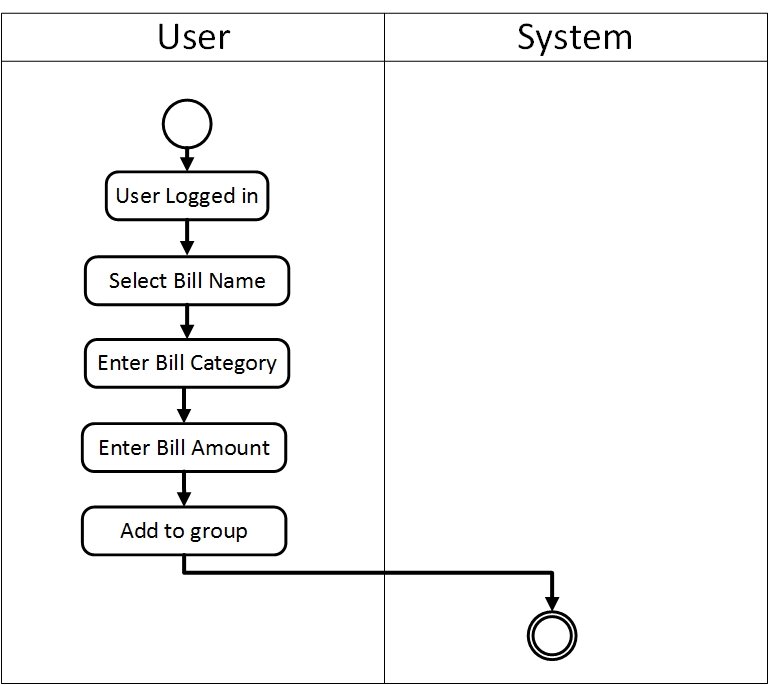
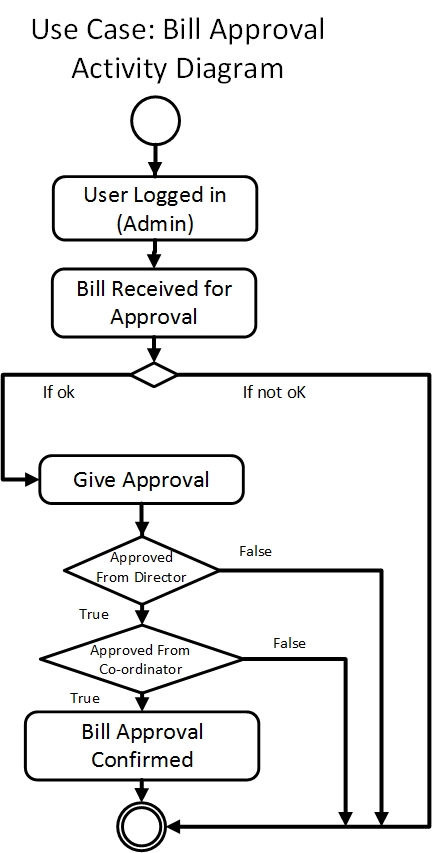


Figure 15: Swim lane for Bill Creation

Following is the activity diagram of use case Bill Approval.

Figure 16: Activity diagram for Bill Approval

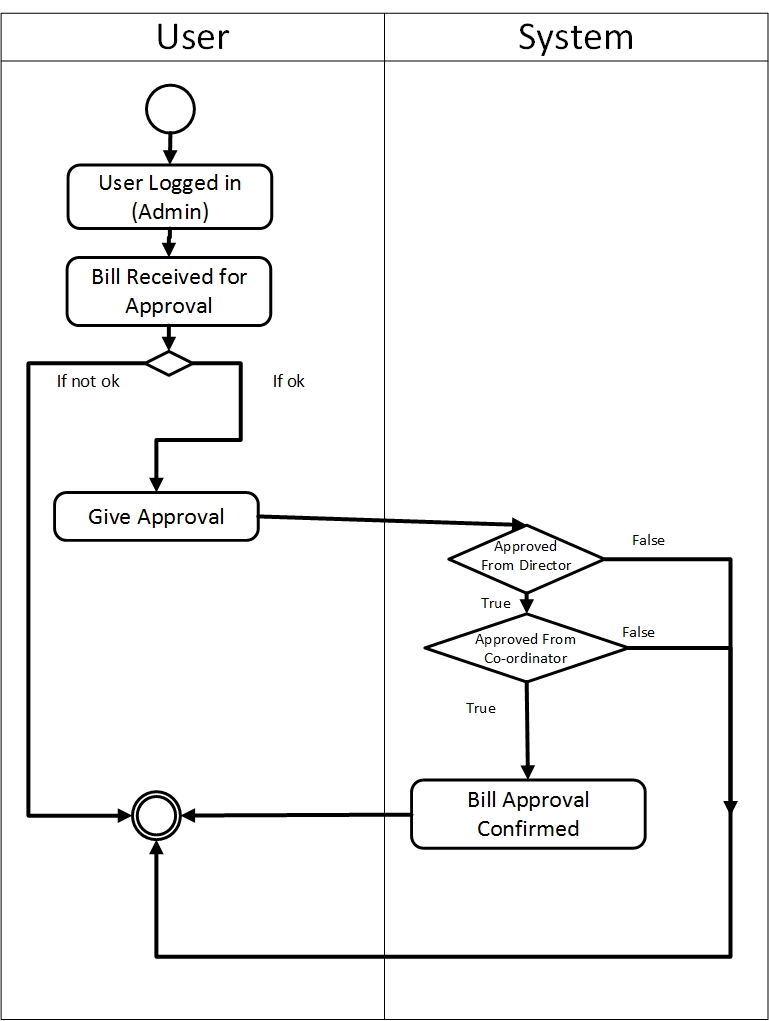
Following is the swim lane of use case Bill Approval.

