**MAJOR PROJECT ON:**

# INTERNET BANKING

**Major Project Submitted in Partial Fulfilment of the Requirements for the degree of Bachelor of Computer Application**

**Prepared By**


| NAME | ROLL NUMBER |
|---|---|
| KUSHANKUR DAS | 30901216087 |


## Under the Guidance of

**Mr. ARINDAM SENGUPTA**


**TECHNO INDIA SALT LAKE**
**EM 4/1, Sector V,**
**Kolkata-700091**


**2016-2019**

**Techno India, Salt Lake**
**Maulana Abul Kalam Azad University of Technology**
**(Formerly WBUT)**

**FACULTY OF BCA DEPARTMENT**

# Certificate of Recommendation

This is to certify that KUSHANKUR DAS have completed their project work titled “Major project on: INTERNET BANKING”, under the direct supervision and guidance of Mr.ARINDAM SENGUPTA.We are satisfied with their work, which is being presented for the partial fulfilment of the degree of Bachelor of Computer Application (BCA), West Bengal University of Technology (WBUT), Kolkata– 700032.

Date: _____ _____

Teacher in charge of project
ARINDAM SENGUPTA

Date: _____ _____

MONALISA BANERJEE
HOD BCA Department
TECHNO INDIA, SALTLAKE

# Maulana Abul Kalam Azad University of Technology
# Formerly WBUT

## FACULTY OF BCA DEPARTMENT

### Certificate of Approval

The foregoing Major project is hereby approved as a creditable study of Bachelor of Computer Application (BCA) and presented in a manner satisfactory to warrant its acceptance as a pre-requisite to the degree for which it has been submitted. It is understood that by this approval the undersigned do not necessarily endorse or any statement made, opinion expressed or conclusion therein but approve this Major project only for the purpose for which it is submitted.

_____

_____

_____

Signature of the Examiners

Final Examination for
Evaluation of the Project

Only in case the Major project is approved.

# PREFACE

## Goal:

The main goal of the software is to help the customer to access and maintain the detaile of their own bank account's information.

## Organization:

### Chapter 1:

**SCOPE OF THE PROJECT -**

In this chapter the introduction of the project is defined. An attempt has been made through this project to do all work ease & fast. It provides add, update& delete facilities to accomplish the desired objectives.

### Chapter 2:

**CONCEPT OF PROBLEM ANALYSIS -**

In this chapter we have discussed about the COCOMO Model and calculated the LOC, KLOC, Development Effort and Development Time.

### Chapter 3:

**THEORITICAL BACKGROUND -**

In this chapter the main background of the required HTML, CSS, JS,Jsp, Servlet, MySQL are described.

### Chapter 4:

**SOFTWARE REQUIREMENT SPECIFICATION -**

1. Introduction- In this chapter we have discussed so far about the main requirements for the project. In this section we have discussed about the project details; its scope; definitions, acronyms and abbreviations used; the references

from where we have got the details to use and the main responsibility of the developer in making this project.

2. General Descriptions- In this section the main overview of the project is given along with the user characteristics and general constraints and assumptions used.
3. Functional Requirements- This section deals with the descriptions of input and outputs in the project of what input will give which output to the user. It defines all the main functional requirements that have been used in making the project interface wise.

4. External Interface Requirements- This section gives us the details of how the interface will be used by the users. The error messages that will be shown are described here.
5. Design Constraints- This section gives us details of the hardware and software used to make the project.

## Chapter 5:

## DESIGN –

1. Data Design- This gives us the detailed diagram of Entity Relational Diagram of the project so made.
2. Interface Design- It holds the screenshot of all the interfaces.
3. Procedural Design- It gives us the details of the class names and its purpose, member function- name of function, argument list, return type, purpose and function algorithm.

## Chapter 6:

## CODING STANDARD FOLLOWED and ASSUMPTIONS –

This gives the details of the standard format of coding and assumptions that are followed.

## Chapter 7:

## TESTING –

This chapter gives us the detailed description of all the validation testing done. It gives us every testing details module wise. It specifies when we give any input what output is shown to us and whether the result passes the requirements.

## Chapter 8:

## FUTURE SCOPE OF THE PROJECT –

This topic gives us details about all the scopes in which we can work in future using this project.

## Chapter 9:

## CONCLUSION –

This topic gives us the brief description of the project's documentation.

## Chapter 10:

## REFERENCE AND BIBLIOGRAPHY –

This gives us the reference and links from where we have used to complete the project or the documentation part.

## Chapter 11:

## APPENDIX –

This section carries the coding of different module of the Internet Banking application.

# ACKNOWLEDGEMENT

It gives us a great sense of pleasure to present the report of BCA Project undertaken during BCA Final Year. We take this opportunity to express our sincere gratitude to all those who helped us in various capacities in undertaking this project and devising the report. We are privileged to express our sense of gratitude to our respected teacher Mr. Arindam Senguptawhose unparalleled knowledge, moral fibre and judgement along with his know-how was an immense support in completing the project. We are also grateful to Mrs. Monalisa Banerjee the Head of our Department, Bachelor of Computer Application for the brainwave and encouragement given. We take this opportunity also to thank our friends and contemporaries for their co-operations and compliance.

KUSHANKUR DAS

# INDEX


| Serial No. | Topic | Page No. |
|---|---|---|
| 1. | Scope of the Project | 10 |
| 2. | Concept and Problem Analysis | 11 |
| | 2.1 COCOMO Model | 11 |
| 3. | Theoretical Background | 12-15 |
| 4. | Software Requirement Specification | 16-24 |
| | 4.1 Introduction | 16 -18 |
| | 4.1.1 Purpose | 16 |
| | 4.1.2 Scope | 16 |
| | 4.1.3 Definition, Acronym and Abbreviation | 17 |
| | 4.1.4 Reference | 17 |
| | 4.1.5 Developer's Responsibility Overview | 18 |
| | 4.2 General Description | 18-19 |
| | 4.2.1 Product Overview | 18 |
| | 4.2.2 User Characteristics | 18 |
| | 4.2.3 General Constraints and Assumptions | 19 |
| | 4.3 Functional Requirements | 19 |
| | 4.3.1 General Description of Input and Output | 19 |
| | 4.3.2 Functional Requirements | 19 |
| | 4.4 External Interface Requirement | 20-23 |
| | 4.4.1 User Interface | 20-21 |
| | 4.4.2 Error Message | 22-23 |
| | 4.5 Design Constraints | 24 |
| | 4.5.1 Hardware Constraints | 24 |
| | 4.5.2 Software Constraints | 24 |

| | | |
|---|---|---|
| 5. | Design<br>5.1 Data Design<br>5.1.1 Enhanced Entity Relationship Diagram<br>5.1.2 Activity Diagram<br>5.1.3 Use Case<br>5.2 Interface Design<br>5.3 Procedural Design | 25-42<br>25<br>25<br>26<br>27<br>28-38<br>39-42 |
| 6. | Coding Standard followed and Assumption | 43-46 |
| 7. | Testing | 47-48 |
| 8. | Future Scope of the Project | 49 |
| 9. | Conclusion | 50 |
| 10. | Reference and Bibliography | 51 |
| 11. | Appendix | 52-98 |

# 1. SCOPE OF THE PROJECT

Without a sound and effective banking system in India it cannot have a healthy economy. The banking system of India should not only be hassle free but it should be able to meet new challenges posed by the technology and any other external and internal factors. Business of cooperative bank has increased phenomenally in recent years due to the sharp increase in numbers of urban co-operative banks. This exponential growth of Cooperative Banks in India is attributed mainly to their much better local reach, personal interaction with customers, and their ability to solve the problems of the local customers. The purpose of the project is to build and application program to reduce the manual work for bankers in bank. It also provides features such as viewing account balances, obtaining statements, checking recent transaction and making payments.

