Project Proposal on

**F*inancing* Management S*ystem***

**Suman Shahi**

*00174381*

Computing Project

Level 5 Diploma in Computing

Softwarica College of IT & E-Commerce

Kathmandu, Nepal

Submitted to

Niman Maharjan



# **Chapter 3: Design Specification**

# Introduction

After analysis specification in SDLC the next phase is Design Specification, in this phase requirements are carried out form analysis phase which are collected to evaluate essential resources for system develop. Design specifies system design, and information regarding project features to achieve goals for developers to fulfill the user requirements. Numerous activities represent structural, behavior models, sequence diagram that are included in development phase. Various problems rise in the system which are solved in these phases. Structural Model, Behavior Model, Database Model are represented by Unified Modelling Language UML.

UML is representation method, providing details of the software system beneath design. UML don’t provide any command keywords or syntax therefore it is a non-programming language. It is a mixture of OOP notation with web application which models real world objects.

For the project I have used UML and Visual paradigm software which makes easier with more function to use.

# 3.1 Structural Design

Structural Design represents class diagrams and object diagrams. It provides solution to the conceptualization of problem with better understanding for solving the problem. It helps to concentrate on the accurate problem for designer by distinguishing the structure of the objects. It emphasizes the structure with classifiers, attributes, methods, components, inter-relationships and operation of the system. It views the static structure of the system through class diagram:

## 3.1.1 Class Diagram

Class diagram refers Analysis and Design of the static view of an application that describes the responsibilities of a system. Its base for component and deployment diagrams. Class diagram is provided using Star UML.

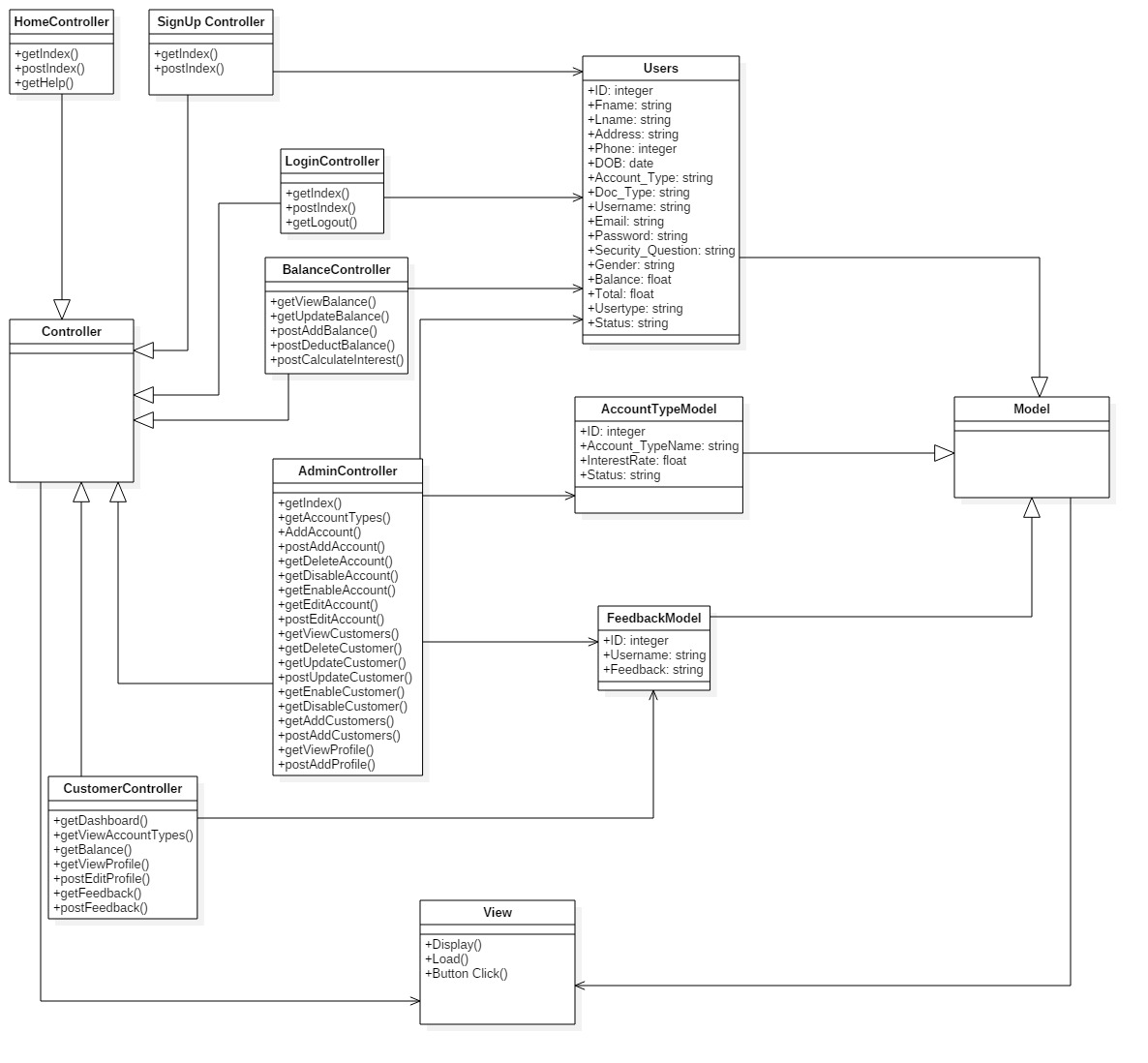


Fig: Final Class Diagram

MVC Design pattern is applied on the above class diagram. Controller classes are extended from controllers like Admin Controller, Customer Controller, Balance Controller, Login Controller, Sign UP controller and Home Controller. Admin is main controller where user is focused on customer controller for the view of the website. Signup Controller is used by customer for new registration account. Login Controller provide access to the system by login details. Balance Controller is both can be used and view by admin and customer where admin is able to see and change the balance of customer and customer are able to check their own balance. Admin controller controls the balance and customer where admin can enable account of customer add balance, deduct balance and calculate the interest rate of customer account saving. The Controller acts as an intermediary between view and model in class diagram.