Figure 17: Swim lane for Bill Approval

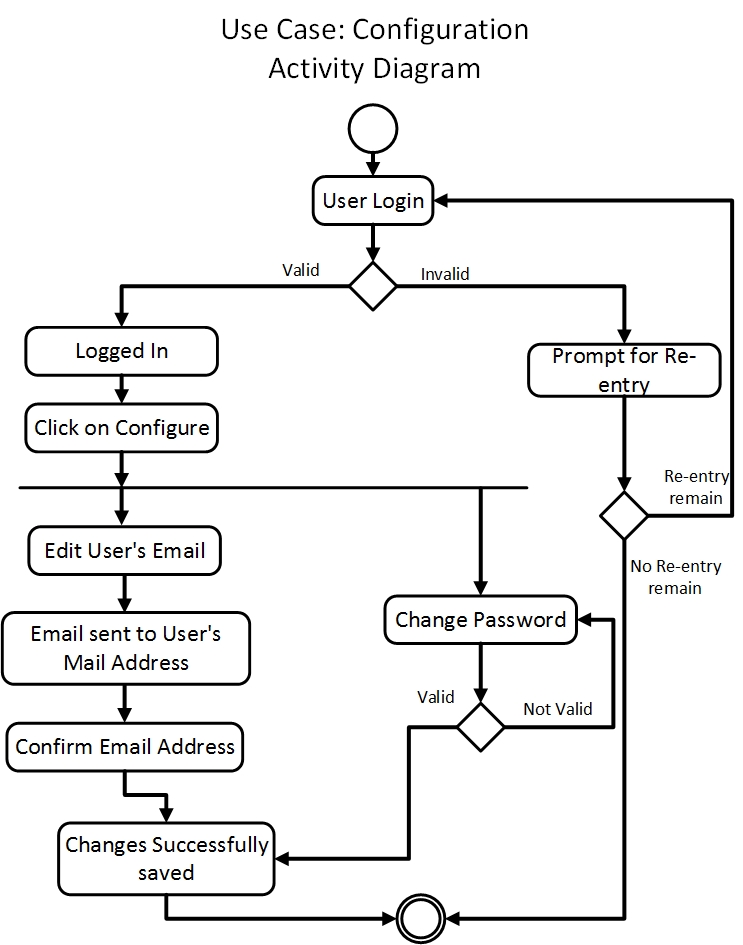
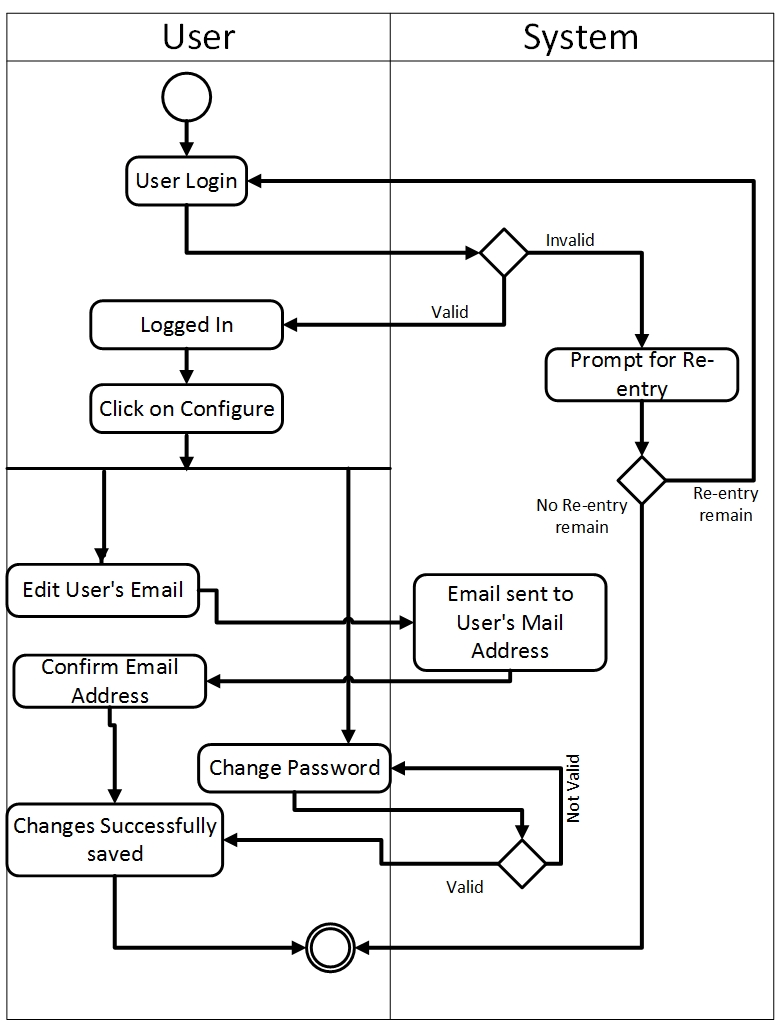
Following is the activity diagram of use case Configuration.

Figure 18: Activity diagram for Configuration

Following is the swim lane diagram of use case Configuration.