# 2.CONCEPT AND PROBLEM ANALYSIS

## 2.1 Cost Analysis using COCOMO Model:

We consider following cost drivers:

| Cost drivers | Ratting |
|---|---|
| Required software reliability | 1 |
| Complexity of the product | 1 |
| Required turnabout time | 1 |
| Analyst capability | 1 |
| Applications experience | 1 |
| Programming language experience | 1 |
| Use of software tools | 1 |

Total LOC = 1500
Then, KLOC = 1.5

EAF = 1*1*1*1*1*1*1 = 1

Development Effort = $3.2 * (KLOC)^{1.05} * EAF$
$= 3.2 * 1.5^{1.05} * 1$
= 4.9 PM (approx.)

Development time = $2.5 * (Effort)^{0.38}$
= 4.6 months (approx.).
= 4 months 18 days (approx.).

Note: we consider our project as organic type. And we follow Intermediate COCOMO estimation method to calculate effort and development time.

# 3.THEORETICAL BACK GROUND

## HTML (Hypertext Markup Language)

HTML is a computer language devised to allow website creation and also an evolving language. It doesn't stay the same for long before a revised set of standards and specifications are brought in to allow easier creation of prettier and more efficient sites. These websites can then be viewed by anyone else connected to the Internet.Hypertext is the method by which you move around on the web — by clicking on special text called hyperlinks which bring you to the next page. The fact that it is hyper just means it is not linear— i.e. we can go to any place on the Internet whenever we want by clicking on links — there is no set order to do things in.

**Markup** is what **HTML tags** do to the text inside them. They mark it as a certain type of text.

HTML can do anything since making websites became more popular and needs increased many other supporting languages have been created to allow new stuff to happen, plus HTML is modified every few years to make way for improvements.

Cascading Stylesheets are used to control how your pages are presented, and make pages more accessible. Basic special effects and interaction is provided by JavaScript, which adds a lot of power to basic HTML. Most of this advanced stuff is for later down the road.

## CSS (Cascading Style Sheets)

CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes.**C**ascading **S**tyle **S**heets, fondly referred to as CSS, is a simple design language intended to simplify the process of making web pages presentable.

CSS handles the look and feel part of a web page. Using CSS, you can control the color of the text, the style of fonts, the spacing between paragraphs, how columns are sized and laid out, what background images or colors are used, layout designs, variations in display for different devices and screen sizes as well as a variety of other effects.

CSS is easy to learn and understand but it provides powerful control over the presentation of an HTML document. Most commonly, CSS is combined with the markup languages HTML or XHTM.

## JS (JAVA SCRIPT)

JavaScript is a dynamic computer programming language. It is lightweight and most commonly used as a part of web pages, whose implementations allow client-side script to interact with the user and make dynamic pages. It is an interpreted programming language with object-oriented capabilities.

JavaScript was first known as **Live Script,** but Netscape changed its name to JavaScript, possibly because of the excitement being generated by Java. JavaScript made its first appearance in Netscape 2.0 in 1995 with the name **Live Script**. The general-purpose core of the language has been embedded in Netscape, Internet Explorer, and other web browsers.

Client-side JavaScript is the most common form of the language. The script should be included in or referenced by an HTML document for the code to be interpreted by the browser. The JavaScript client-side mechanism provides many advantages over traditional CGI server-side scripts. For example, you might use JavaScript to check if the user has entered a valid e-mail address in a form field.

The JavaScript code is executed when the user submits the form, and only if all the entries are valid, they would be submitted to the Web Server.

JavaScript can be used to trap user-initiated events such as button clicks, link navigation, and other actions that the user initiates explicitly or implicitly.

## Java

Java is a general-purpose computer-programming language that is concurrent, class based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to bytecode that can run on any Java virtual machine (JVM) regardless of computer architecture. As of now, Java is one of the most popular programming languages in use, particularly for client-server web applications, with a reported 9 million developers. Java was originally developed by James Gosling at Sun Microsystems and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

The original and reference implementation Java compilers, virtual machines, and class libraries were originally released by Sun under proprietary licenses. As of May 2007, in compliance with the specifications of the Java Community Process, Sun relicensed most of its Java technologies under the GNU General Public License. Others have also developed alternative implementations of these Sun technologies, such as the GNU Compiler for Java (bytecode compiler), GNU Classpath (standard libraries), and Iced Tea-Web (browser plugin for applets).
The latest version is Java 11, released on September 25, 2018, which follows Java 10 after only six months in line with the new release schedule. Java 8 is still supported but there will be no more security updates for Java 9. Versions earlier than Java 8 are supported by companies on a commercial basis.

## JSP (Java Server Pages)

Java Server Pages (JSP) is a server-side programming technology that enables the creation of dynamic, platform-independent method for building Web-based applications. JSP have access to the entire family of Java APIs, including the JDBC API to access enterprise databases. This tutorial will teach you how to use Java Server Pages to develop your web applications in simple and easy steps.

This is a technology for developing Webpages that supports dynamic content. This helps developers insert java code in HTML pages by making use of special JSP tags, most of which start with <% and end with %>.
A Java Server Pages component is a type of Java servlet that is designed to fulfill the role of a user interface for a Java web application. Web developers write JSPs as text files that combine HTML or XHTML code, XML elements, and embedded JSP actions and commands. JSP tags can be used for a variety of purposes, such as retrieving information from a database or registering user preferences, accessing JavaBeans components, passing control between pages, and sharing information between requests, pages etc.

## Servlet

Servlet technology is used to create a web application (resides at server side and generates a dynamic web page). Servlet technology is robust and scalable because of java language. Before Servlet, CGI (Common Gateway Interface) scripting language was common as a server-side programming language. However, there were many disadvantages to this technology. We have discussed these disadvantages below.

There are many interfaces and classes in the Servlet API such as Servlet, Generic Servlet, HttpServlet, Servlet Request, Servlet Response, etc.

Today we all are aware of the need of creating dynamic web pages i.e the ones which have the capability to change the site contents according to the time or are able to generate the contents according to the request received by the client. If you like coding in Java, then you will be happy to know that using Java there also exists a way to generate dynamic web pages and that way is Java Servlet. But before we move forward with our topic let's first understand the need for server-side extensions.

Servlets are the Java programs that runs on the Java-enabled web server or application server. They are used to handle the request obtained from the web server, process the request, produce the response, then send response back to the web server.