## 3.1.2 DFD

It is used to graphically represent the flow of data in information system. System process to transfer data from the input to data storage and generate reports. Visual representation makes good communication tool between User and System Design.

DFD diagram is drawn on visual paradigm with 3 levels of data flow diagrams for customer and admin.

**DFD symbols**

There are four basic symbols that are used to represent a data flow diagram.

|  |  |  |  |
| --- | --- | --- | --- |
| **SN** | **Name of Notation** | **Notation** | **Description** |
| 1 | Entity |  | It initializes the object outside the system for communication. It provides the destination of the system input and output. |
| 2 | Process |  | It provides a process of incoming data flow into outgoing data flow. |
| 3 | Data Store |  | Datastores are repositories of data in the system. |
| 4 | Data Flow |  | it is the pipelines that shows the direction of information flow. |

## Level 0

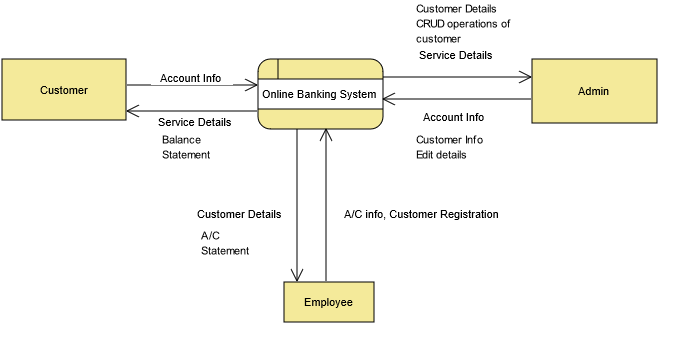


Fig: Level 0 DFD diagram

Above diagram presents that Customers are the external entities for using online banking system process and admin and employee are also external entities. It shows the data flow of the customer, admin and employees to the process.

## **Level 1**

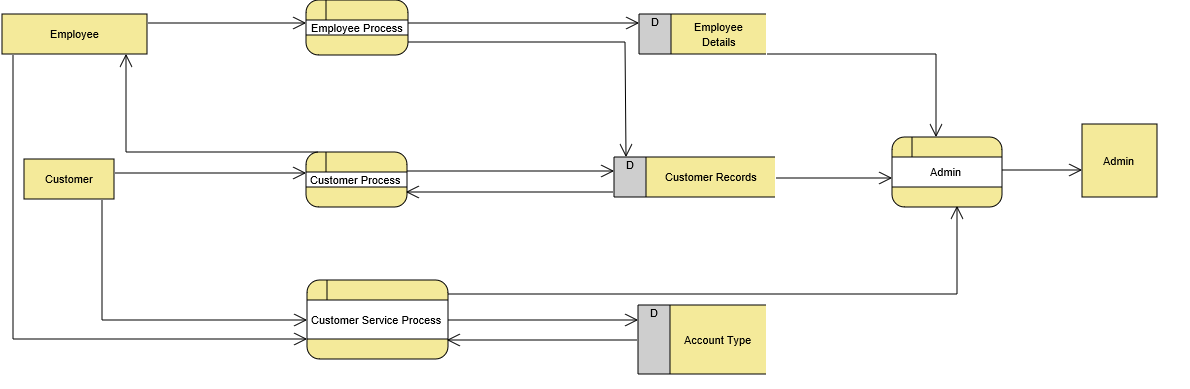


Fig: Level 1 DFD diagram

Above diagram represents the level 1 Data flow of the admin, customer and employee. Customer register the account by login in the system by providing the customer records and customer are provided with the account type to register and service process is provided. Employee are provided for employee process and customer process employee add customer to the customer records and admin enables the account of the customer.

## **Level 2 Admin**

## 

Fig: Level 2 Admin DFD diagram

In level 2 admin first login in the system and controls the process by providing service to the customer, provide account type to the new customer, visualize the customer transaction and statement of customer. Here admin enables the customer account and adding balance, deducting balance and add interest to customer account with many operation admin can delete the customer and disable the account also.

## **Level 2 Customer**

## 

Fig: Level 2 Customer DFD diagram

In level 2 customer first login in the system and they can access to their account for their account statements. Customer can view services provided by the admin and edit their profile. Customer get the preview of their account details like money added, money deducted and money transaction.

# 3.2 Behavioral Model

Behavioral models are models of the dynamic behavior of the system as it is executing. They show what happens or what is supposed to happen when a system responds to a stimulus from its environment. It’s a conceptual framework that defines control structure of operations. It helps to deal with requirement engineering and state model of software systems.

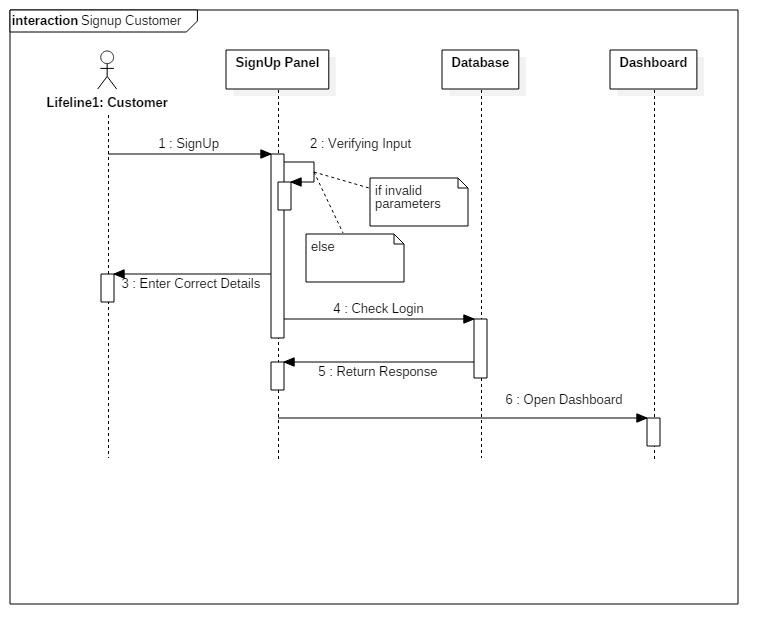
## **3.2.1 Sequence Diagram**

Sequence diagram is pictorial representation of how classes operation with one another and interchange the message between objects of the system based on lifelines. Sequence diagram consists of high-level abstraction. It helps to visualize and predict how a program would run with various runtime operations. Sequence diagram is also known as Event diagrams.