Figure 19: Swim lane diagram for Configuration

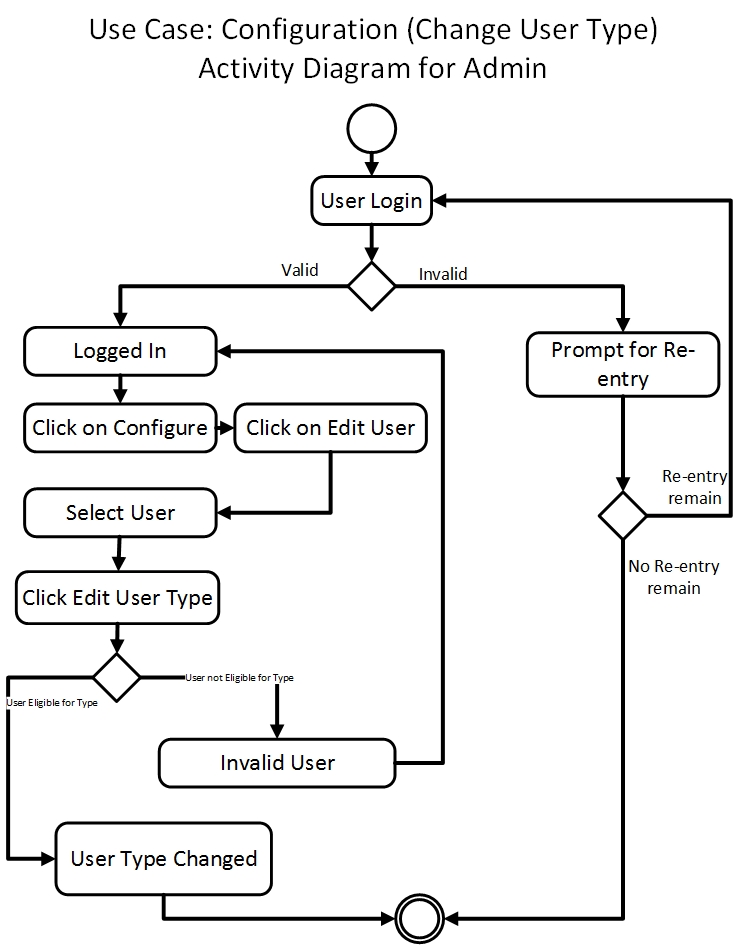
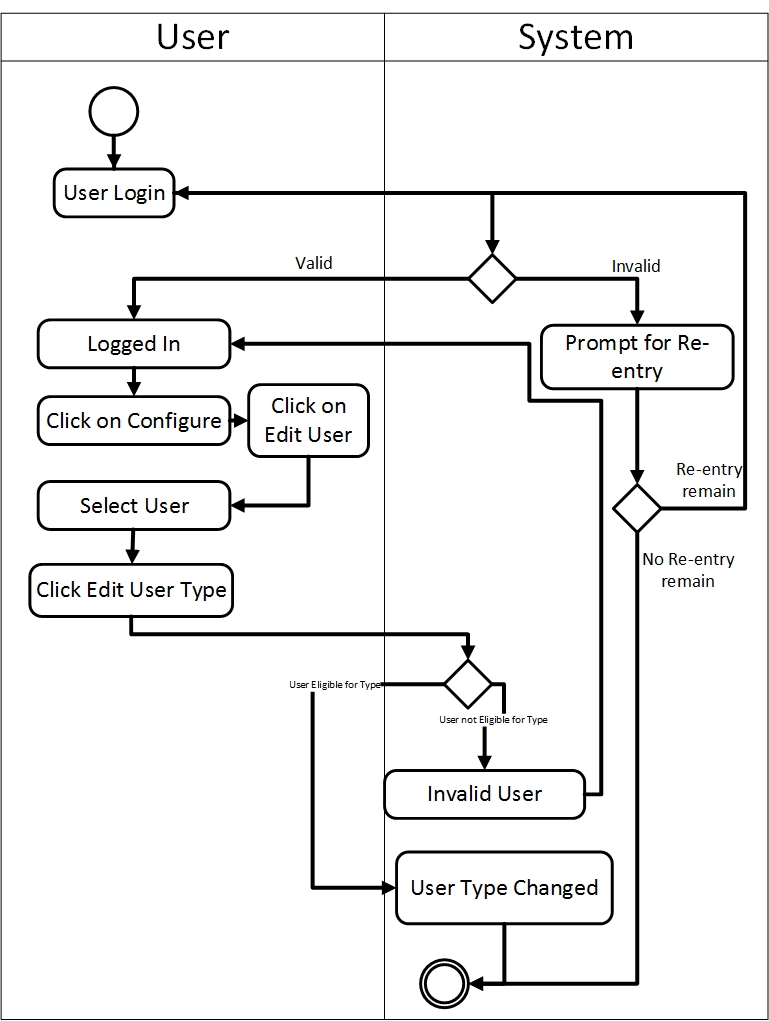
Following is the activity diagram of use case Configuration (change user type)

Figure 20: Activity diagram for Configuration (for admin)

Following is the swim lane diagram of use case Configuration (change user type).

Figure 21: Swim lane diagram for Profile Post

## 5.5 Conclusion

This type of scenario based modeling helps to improve the idea about our project from user perspective. Each diagram will help the user to understand the system more closely. They can define from these diagrams whether the product will meet their need or fulfill their usability.

# Chapter 6: Data Model

In this chapter we will discuss about the data models of our system.

## 6.1 Data Modeling Concept

If software requirements include the necessity to create, extend or interact with a data base or complex data structures need be constructed and manipulated, then the software team choose to create data model as part of overall requirements modeling. The entity-relationship diagram (ERD) defines all data objects that are processed within the system, the relationships between the data objects and the information that how the data objects are entered, stored, transformed and produced within the system.

## 6.2 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.

### 6.2.1 Grammatical parsing & analysis

We identified all the nouns whether they are in problem space or in solution space from our usage scenario and categorized them according to their attributes.

| Nouns | Problem space/solution space | Attributes |
| --- | --- | --- |
| 1. Billing System | P |  |
| 1. Admin | S | 22, 27, 28, 29, 30 |
| 1. Director | S | 22, 27, 28, 29, 30 |
| 1. Coordinator | S | 22, 27, 28, 29, 30 |
| 1. Supreme Authority | P |  |
| 1. Category | S | 23, 24, 26 |
| 1. Document of Bill | S | 22, 23, 24, 25, 26 |
| 1. Validation | P |  |
| 1. User | S | 22, 27, 28, 29, 30 |
| 1. Approval | P |  |
| 1. Admission | P |  |
| 1. Examination | P |  |
| 1. Extra Curriculum Activities | P |  |
| 1. Picnic | P |  |
| 1. Script Checking | P |  |
| 1. Overtime | P |  |
| 1. Food | P |  |
| 1. Training Program | P |  |
| 1. Regular Course | P |  |
| 1. Short Course | P |  |
| 1. Maintenance | P |  |
| 1. User Id | S |  |
| 1. Bill Id | S |  |
| 1. Bill Name | S |  |
| 1. Bill Amount | S |  |
| 1. Category Id | S |  |
| 1. Password | S |  |
| 1. Email Id | S |  |
| 1. User Type | S |  |
| 1. User Name | S |  |

### 6.2.2 Identify Data Objects

Nouns having attributes are selected as data object. Those who doesn’t have any attributes have covered under the data objects.

**Data Object:** Admin

**Attributes:**

* User Id
* Email
* Password
* User Name
* User Type

**Data Object:** Director

**Attributes:**

* User Id
* Email
* Password
* User Name
* User Type

**Data Object:** Coordinator

**Attributes:**

* User Id
* Email
* Password
* User Name
* User Type

**Data Object**: User

**Attributes:**

* User Id
* Email
* Password
* User Name

**Data Object:** Document of Bill (Bill)

**Attributes:**

* Bill Id
* Bill Name
* Bill Amount
* Category Id
* User Id

**Data Object:** Category

**Attributes:**

* Bill Id
* Bill Name
* Category Id

Here, we found that Director, Coordinator and Admin have same attributes. So we can consider a new data object named “Admin Panel” which will hold the attributes of these data objects.