**Execution of Servlets:**

1. The clients send the request to the web server.
2. The web server receives the request.
3. The web server passes the request to the corresponding servlet.
4. The servlet processes the request and generate the response in the form of output.
5. The servlet send the response back to the web server.

## MySQL

MySQL is an open source relational database management system (RDBMS). Its name is a combination of "My", the name of co-founder Michael Widenius's daughter, and "SQL", the abbreviation for Structured Query Language. The MySQL development project has made its source code available under the terms of the GNU General Public License, as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for-profit firm, the Swedish company MySQL AB, now owned by Oracle Corporation. For proprietary use, several paid editions are available, and offer additional functionality.

MySQL is written in C and C++. Its SQL parser is written in yacc, but it uses a homebrewed lexical analyzer. MySQL works on many system platforms, including AIX, BSDi, FreeBSD, HP-UX, eComStation, i5/OS, IRIX, Linux, macOS, Microsoft Windows, NetBSD, Novell NetWare, OpenBSD, OpenSolaris, OS/2 Warp, QNX, Oracle Solaris, Symbian, SunOS, SCO OpenServer, SCO UnixWare, Sanos and Tru64. A port of MySQL to OpenVMS also exists.

The MySQL server software itself and the client libraries use dual-licensing distribution. They are offered under GPL version 2, beginning from 28 June 2000 (which in 2009 has been extended with a FLOSS License Exception) or to use a proprietary license.

Support can be obtained from the official manual. Free support additionally is available in different IRC channels and forums. Oracle offers paid support via its MySQL Enterprise products. They differ in the scope of services and in price. Additionally, a number of third party organisations exist to provide support and services, including MariaDB and Percona.

MySQL has received positive reviews, and reviewers noticed it "performs extremely well in the average case" and that the "developer interfaces are there, and the documentation (not to mention feedback in the real world via Web sites and the like) is very, very good". It has also been tested to be a "fast, stable and true multi-user, multithreaded sql database server".

# 4. SOFTWARE REQUIREMENT SPECIFICATION

## 4.1 INTRODUCTION

**Online banking**, also known as **internet banking**, is an electronic payment system that enables customers of a bank or other financial institution to conduct a range of financial transactions through the financial institution's website. The online banking system will typically connect to or be part of the core banking system operated by a bank and is in contrast to branch banking which was the traditional way customers accessed banking services.

Some banks operate as a "direct bank" (or “virtual bank”), where they rely completely on internet banking.

Internet banking software provides personal and corporate banking services offering features such as viewing account balances, obtaining statements, checking recent transaction and making payments. Access is usually through a secure web site using a username and password, but security is a key consideration in internet banking and many banks also offer two factor authentication using a security token.

### 4.1.1Purpose

The main purpose of the project on Internet Banking System is to manage the details of customer profiles, general information, banking details, account statements etc. The project is totally built at customer and administrative end. The purpose of the project is to build an application program to reduce the manual work for bankers in bank. It also provides features such as viewing account balances, obtaining statements, checking recent transaction and making payments.

### 4.1.2 Scope

The present project has been developed to meet the aspirations indicated in the modern age. An attempt has been made through this project to do all work ease & fast. The purpose of the project is to build and application program to reduce the manual

work for bankers in bank. It also provides features such as viewing account balances, obtaining statements, checking recent transaction and making payments.

It may help collecting perfect management in details. In a very short time, checking out details of one's account will be obvious, simple and sensible. It will help a person to know and manage such as viewing account balances, obtaining statements, checking recent transaction and making payments.

### 4.1.3 Definitions, Acronyms, Abbreviations

- IB: Internet Banking
- SERVER: refers to the Host machine
- USER: Refers to the user of IB
- SQL: Structure Query Language; used to retrieve information, database
- BOOLEAN: A true/false notation
- UNIQUE key: Use to deferential entries in a data base.
- LAYER: Represent a section of the project
- DATA BASE: The section of the assignment referring to where all data is recorded.

## 4.1.4 References

https://www.w3schools.com/html/default.asp

https://www.w3schools.com/css/default.asp

https://www.w3schools.com/js/default.asp

https://www.javatpoint.com/java-tutorial

https://www.tutorialspoint.com/html_online_training/index.asp

https://www.tutorialspoint.com/css_online_training/index.asp

https://en.wikipedia.org/wiki/Cascading_Style_Sheets

### 4.1.5Developer's Responsibility Overview

The roles and responsibilities of the developer of this project is very vast. Here are common examples that are to be followed:

- Designing, implementing, and maintaining the application as it is required for mission-critical system.
- Delivering high availability and performance.
- Contributing in all phases of the development lifecycle
- Writing well-designed, efficient, and testable code.
- Conducting software analysis, programming, testing, and debugging.
- Managing Java application development.
- Ensuring designs comply with specifications.
- Preparing and producing releases of software components.
- Transforming requirements into stipulations.
- Maintaining user friendly operations with proper direction of use.
- Support continuous improvement.
- Investigating alternatives and technologies.
- Presenting for architectural review.

## 4.2 General Descriptions

### 4.2.1 Product Overview

Product function will include the following functional areas:

- Admin log into the application with user name and password.
- If the administrator enters invalid name or password, then they will not be allowed to do any operations.
- In case of normal user, i can also log into the application with login id and password.
- If the user enters invalid name or password, then they will not be able to update their information.
- User can see their bank account details and can get update anytime they want.
- The log in id must be unique.

### 4.2.2 User Characteristics

- Case of new user, i must go through the sign up button to create their id. They need to give their information like: - login id, password, name, sex, birth date, contact no., email id and address.

### 4.2.3 General Constraints and Assumptions

We need to submit this project on 17 May 2019.
If we could get more time, then we would have definitely improved our software by adding more facilities.

## 4.3 Functional Requirements

### 4.3.1 General Description of Input and Output

Mainly the software is being used by the customer of the bank.

**In the user mode-** When i create new user account profile by using provided login id and password and giving the details we save them. Inside the interface i can check out account balances, obtaining statements, checking recent transaction and making payments.

### 4.3.2 Functional Requirements

The main functional requirement is to maintain Internet Banking System:

**User mode:**

- Make sure that they can view his/her information.
- They can view and search the details of banking statements.
- They can always be updated anywhere and check provided details of the bank viewing account balances, obtaining statements, checking recent transaction and making payments.

# 4.4External Interface Requirements

## 4.4.1 User Interface

**Interface 1.: -**In the login interface the admin or any other user will provide the login and register.

**Interface 1.1: -**In new account registration there is user name, password, name, phone number, email address, date of birth, address fields which has to be filled up and new user id will be created.

**Interface 1.2: -** In case a user or an admin wants to login then they will go through the Login button and provide user name and password and the following interface will be opened.

**Interface 2.1.0: -**In home page inside snapshot user can see last five transactions of the user. On right side user can see the 'current balance' and using quick action user can 'update profile' and 'change password' easily.

**Interface 2.1.1: -**In 'update profile' there are user name, name, phone number, email address, birth date, address fields which has to be filled up by the authorized user and profile will be updated easily.

**Interface 2.1.2: -**In 'change password' there are current password, new password, re-type password which has to filled up by the user and by clicking change, new password will be set.