Notation like actors, message, lifeline, execution is used for constructing sequence diagrams.

**Notations:**

|  |  |  |  |
| --- | --- | --- | --- |
| **SN** | **Notation Name** | **Notation** | **Description** |
| 1 | Actor |  | It interacts with internal and external of system |
| 2 | Life Line |  | It show the time schedule that take place |
| 3 | Option |  | Model used for if/then scenario |
| 4 | Alternative |  | Used for choosing the message between sequences of message |
| 5 | Message |  | For sending and replying the message |
| 6 | Recursive Message |  | Sending message to itself |
| 7 | Object |  | Presenting the class or object |

 Fig: Sequence Diagram for Sign UP

In above diagram class and object are involved in the system. Customer needs to register their detail for the new account. Once they are registered in the system they are required and able to login in the system. With provided details customer need to input and if the parameter is invalid, they are not able to login in the system, hence they need correct input with correct parameter. If the login is successful, they are transferred to the dashboard.

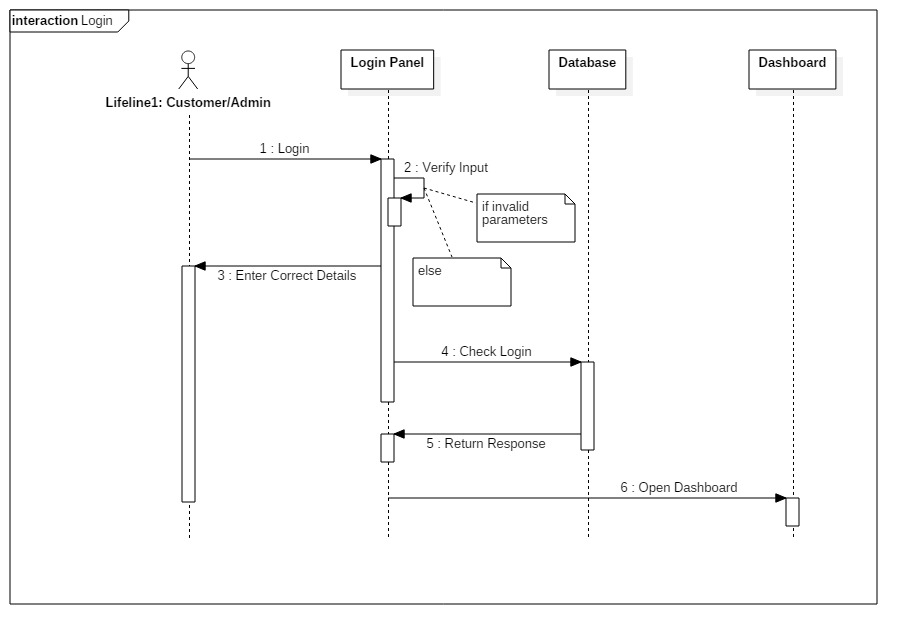


Fig: Sequence Diagram for Login

In above diagram both the customer and admin need to login in the system. Once the customer provide detail to login in the system, details are checked in the database if its correct or not. If the data is correct customer are able to login in their system and dashboard is open. Here, admin also required login for the system. Only one admin is able to run the system. Admin also required detail data for login, with correct parameter on the data base admin login in the system and the dashboard is popup.

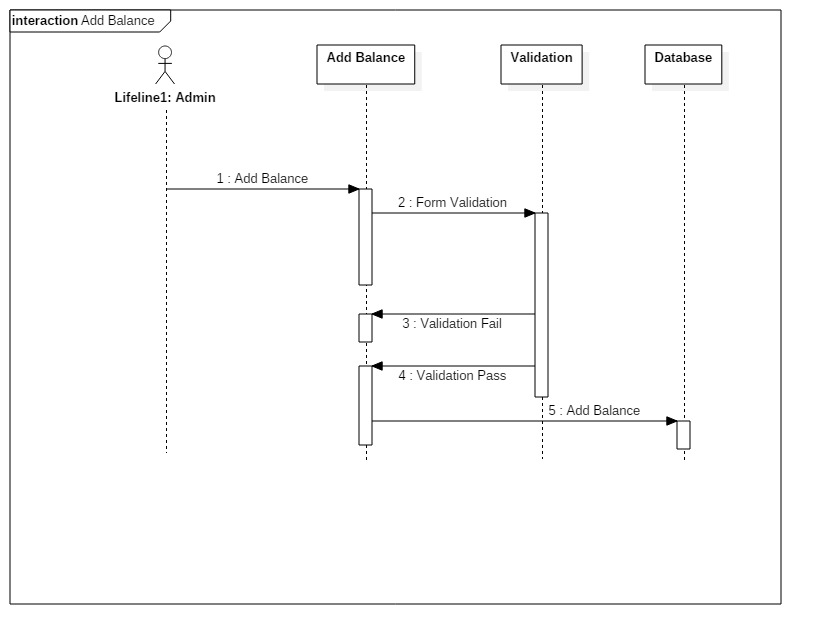


Fig: Sequence Diagram for Adding Balance

In above diagram, admin is the main actor for providing service for customer. Here admin plays the main role for adding customers, enabling the customer account, deactivating the customer account. Once the customer account is created customer deposit money into their account and to preview the balance of customer admin add the balance of the customer into the system and to their profile. Validation is carried for adding the balance. If validation is failed amount wont be added until the validation is passed the balance is added successfully to the customer account.