**Data Object**: Admin Panel

**Attributes:**

* User Id
* Email
* Password
* User Name
* User Type

### 6.2.3 Data Object Relation

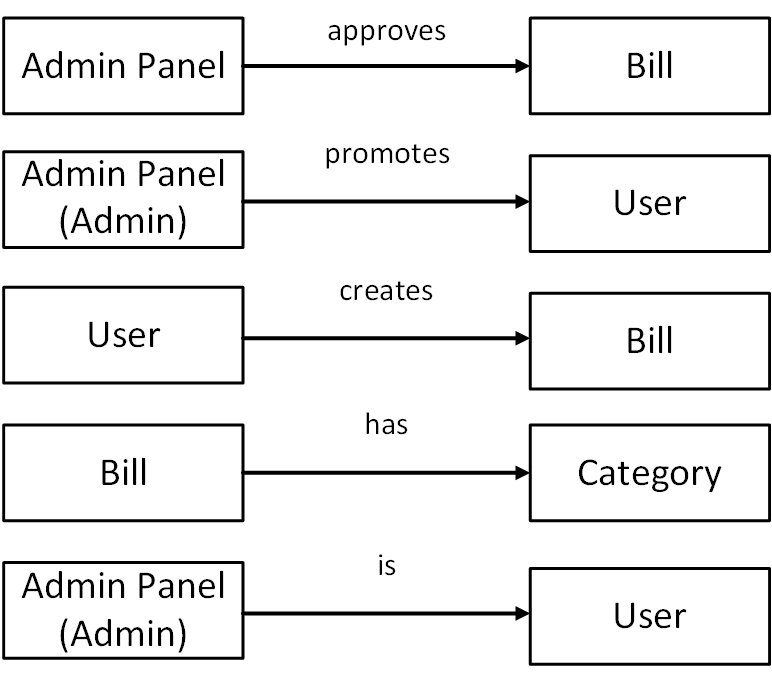


Figure 22: Data Object Relation Diagram

### 6.2.4 E-R Diagram

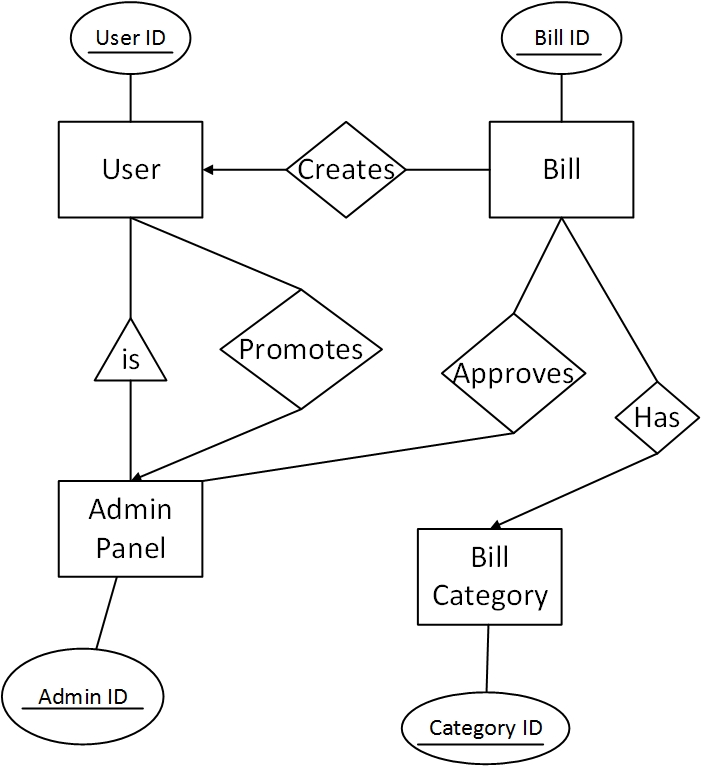


Figure 23: E-R Diagram

### 6.3.5 Schema Form (Tables)

Figure 24: Data Schema

## 6.3 Conclusion

Data modeling serves the purpose of the technical view of the system. It is helpful for the developers to design the database. This modeling gives us the idea how data will be manipulated, how to store an information of our system in the database or how to retrieve information from the database.

# Chapter 7: Class Based Model

This Chapter is intended to describe class based modeling of Billing System Website.

## 7.1 Class Based Modeling Concept

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

## 7.2 Identifying Analysis Class

To identify our analysis class, we firstly grammatically parsed all the nouns and then categorized them according to general classification and selection criteria.

Following are the steps we used to analysis the classes for our system.

|  |  |
| --- | --- |
| External Entities | User |
| Things | Password, User Name, Email, Bill Name, Category, Bill Amount, User Type, Bill |
| Occurrence or events | - |
| Roles | Admin, Director, Coordinator |
| Organizational units | - |
| Places | - |
| Structure | - |

**Step-1:** Grammatical parsing (noun identifying) and categorizing using general classification:

After categorizing noun using general classification, we follow the next step to identify preliminary classes. In this step, we find the noun which fulfill at least three selection criteria or is an essential requirement.

**Step-2:** Selection Criteria:

1. Retained information

2. Needed services

3. Multiple attributes

4. Common attributes

5. Common operations

6. Essential requirements

|  |  |  |
| --- | --- | --- |
| **Potential Class** | **Characteristic Number That Applies** | **Remarks** |
| User | 2,3,6 | Accepted |
| Admin | 2,3,6 | Accepted |
| Director | 2,3,6 | Accepted |
| Coordinator | 2,3,6 | Accepted |
| Category | 1,3 | Accepted |
| Password | 1 | Rejected |
| User Name | 1 | Rejected |
| Bill Name | 1 | Rejected |
| Bill Amount | 1 | Rejected |
| Email | 1,2 | Rejected |
| User Type | 1 | Rejected |
| Bill | 1,3,6 | Accepted |

Nouns that hold essential requirement we took them essential for our system and include them in our analysis classes.