**Interface 2.2.0: -**In homepage inside accounts user can see 'savings account details', 'fixed deposit details', 'recurring deposits details' of the authorized user. User can also check out balance, rate of interest, date of maturity. Below the account's details, there is a button of "create new account" which user can use to create new account easily.

**Interface 2.2.1: -**In create new account there is a 'type' dropdown list from which user can choose from fixed, recurring, savings account. In second row number of months can be selected by user for their respected account creation. In third row amount must be placed by user for their respected account creation.

**Interface 2.3.0: -**In homepage inside transfer funds user can see all the details of list of payee (that is users transfer details amount to another user) through online banking. Users can always keep track of every payee details in 'Transfer funds' tab.

**Interface 2.3.1: -**If user clicks on ‘Add payee’ tab then a popup screen will appear in which user have to give all the details such as account no, holder name, bank name, and then if user clicks on add tab then details of new payee will appear under ’list of payee’.

**Interface 2.3.2: -**If user clicks on ‘pay’ tab then user have to fill in following details like: ‘account no’, ‘to payee account details’, ‘holder name’, ‘bank name’, ‘password’. Then user can make transaction to another payee easily.

**Interface 2.4.0: -**In homepage inside statement tab user can see all the transactions details that the user has credited or debited from any respected dates user wants.

**Interface 2.5.0: -**This Forex features is included in future scope of the project. Later, we will try to input the currency exchange feature in it.

**Interface 3.1.0:-**In snapshot that is in homepage of admin can check out total no of customers, total no of (savings ,fixed ,recurring deposits) and also using quick action admin can change password as well as update profile.

**Interface 3.1.1: -**In ‘update profile’ there are user name, name, phone number, email address, birth date, address fields which has to be filled up by the authorized admin and profile will be updated easily.

**Interface 3.1.2: -**In ‘change password’ there are current password, new password, re-type password which has to filled up by the admin and by clicking change new password will be set.

**Interface 3.2.0:-**In customer tab it will show all the list and details of customers as per needs of admin. Admin can get all the details about their customers whenever they need.

**Interface 3.2.1:-**If admin wants to add new customer details then they can click on ‘new customer’ tab and fill up the following details like user name, password, name, phone number, email address, birth date, address and then by clicking create admin can create a new ‘user id’ and ‘password’ for customers.

**Interface 3.3.0: -**In homepage inside accounts tab admin can see ‘savings account details’, ‘fixed deposit details’, ‘recurring deposits details’ of the authorized user. Admin can also check out their ‘active’ or ‘inactive’ status including account no, holder name, balance, interest rate, no of months of maturity.

**Interface 3.3.1: -**On right hand side we can see create new account which admin can use to create new account easily. In create new account there is a ‘account type’ dropdown menu from which admin can choose from fixed, recurring, savings account. In second row number of months can be selected by admin for their respected account creation. In third row amount must be placed by admin for their respected account creation. Interest rate must also be provided by the admin before creation of new account.

**Interface 3.4.0: -**In homepage inside transaction tab admin can see all the transactions details of the user that have been credited or debited from any respected dates admin wants.

## 4.4.2 Error Messages

### Internet Banking - Login Screen

1. If we enter wrong id or password at the time of log in, then there will be shown an error message as: “Wrong Id or Password”.

### Internet Banking - Sign Up

1. In the Signup form if we enter wrong phone number, or any other detail then the error message will be shown as: “Value must be greater than or equal to 6000000000”.
2. Email id should be correct or an error message will be shown as: “Value must be the correct email address”.
3. If during signup user id with same credentials an error message will be shown as: “User Name is not available, please try again later”.

### Internet Banking - Update Profile.

1. Phone Number should be of 10 digits or anerror message will be shown as: “Value must be greater than or equal to 6000000000”.
2. Email id should be correct or anerror message will be shown as: “Value must be the correct email address”.

### Internet Banking - Change Password

1. Current password should be correct or error message will be displayed as “Wrong current password”.

### Internet Banking -Fund Transfer

1. Balance should not be less than minimum balance or transaction will not take place or an error message will be shown as: “Insufficient Balance”.
2. Account number should be correct or an error message will be shown as: “Invalid Payee Information”.

### Internet Banking -Accounts

1. Create a new Account for fixed or recurring account or an error message will be shown as: “Insufficient Balance”.

### Internet Banking -Customer

1. Phone Number should be of 10 digits or an error message will be shown

as: “Value must be greater than or equal to 6000000000”..
2. Email id should be correct or an error message will be shown as: “Value must be the correct email address”..

## Internet Banking –Customer (Add new Customer)

1. If during signup user id with same credentials an error message will be shown
as: “User Name is not Available Please try again later”.

2. Phone Number should be of 10 digits or an error message will be shown as: “Value must be greater than or equal to 6000000000”.