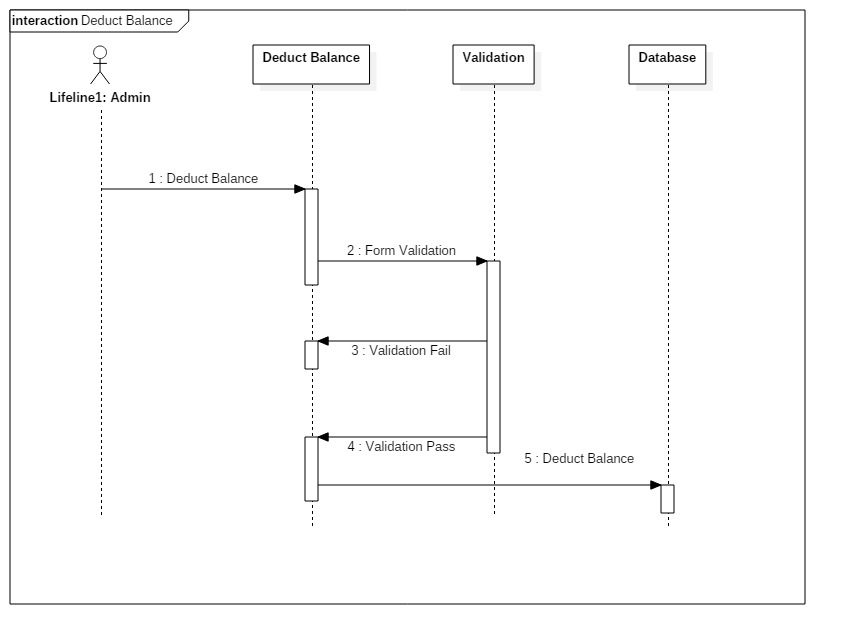


Fig: Sequence Diagram for Deducting Balance

In above diagram, customer withdrawn the amount form their account for their own use. Where admin need to update the balance that has been deducted. Admin goes to customer profile and deduct the balance.

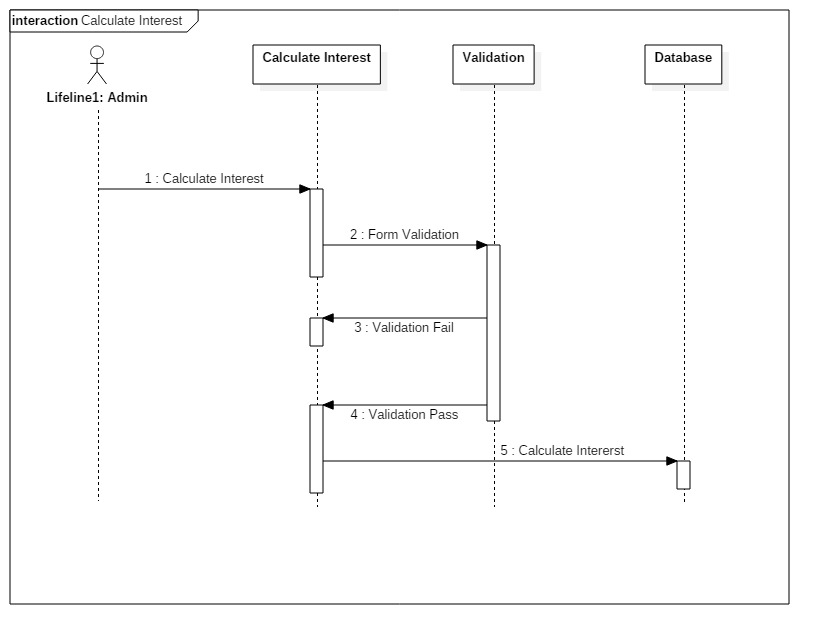


Fig: Sequence Diagram for Calculating Interest

In above diagram, customer is stored according to their account types, different account types have provided different interest. Customers are provided the interest within their account so admin calculate the interest with customer balance and sum up the balance.

# 3.2.2 Activity Diagram

Activity diagram helps to show the sequence of actions which are simple and intuitive pictorial representation. It is known as Work Flow Diagram. Activity diagram assists to show the process to execute the task in the way of flowchart. It demonstrates both computational and organizational operations of the user and client. Its easy for user to understand the structured workflow of the system from starting point to end point.

|  |  |  |  |
| --- | --- | --- | --- |
| SN | Notation Name | Notation | Description |
| 1 | Action |  | To represent the action |
| 2 | Control Flow |  | Shows the sequence of execution |
| 3 | Object Flow |  | Show the flow of one activity to another activity |
| 4 | Initial Node |  | Beginning of action |
| 5 | Final Node |  | End of the action |
| 6 | Object Node |  | Represents an object that is connected to a set of object flows |
| 7 | Decision Node |  | Condition checking to provide the path |
| 8 | Merge Node |  | Bringing different condition path to one node |
| 9 | Fork Node |  | Split behavior into set of parallel activities |
| 10 | Join Node |  | Bring back together a set of parallel activities |
| 11 | Swim Lane |  | A way to group activities performed by the same actor on an activity diagram. |