**Preliminary Classes:**

1. User
2. Director
3. Coordinator
4. Admin
5. Bill

**Attributes Selection:**

So, the retrieved attributes are listed below with the preliminary classes.

|  |  |
| --- | --- |
| **Class Name** | **Attributes** |
| User | User Name  User Id  Password  Email |
| Director | User Name  Password  User Id  User Type |
| Admin | User Name  Password  User Id  User Type |
| Coordinator | User Name  Password  User Id  User Type |
| Bill | Bill Name  Bill Id  Bill Amount |

**Step-3: Method Identification:**

To identify methods, we will find out the verbs of the user story and will identify which one is in scope of the problem and which one is out of scope.

**Verb Detection:**

|  |  |  |
| --- | --- | --- |
| No | Verb | Remark |
|  | Promote | Yes |
|  | Demote | Yes |
|  | Sign Up | Yes |
|  | Sign In | Yes |
|  | Sign Out | Yes |
|  | Provide | Out of Scope |
|  | Select | Out of Scope |
|  | Approve | Yes |
|  | Enter | Out of Scope |
|  | Pdf Generate | Yes |
|  | Notify | Yes |
|  | Bill Create | Yes |
|  | Reject | Yes |
|  | Add | Yes |
|  | Reject | Yes |
|  | Send | Out of Scope |
|  | Check | Out of Scope |
|  | Edit | Yes |

So, the methods of the classes are listed below.

|  |  |
| --- | --- |
| **Class Name** | **Methods** |
| User | Sign Out ()  Sign In ()  Sign Up ()  Pdf Generate ()  Edit () |
| Admin | Sign Out ()  Sign In ()  Sign Up ()  Promote ()  Demote ()  Approve ()  Reject () |
| Director | Sign Out ()  Sign In ()  Sign Up ()  Approve ()  Reject () |
| Coordinator | Sign Out ()  Sign In ()  Sign Up ()  Approve ()  Reject () |
| Bill | Edit Bill ()  Delete Bill ()  Generate Bill () |

**Class Card:**

|  |  |
| --- | --- |
| **User** | |
| **Attributes** | **Methods** |
| User ID  User Name  Password  Email | Sign Out ()  Sign In ()  Sign Up ()  Pdf Generate ()  Edit () |
| **Responsibilities** | **Collaborative Class** |
| Creates bill | Bill |
| Generates pdf | Admin, Director, Coordinator |

|  |  |
| --- | --- |
| **Admin** | |
| **Attributes** | **Methods** |
| User ID  User Name  Password  Email  User Type | Sign Out ()  Sign In ()  Sign Up ()  Promote ()  Demote ()  Approve () |
| **Responsibilities** | **Collaborative Class** |
| Promotes coordinator | Coordinator, User |
| Demotes coordinator | Coordinator |
| Approves bill | Bill |
| Rejects bill | Bill |

|  |  |
| --- | --- |
| **Director** | |
| **Attributes** | **Methods** |
| User ID  User Name  Password  Email  User Type | Sign Out ()  Sign In ()  Sign Up ()  Approve ()  Reject () |
| **Responsibilities** | **Collaborative Class** |
| Approves bill | Bill |
| Rejects bill | Bill |

|  |  |
| --- | --- |
| **Coordinator** | |
| **Attributes** | **Methods** |
| User ID  User Name  Password  Email  User Type | Sign Out ()  Sign In ()  Sign Up ()  Approve ()  Reject () |
| **Responsibilities** | **Collaborative Class** |
| Approves bill | Bill |
| Rejects bill | Bill |

|  |  |
| --- | --- |
| **Bill** | |
| Attributes | Methods |
| Bill Id  Bill Name  Category | Edit Bill ()  Delete Bill ()  Generate Bill () |
| **Responsibilities** | **Collaborative Class** |
| Generates bill | User |
| Deletes bill | User |
| Edit bill | User |

After analyzing each classes we found that Admin, Coordinator, Director have similar attributes and methods. So we can consider a super class named “Admin Panel” of them which will hold common attributes and common methods. Other classes will be sub class of this super class.