3. Email id should be correct or an error message will be shown as: “Value must be the correct email address”.

# 4.5Design Constraints

## 4.5.1 Hardware Requirements

**Server side: -**

- Processor: x64, 1.4Ghz
- Memory: 1GB
- Drive space: 1GB

**Client side: -**

- Processor: x32 or x64 1Ghz
- Memory: 256MB
- Drive space: 100MB

## 4.5.2 Software Requirements

**Server side: -**

- MySQL server
- Tomcat Apache server
- Java runtime environment

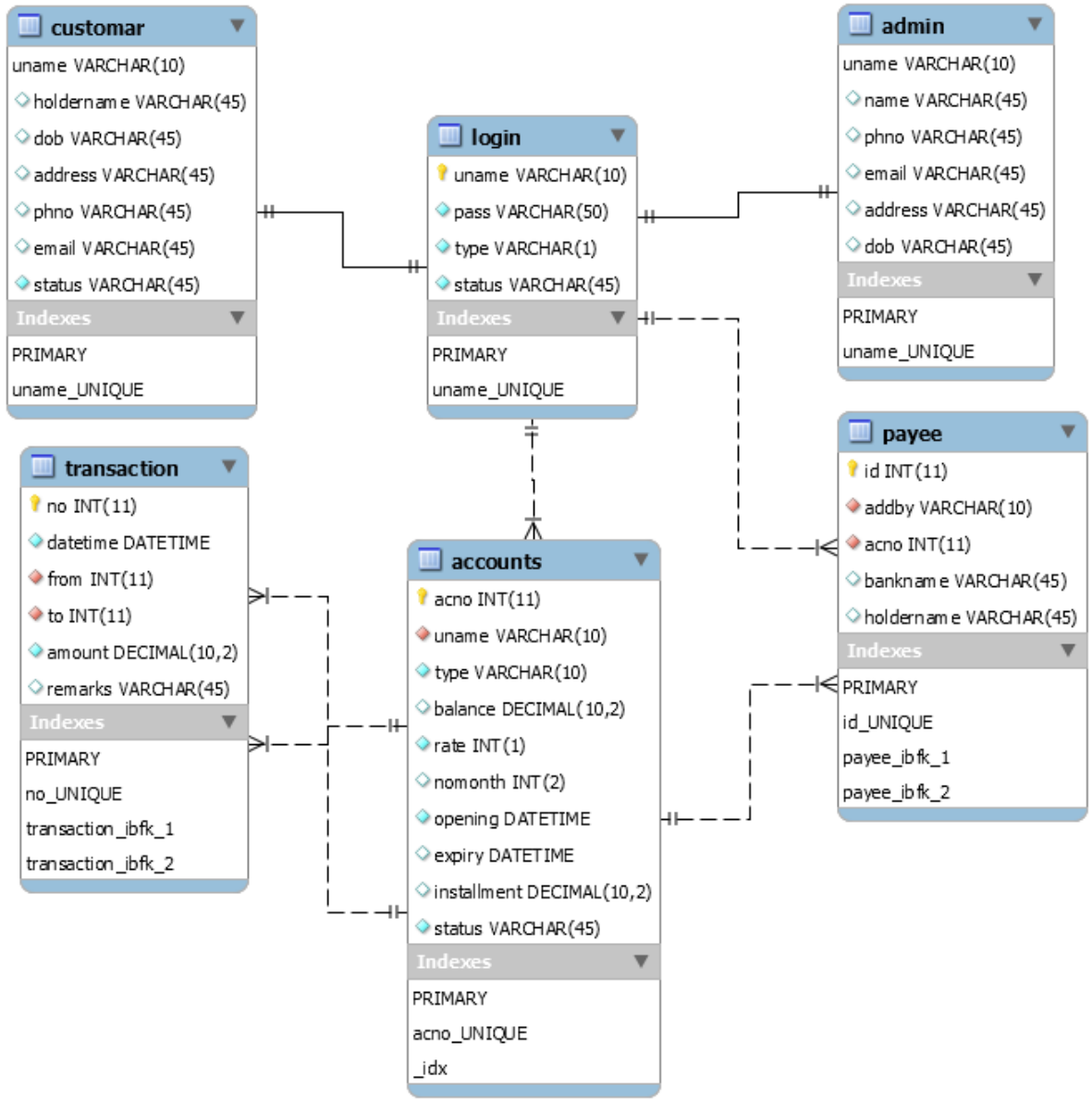
**Client side: -**

- Web browser

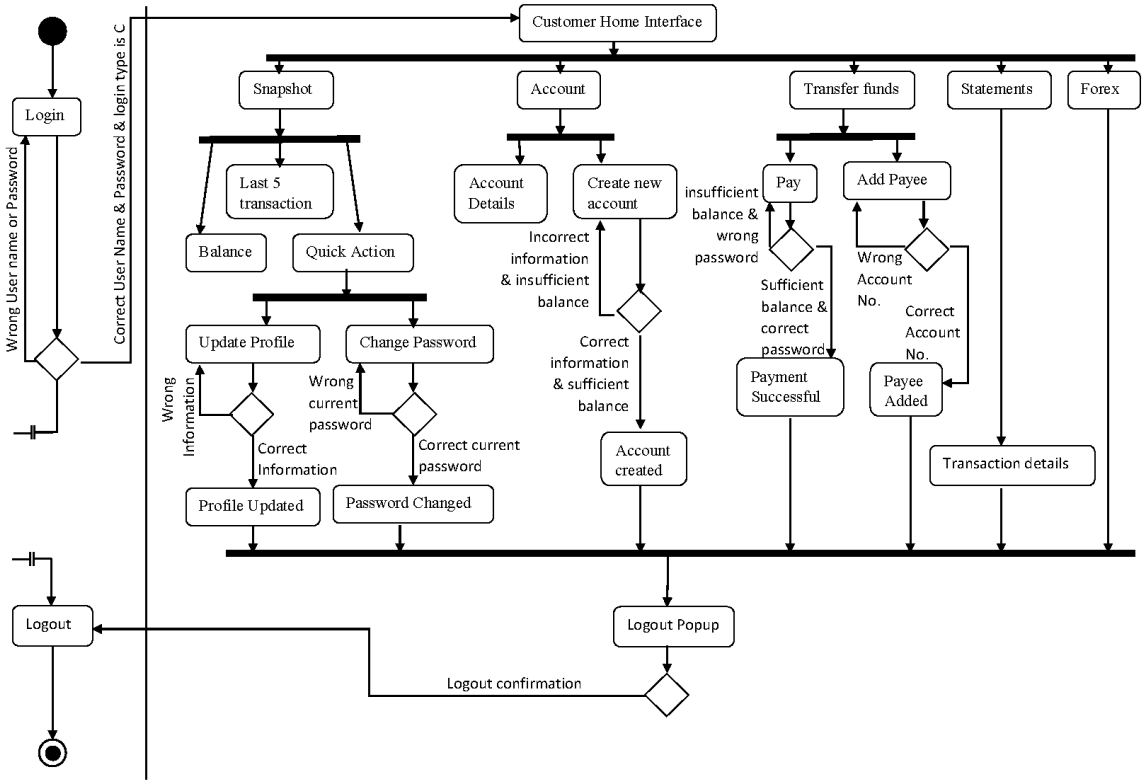
# 5. DESIGN

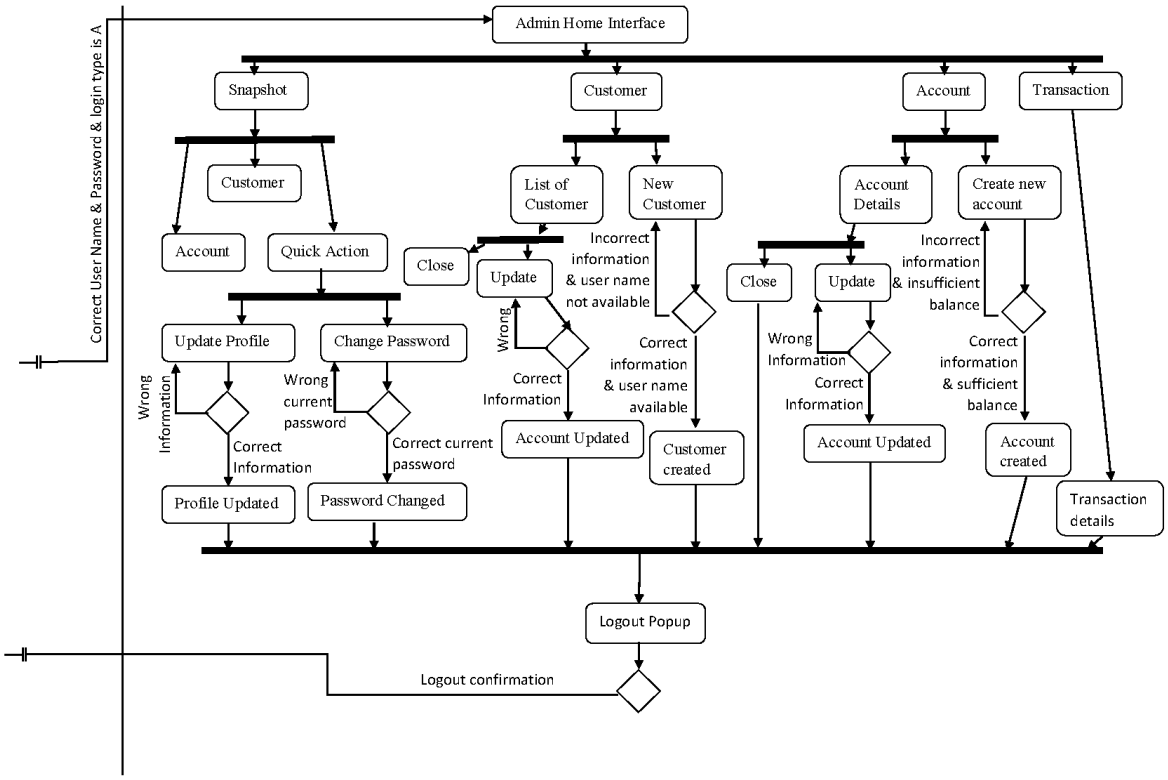
## 5.1Data Design

### 5.1.1 Enhanced Entity Relationship Diagram

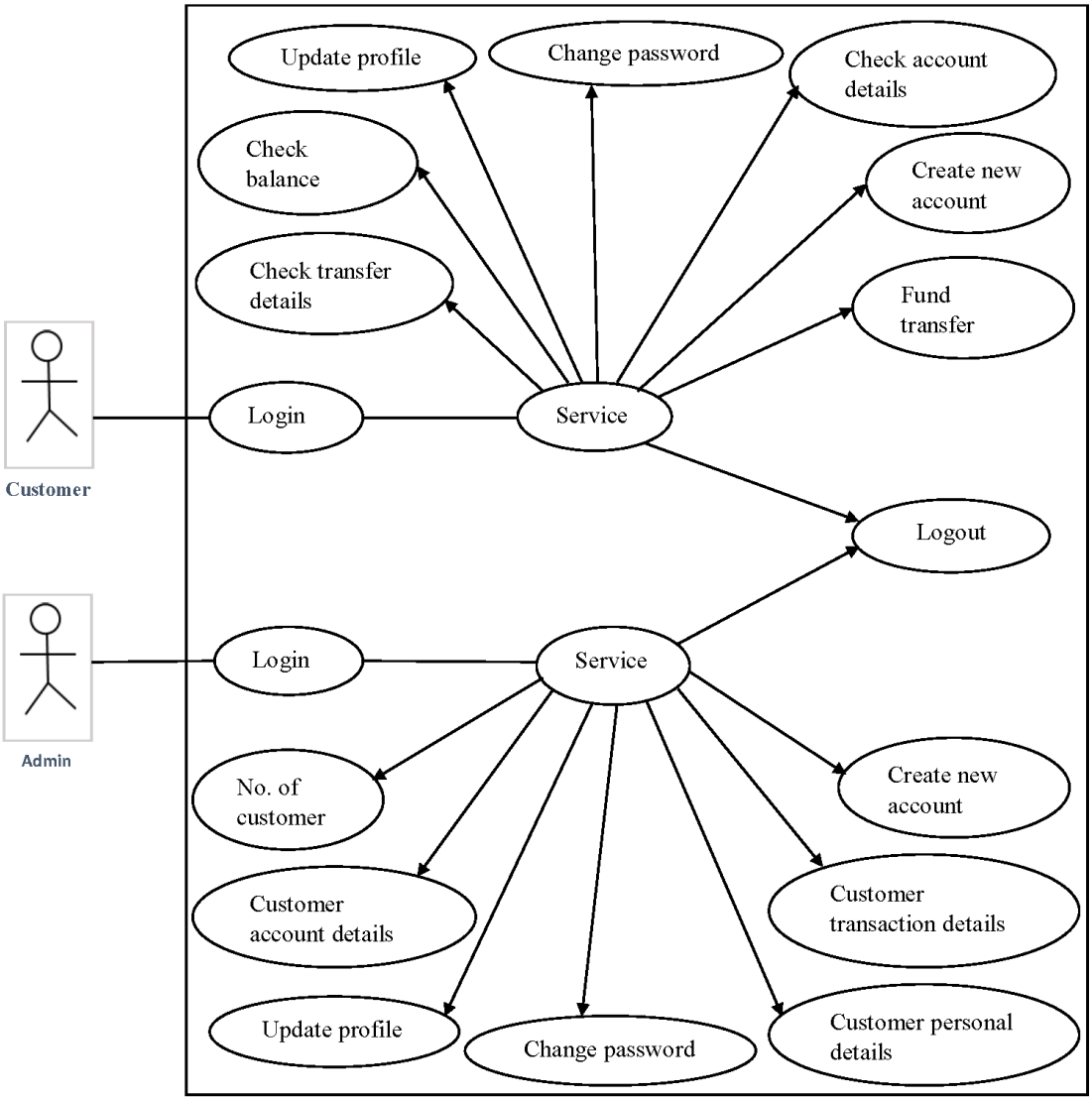


# 5.1.2 Activity Diagram

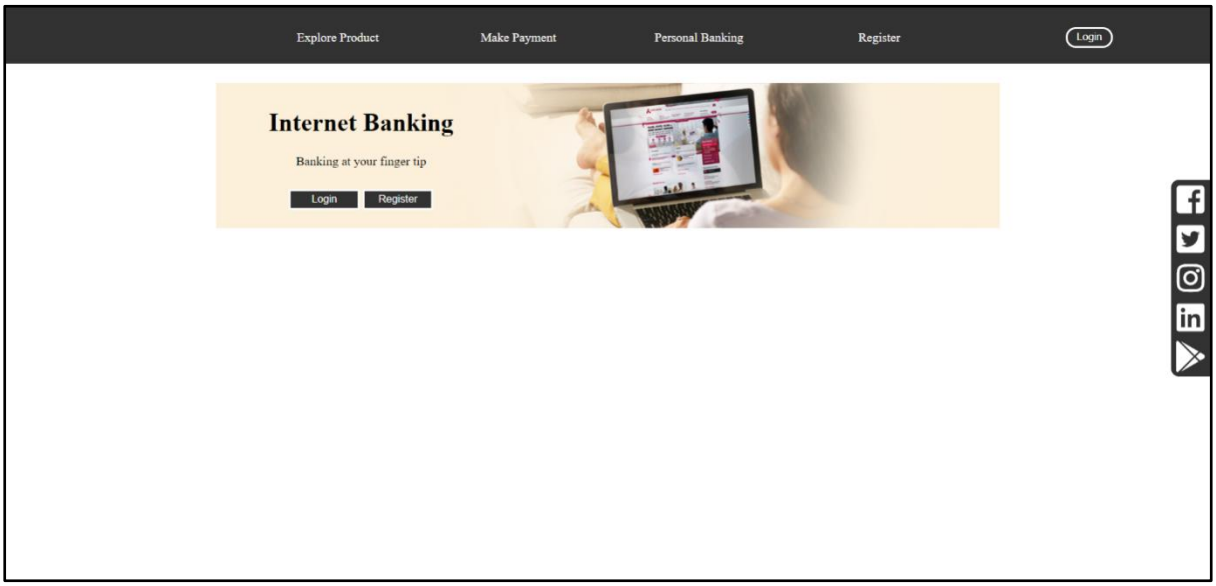