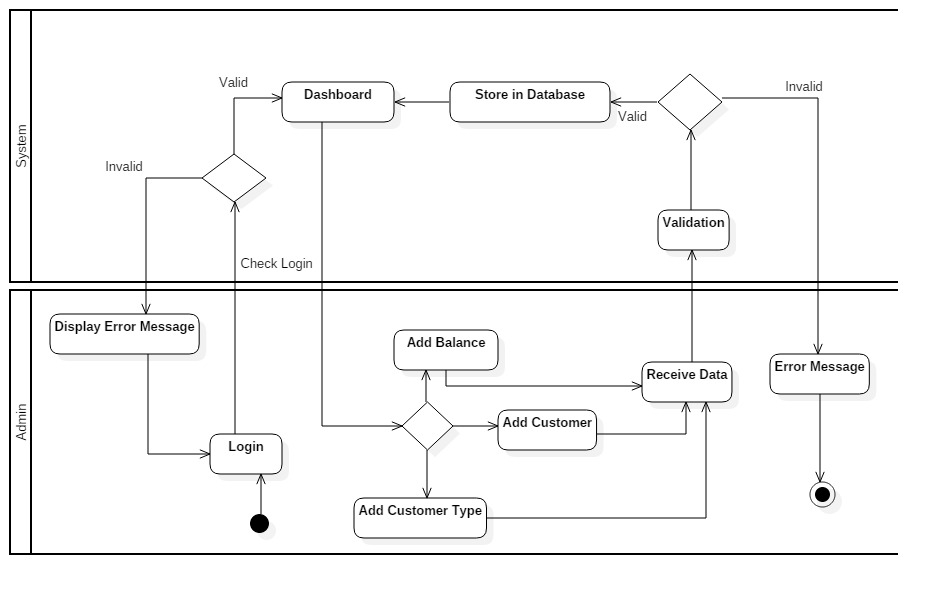


Fig: Activity Diagram to admin to insert

In above diagram, admin plays main role to the system where admin login in the system, if the detail is not verified then error message is displayed. Valid detail needs to be input for login after successful login dashboard page is popup. Admin address the customer account by adding customer to the system database. Admin enable the account for customer where customer deposit the balance and admin add balance to the customer account. All the records are stored in the system database if the data is invalid error message is passed.

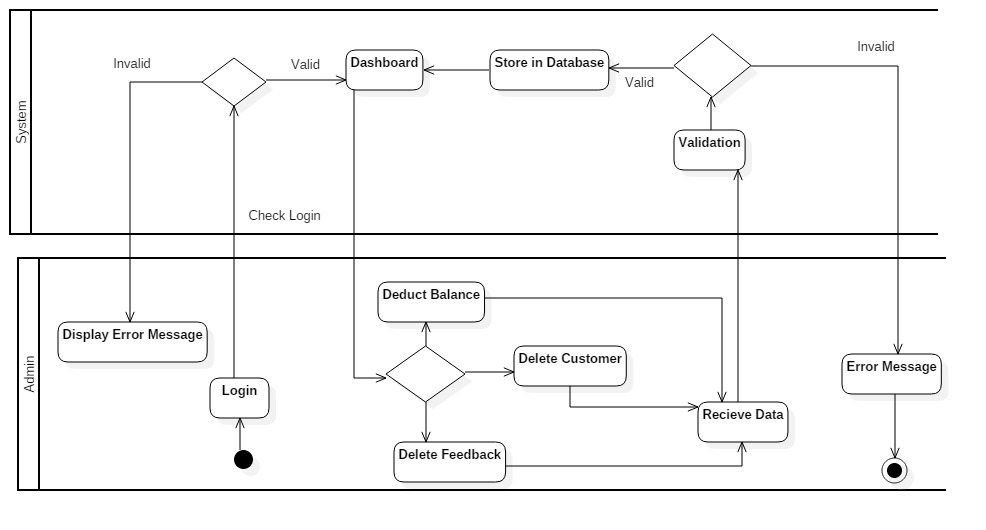


Fig: Activity diagram for admin to delete

Above diagram represents admin to delete the account of the customer if customer is not interested in using the system or out of the system. There account is disabled by the admin and delete from the database. Where having an account in the system, customer withdraw the amount from their account and admin deduct the balance from their account and update the balance.

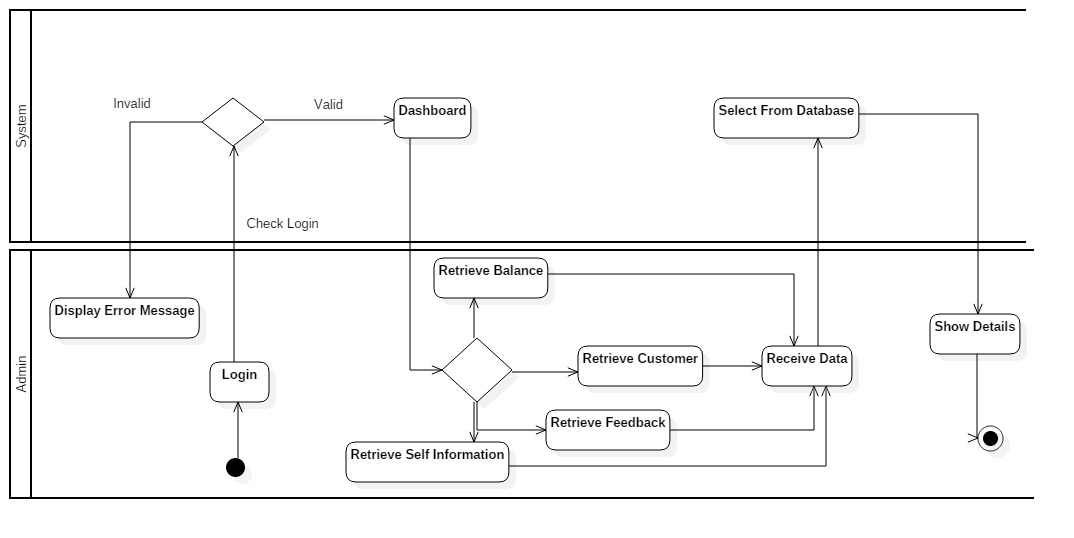


Fig: Activity diagram for admin to retrieve data

Above diagram, admin login in the system to retrieve the information of customer. Admin retrieve the balance of each customer, retrieve the all customer details and their feedback that are receive data in the database and provide details to the admin.