**Common Attributes:** User ID, User Name, Password, Email, User Type

**Common Methods:** SignOut(), SignIn(), SignUp(), Approve(), Reject()

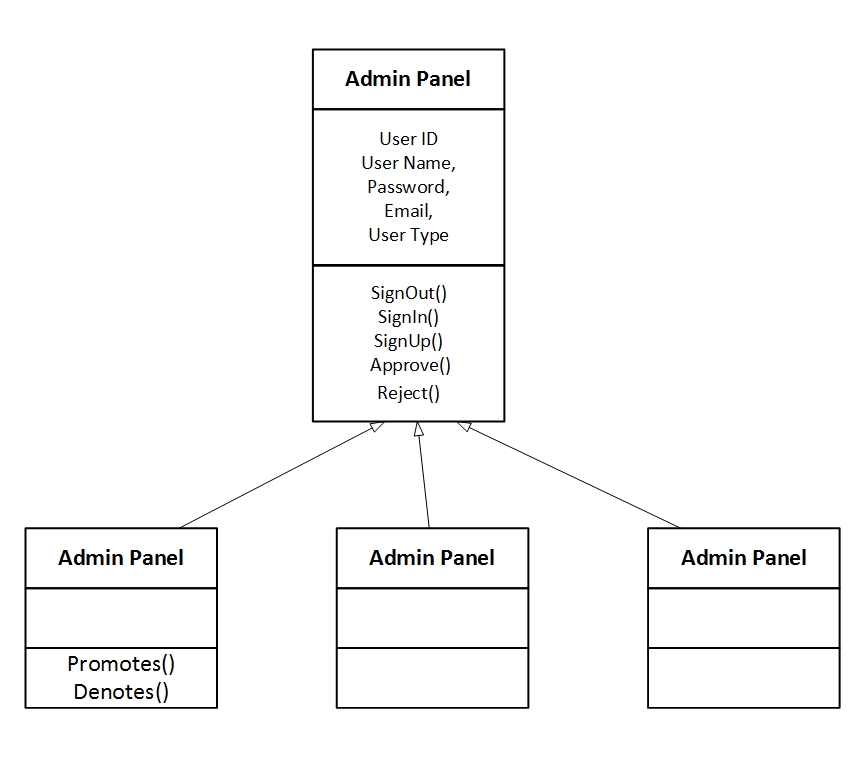


Figure 25:“Admin Panel” Super Class

### **7.2.1 Class Responsibility Collaboration (CRC)**

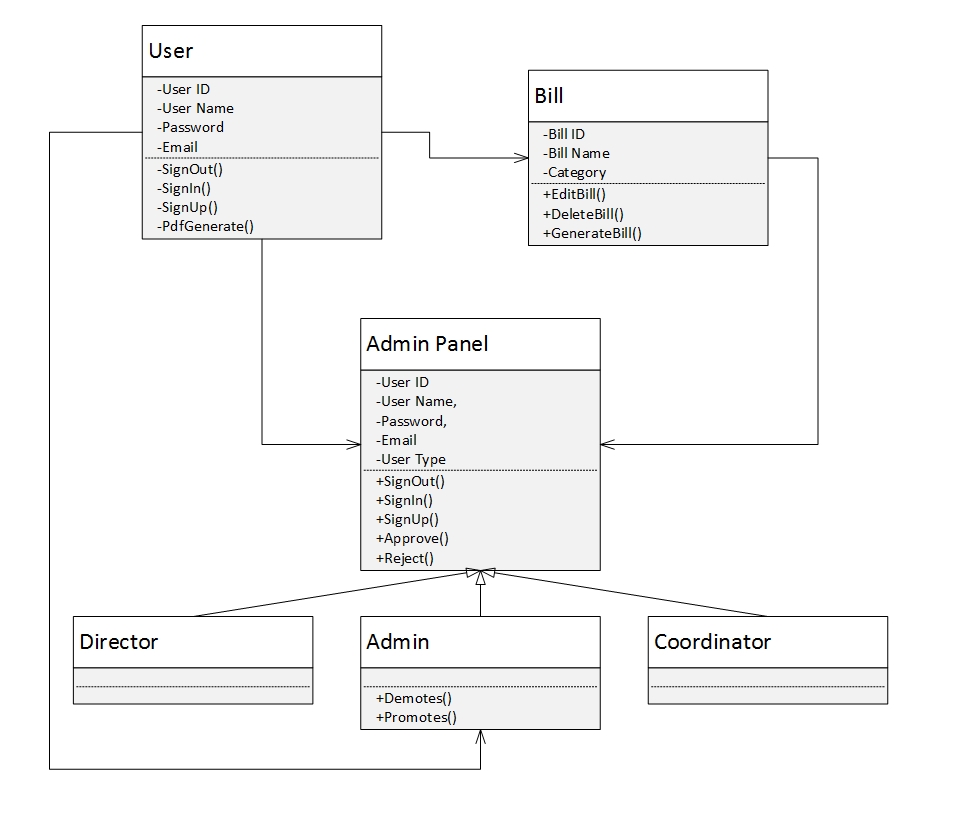


Figure 26: CRC Diagram

## 7.3 Conclusion

The elements of a class-based model include classes and objects, attributes, operations, class responsibility-collaborator (CRC) models, collaboration diagrams, and packages. This mainly represents the relationship between objects. Class based modeling normally helps a system to be developed in a way for future reusability.

# Chapter 8: Flow Oriented Model

This chapter focuses on the flow oriented modeling.

## 8.1 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. It provides additional insight into system requirements and data flow. The data flow diagram enables you to develop models of the information domain and functional domain. As the DFD is refined into greater levels of detail, you perform an implicit functional decomposition of the system.

## 8.2 Data Flow Diagram

The DFD takes an input-process-output view of a system. That is, data objects  
flow into the software, are transformed by processing elements, and resultant data objects flow out of the software. In the figures, data objects are represented by labeled arrows and transformations are represented by circles.

**Level-0 DFD:** The level 0 data flow diagram should depict the software/system as a single bubble. Here, Level-0 DFD is describing the overall system’s input and output data.

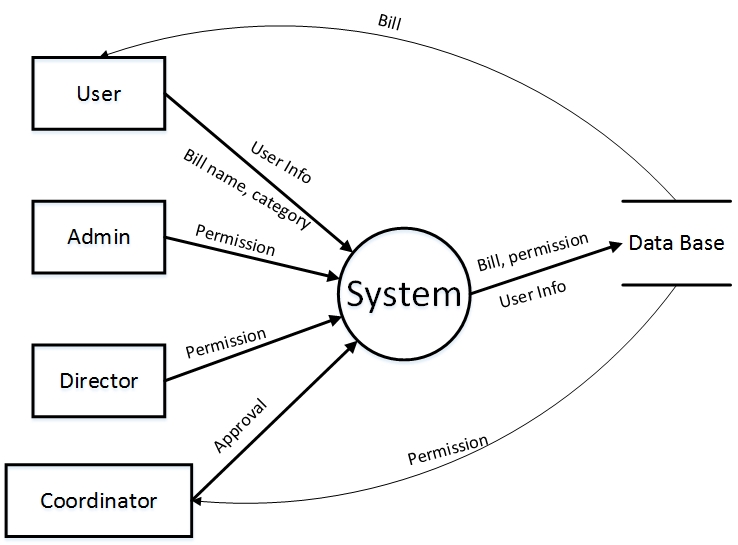


Figure 27: Level-0 DFD

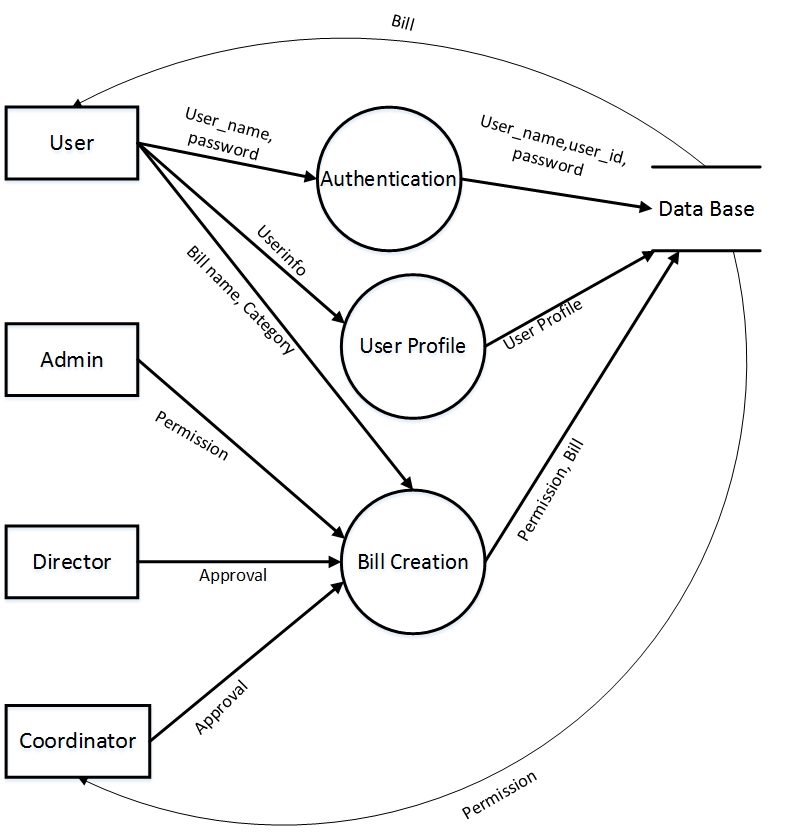
**Level-1 DFD:** At DFD level 1 can be further refined into lower levels. This level-1 DFD derived from level-0 DFD.