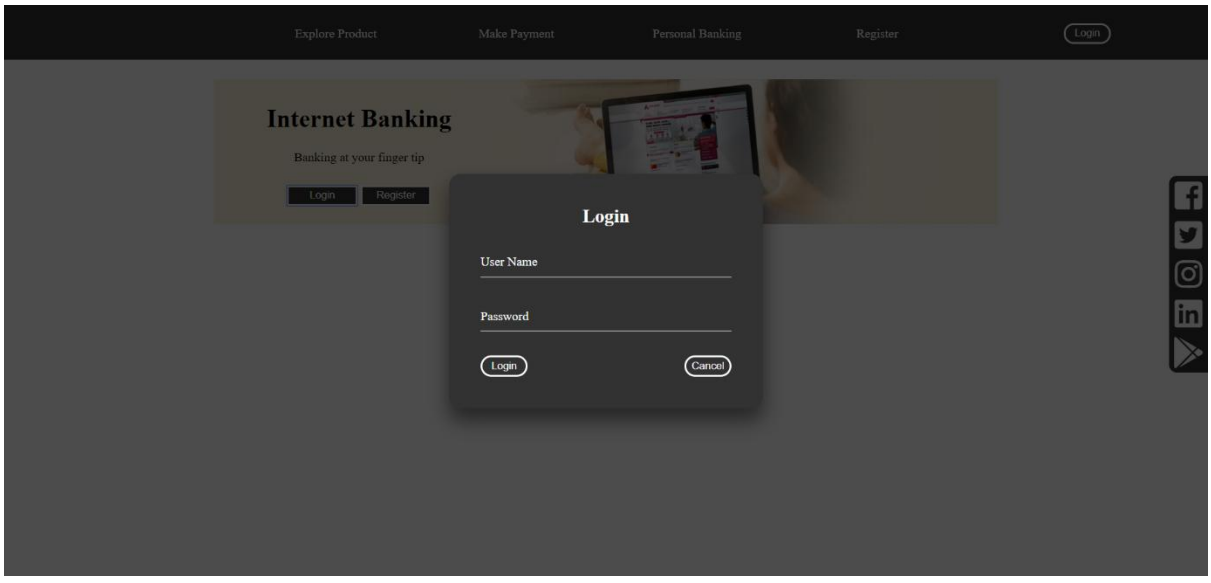
## 5.1.3 Use Case


Update profile
Change password
Check account details
Check balance
Create new account
Check transfer details
Fund transfer
Customer
Login
Service
Logout
Admin
Login
Service
No. of customer
Create new account
Customer account details
Customer transaction details
Update profile
Change password
Customer personal details