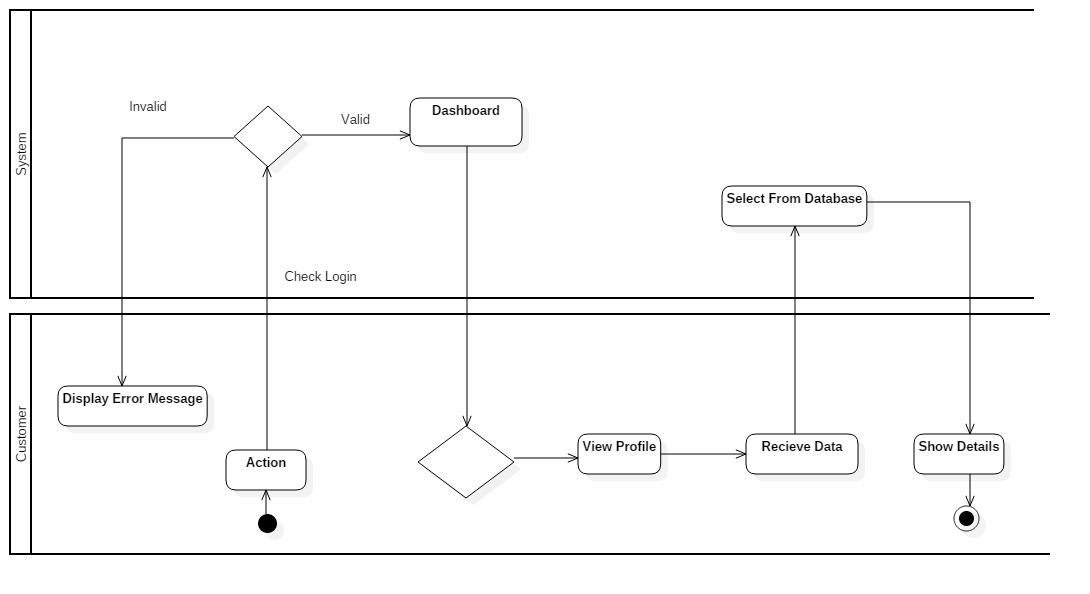


Fig: Activity diagram for admin to retrieve data

Above diagram presents even customer can retrieve their information through the website by login into the system. They can access their profile and edit it and update it and retrieve the account information and statement of their transactions.

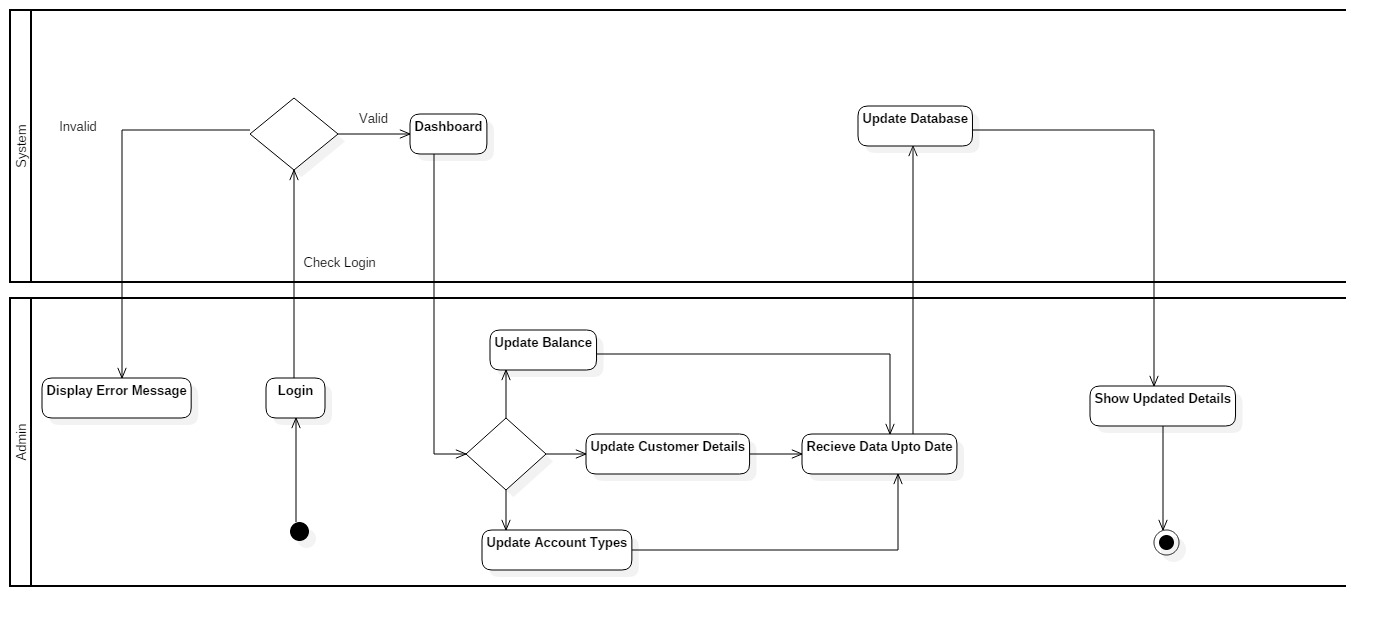


Fig: Activity diagram for admin to Update data

Above diagram, admin is main role of the system where admin has the power and responsibility of changing/ editing and updating the data of customers detail, account types and update the balance of each customer.

# 3.3 Database Model

Database model describes logical structure of a database tables and relationships between them. It determines how data can be stored, processed and organized. Database model are designed for a system, based on the rules of designer. Database models are represented through database Entity Relationship Diagram.