Figure 28: Level-1 DFD

**Level-2 DFD:** Every process of data flow can be described into many sub process. And thus we got level-2 DFD from the process Authentication in level-1 DFD.

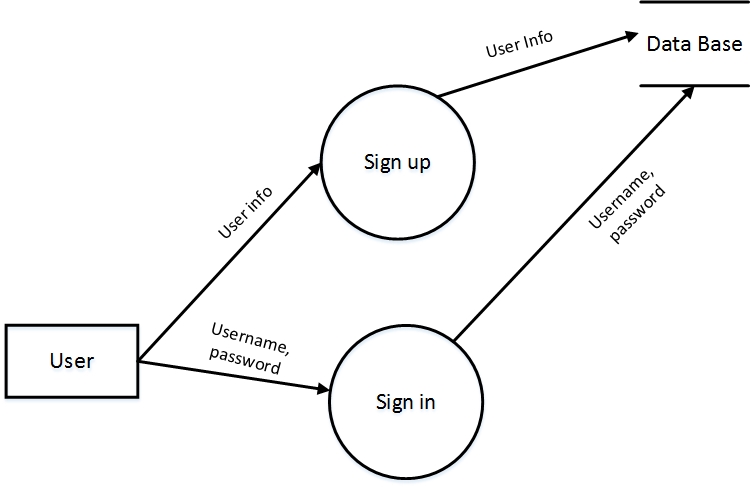


Figure 29: Level-2 DFD

## 8.3 Conclusion

The refinement of DFDs continues until each bubble performs a simple function. That is, until the process represented by the bubble performs a function that would be easily implemented as a program component.

# Chapter 9: Behavioral Model

The behavioral model indicates how software will respond to external events. Two different behavioral representations are discussed in this chapter that follow. The first indicates how an individual class changes state based on external events and the second shows the behavior of the software as a function of time.

## 9.1 State Transition Diagram

State diagram represents active states for each class the events (triggers). For this we identified all the events, their initiators and collaborators.

| Event | Initiator | Collaborations |
| --- | --- | --- |
| 1. Generating pdf | User | User, Admin Panel |
| 1. Promotes user | Admin Panel | User |
| 1. Demotes user | Admin Panel | User |
| 1. Approves bill | Admin Panel | Bill |
| 1. Rejects bill | Admin Panel | Bill |
| 1. Generating bill | Bill | User |
| 1. Editing bill | Bill | User |
| 1. Deleting bill | Bill | User |

**STD for User:** State transition diagram for User shows us actions and states for every action occurred in this class.

Figure 13: STD for User Class

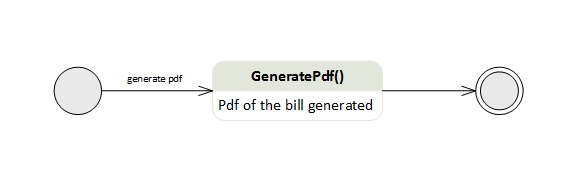
**User:**

Figure 30: STD for User Class

**STD for Bill:** State transition diagram for Bill class shows us actions and states for every action occurred in this class.

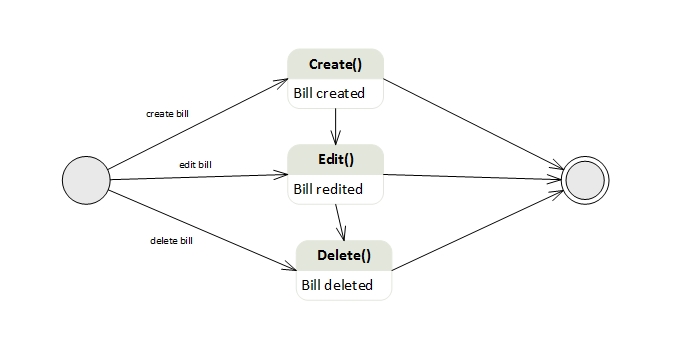
**Bill:**

Figure 31: STD for Bill Class

**STD for Admin Panel:** State transition diagram for Admin-Panel class shows us actions and states for every action occurred in this class.

**Admin Panel:**

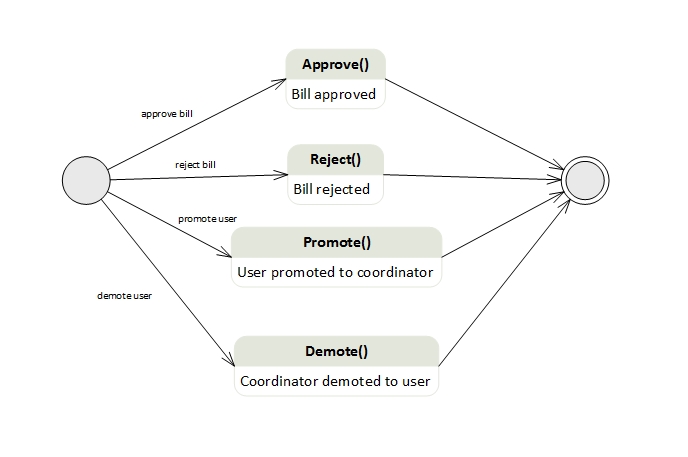


Figure 31: STD for Admin-Panel class

## 9.2 Sequence Diagram

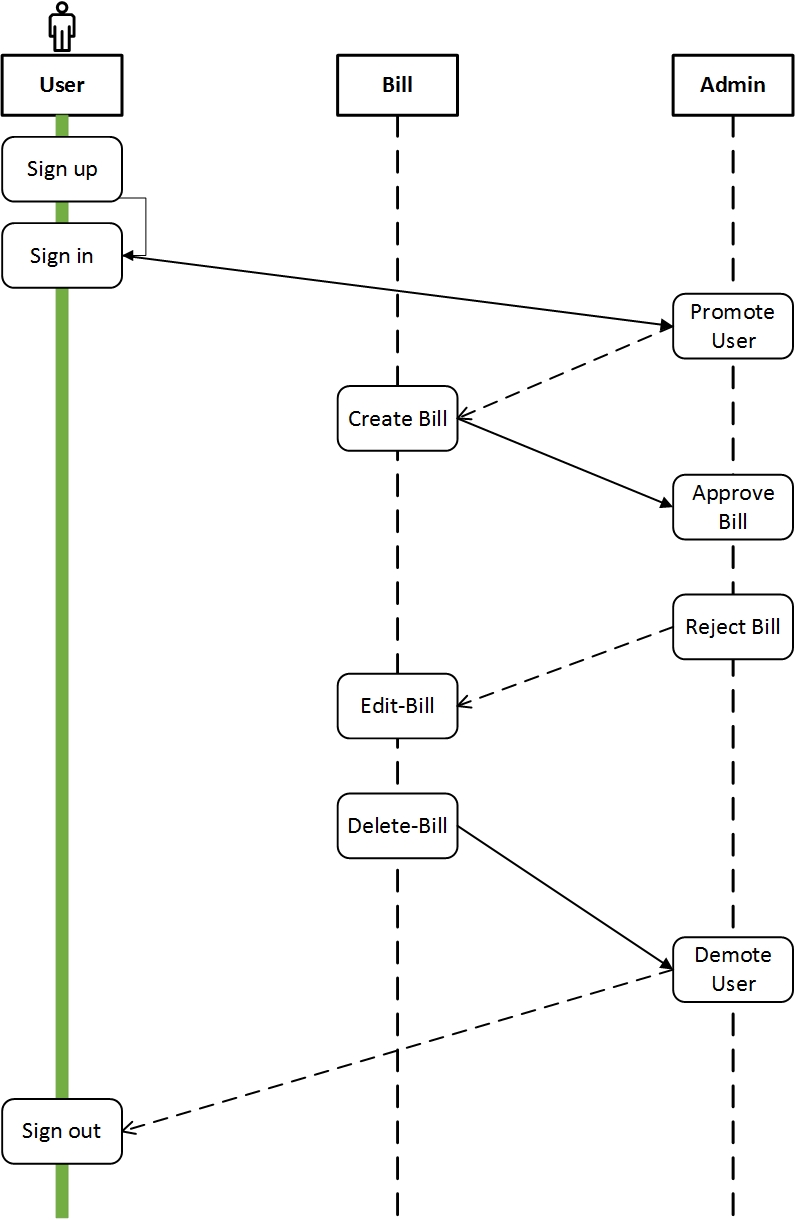
The second type of behavioral representation, called a *sequence diagram* in UML, indicates how events cause transitions from object to object. Once events have been identified by examining a use case.

Figure 32: Sequence Diagram

## 9.3 Conclusion

Although the active state model provides useful insight into  
the “life history” of an object, it is possible to specify additional information to provide more depth in understanding the behavior of an object. A sequence diagram—a representation of how events cause flow from one  
object to another as a function of time. In essence, the sequence diagram is a shorthand version of the use case. It represents key classes and the events that cause behavior to flow from class to class.

# Chapter 10: User Interface

This chapter described the user interface of our project “Billing System”.

## 10.1 Introduction

Our Billing System will definitely a user friendly application which sole intention is to give the user a complete feel of satisfaction about his/her activity. Though the development of the project is not finished yet, still we planned for the user interface for the further development of the project.

## 10.2 Basic Interface

Following is the very simple interface of Billing System which we assumed for the project development.

Figure 18: User Interface of Billing System (Home)

## 10.3 Conclusion

Figure 19: User Interface of Billing System (Set Goal)

The basic interface is only a preliminary planned interface. It may be changed after the development of the total project “Billing System”. Hopefully this preliminary interface will help the users to get a better understand about the project.

# Chapter 11: Conclusion

A software project means a lot of experience. In this section we summarize the experience gained by project team during analysis and learning phase of “Billing System”.

## 11.1 Obstacles

There were several obstacles we faced when started the project. These are:

* Working with web application and server site development are completely a new experience for us. Normally we work with different OO languages, DBMS, mark up languages etc.
* We learned new technologies for getting the work done, still we are learning for the better manageability of our project.
* We are trying to adopt these things by video tutorials, text tutorials, and internet and learning materials given by the tools themselves. It's a matter of time, patience and hard work.

In essence the thing is that, a Billing System project with the social networking facilities is not a project of 2 or 3 months for two people!

## 11.2 Achievements

* Now we know much more about the server site application, how it works, the properties, objects and others.
* We know how a data base model is constructed, and how to manage all the data with the application.
* The main thing is that as a software engineer, skill and expertise to create a SRS document and an overall software product report is now better than before.
* Co-Operation between two group members.
* Develop communication skills
* Growing creative thinking and imagination capability.

## 11.3 Future Plan

* Completing the implementation of our project.
* Introduce new features to the application.
* Launch the application for all types of users.

## 11.4 Last Few Words

We are really very glad to submit the final report on Billing System system. From this, the readers will get a clear and easy view of Billing System system. This document can be used effectively to maintain software development cycle. It will be very easy to conduct the whole project using this document. Hopefully, this document can also help our junior BSSE batch students. We tried our best to remove all dependencies and make effective and fully designed document. We believe that reader will find it in order.

There were times that we almost lost hope but we recovered and will recover through constant concentration and hard work.

If any kind of suggestion, improvements, more efficient development idea please feel free to communicate with us.

# Appendix

**References:**

* **Books**
* Pressman, Roger S. Software Engineering: A Practitioner's Approach (7th Ed.)
* **URLs**