## **3.3.1 Data Dictionary**

**Table: Users**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attributes** | **Data Type** | **Length** | **Key** | **Null** |
| ID | Integer | 10 | Primary | No |
| Fname | Varchar | 225 | - | Yes |
| Lname | Varchar | 225 | - | Yes |
| Address | Varchar | 225 | - | Yes |
| Phone | Varchar | 225 | - | Yes |
| DOB | Date | 10 | - | Yes |
| Account\_Type | Varchar | 225 | - | Yes |
| Doc\_Type | Varchar | 225 | - | Yes |
| Doc\_Image | Varchar | 225 | - | Yes |
| Username | Varchar | 225 | - | Yes |
| Email | Varchar | 225 | - | Yes |
| Password | Varchar | 225 | - | Yes |
| Security\_Question | Varchar | 225 | - | Yes |
| Gender | Varchar | 225 | - | Yes |
| Balance | Double | 10 | - | Yes |
| Total | Double | 10 | - | Yes |
| UserType | Varchar | 225 | - | Yes |
| Status | Varchar | 225 | - | Yes |
| Created\_at | Timestamp | 20 | - | Yes |
| Updated\_at | Timestamp | 20 | - | Yes |

**Table: Account\_Typemodels**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attributes** | **Data Type** | **Length** | **Key** | **Null** |
| ID | Integer | 10 | Primary | No |
| Account\_TypeName | Varchar | 225 | - | Yes |
| InterestRate | Double | 10 | - | Yes |
| Status | Varchar | 225 | - | Yes |
| Created\_at | Timestamp | 20 | - | Yes |
| Updated\_at | Timestamp | 20 | - | Yes |
| UsersID | Integer | 10 | Foreign | Yes |

**Table: Feedback Model**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Attributes** | **Data Type** | **Length** | **Key** | **Null** |
| ID | Integer | 10 | Primary | No |
| Username | Varchar | 225 | - | Yes |
| Feedback | Varchar | 225 | - | Yes |
| Created\_at | Timestamp | 20 | - | Yes |
| Updated\_at | Timestamp | 20 | - | Yes |
| UsersID | Integer | 10 | Foreign | Yes |

## **3.3.2 ER Diagram**

An Entity Relationship (ER) diagram illustrates how entities such as people, objects or system. It is type of flow chart known as data modelling technique used to design and debug relational database in business information system.

Notation of ER Diagram

|  |  |  |  |
| --- | --- | --- | --- |
| SN | Notation Name | Notation | Description |
| 1 | Entity |  | data are stored like persons, place or things |
| 2 | Relationship |  | It determines how entities are related. |
| 3 | Cardinality |  | It demonstrates different cardinality relationships. |

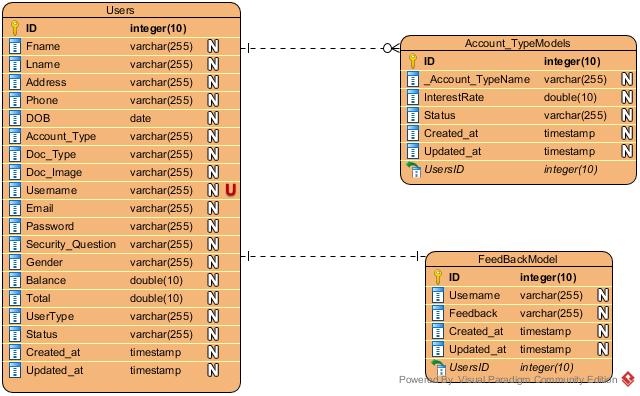


Fig: ER diagram

Entities like User, Account Type Model and Feedback Model are represented above ER Diagram with its attributes. Users are the main property of the project. Users are able to create their account with different account types. As the account type is selected by the users, they will be provided the detail of their interest rate of their account type. Admin here is provided to summarized the account holders detail like adding balance of users, deducting withdrawal amount and adding the interest rate of which account type is selected. Admin is the operator of users account details and able to response the feedback provided by the users.

# 3.4 System Architecture

System architecture illustrates how system work through view and define the structure and behavior of the system. It refers to conceptual model. It helps to understand the system how it works by providing overview of software and hardware of system depends upon operation. It helps in configuration management system. I have used Laravel framework which provide MVC pattern within the system for web application. MVC pattern is easy to understand and easy to make change in system for developers separating into three parts Model, View and controller. According to architecture my project is client-based architecture where database is related to model parts. Client request and output is provided to the clients by front facing operation through view. View is a part where user sees and interact. Inputs are taken by the controller part and passes or execute to view and model. Model, View and Controller are connected to each other and data passes to each other. Architecture is applied for hiding the main logical part from user it uses encapsulation. Change in the logical part won’t affect others as it makes system efficient.

Architecture diagram is demonstrated how is works in system:



Presentation (HTML)



Application (PHP)



Database (XAMPP)

Fig: System Architecture

**Presentation:** It is the user interface of the application. It is what the user sees and through which they communicate with the application. Graphical User Interface (GUI) in HTML is used to make it user friendly instead of command driven application.

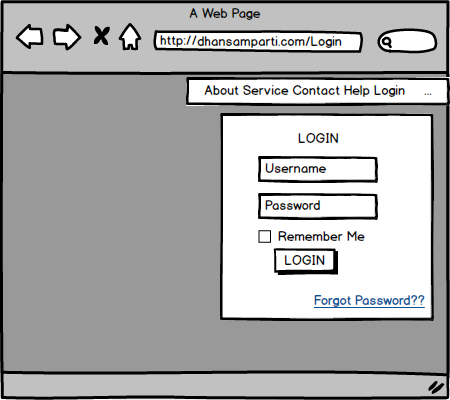
**Application:** It is the logical part of the system. It is where data are executed. It connects the model and view part.

**Database:** In development and testing XAMPP is used for a database. But in real life situation, a remote database would be required

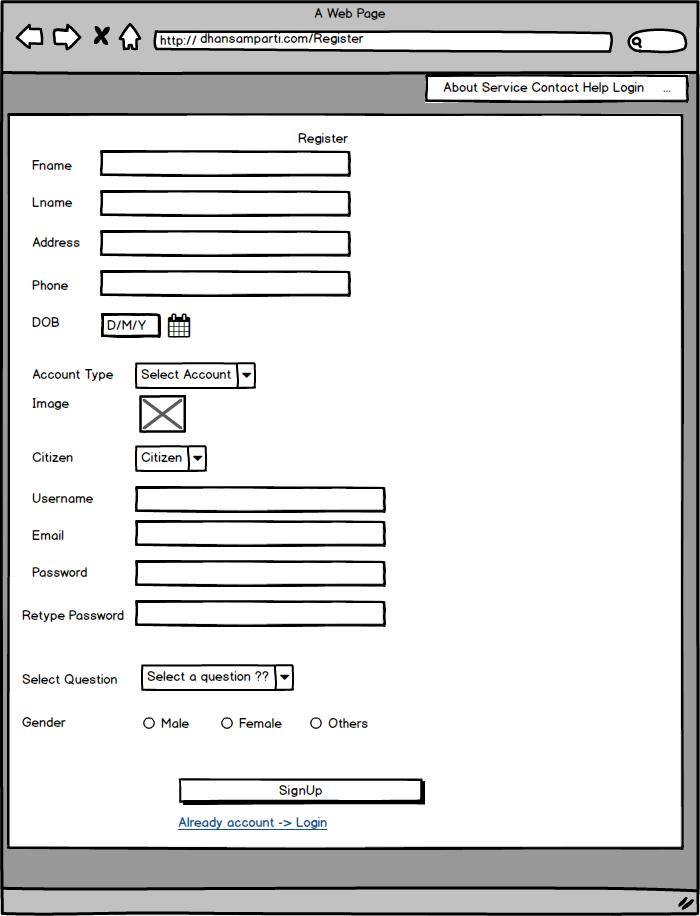
## 3.4.1 UI (User Interface) / Prototype

Prototype provides us how to use and work with system. It provides visualization of the system. It helps to guide for user who are aware of the system probably for the new users. It is a friendly behavior that provides manual guide of the system. To provide the assistance to people for using certain system a document is created.  
prototypes are provided:

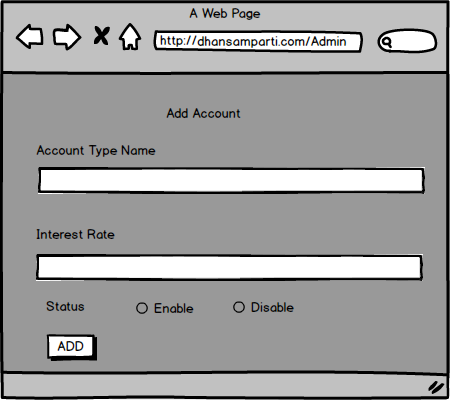
**Login**



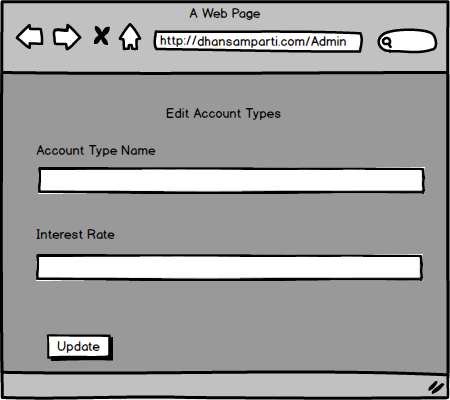
**Signup**

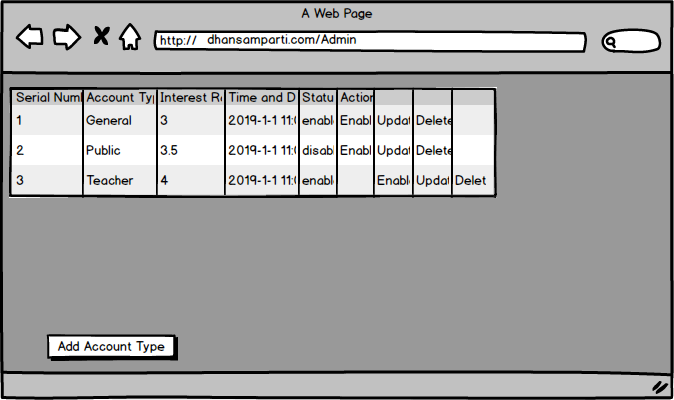


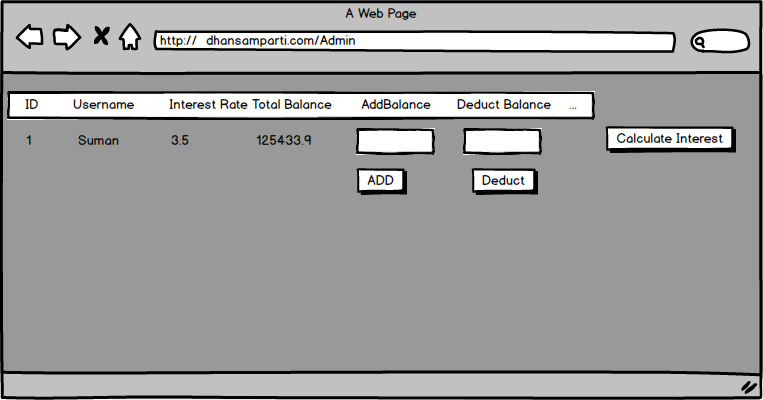
**Add Account Types**

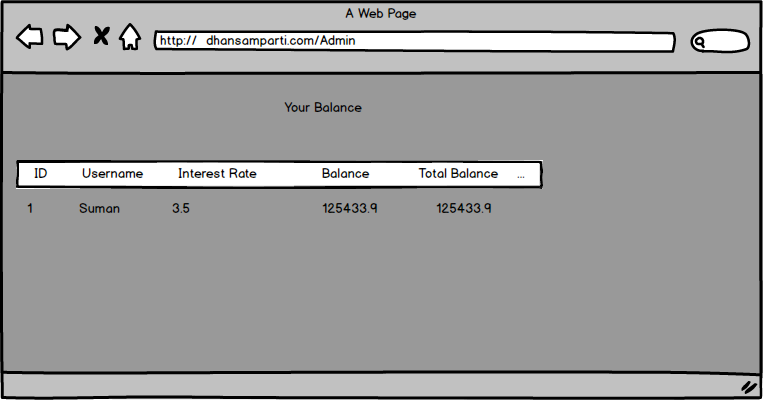


**Edit Account Types**



**Account Types Interface**

**Balance Management Interface**

**View Account Types**

**View/Edit Profile**