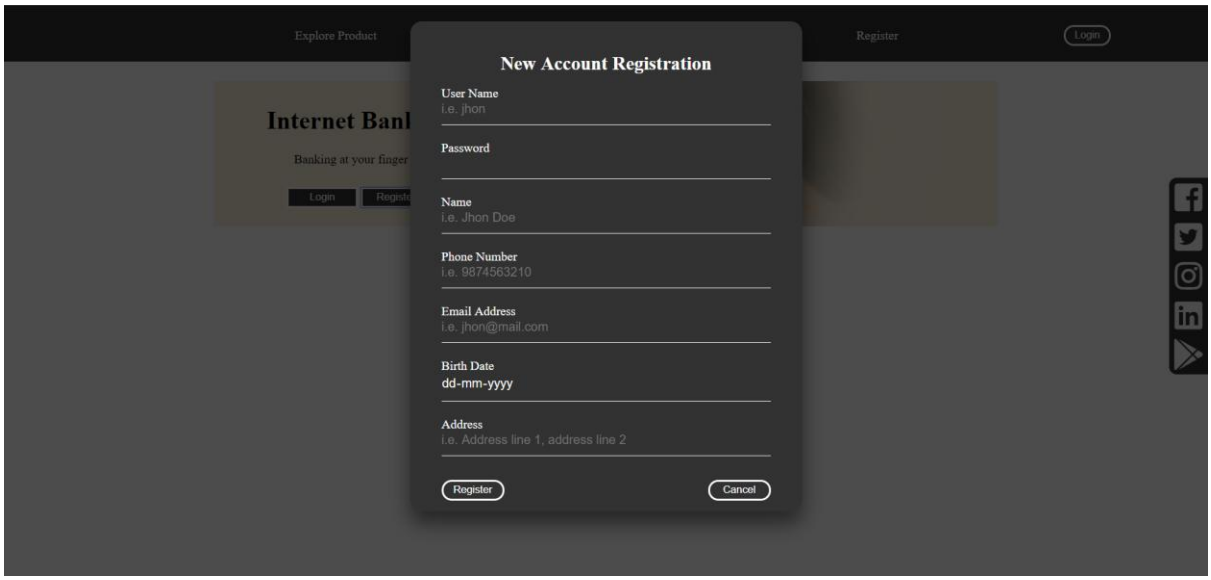
# 5.2Interface Design



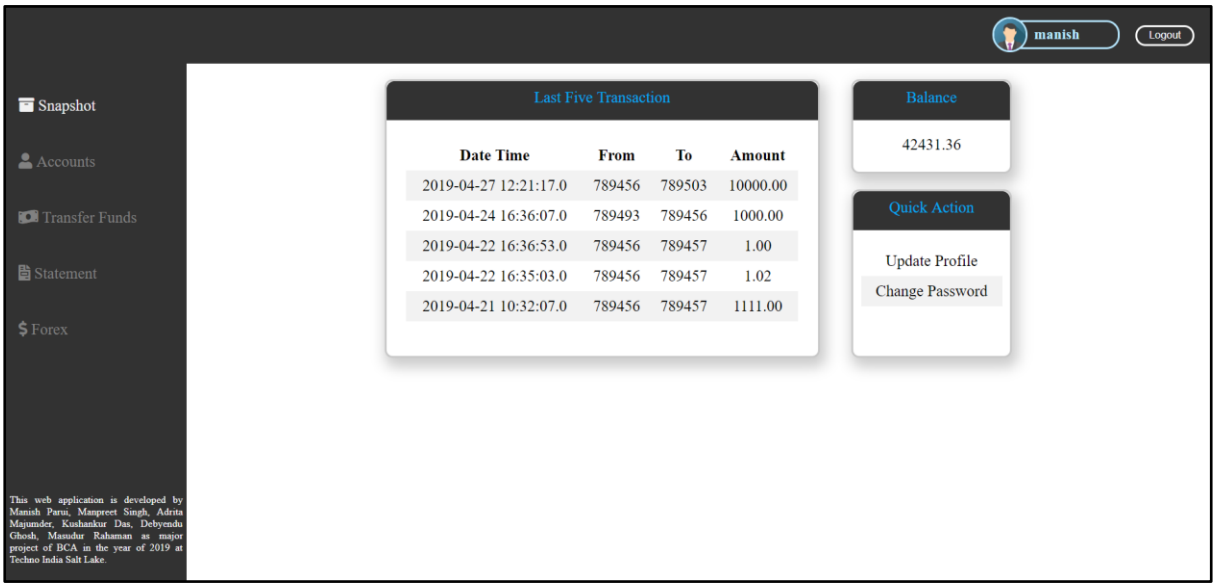

Interface 1.0



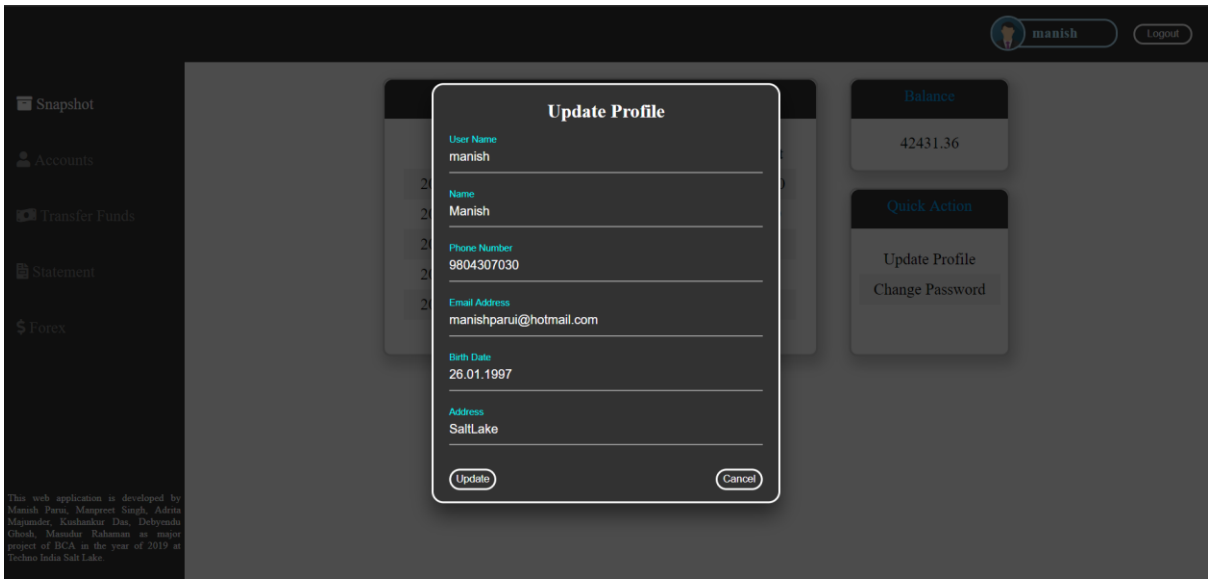

Interface 1.1



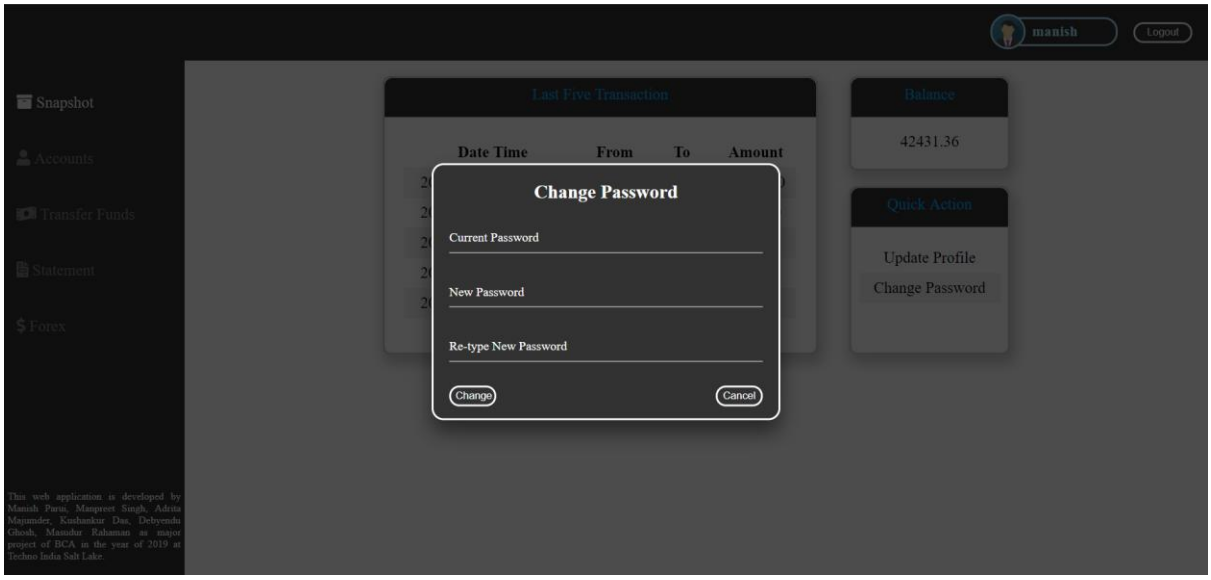

Interface 1.2



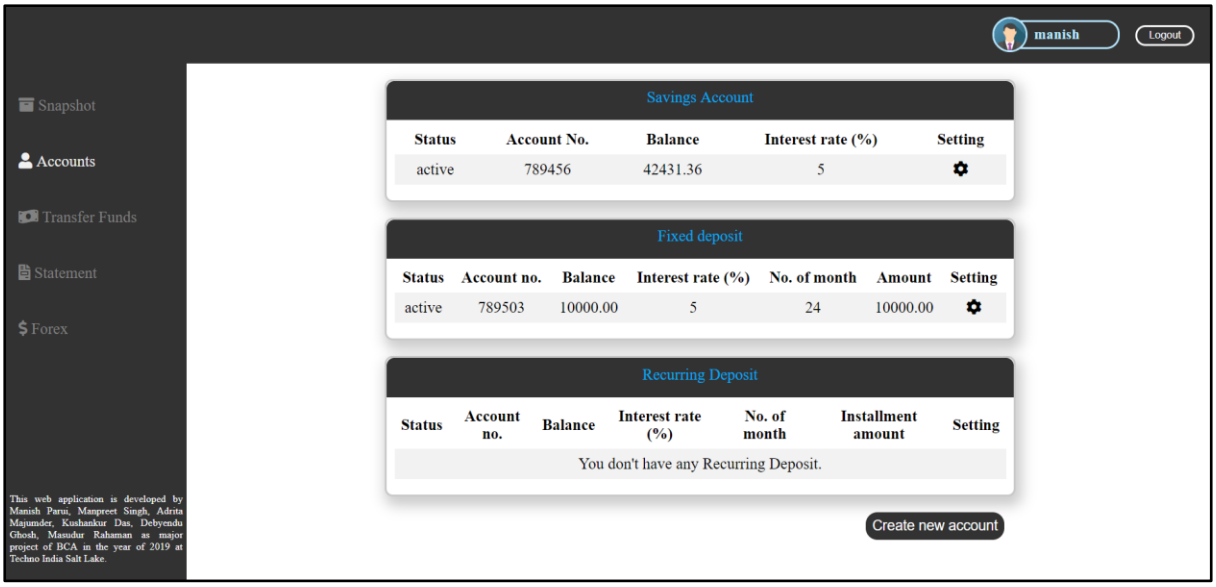

Interface 2.1.0



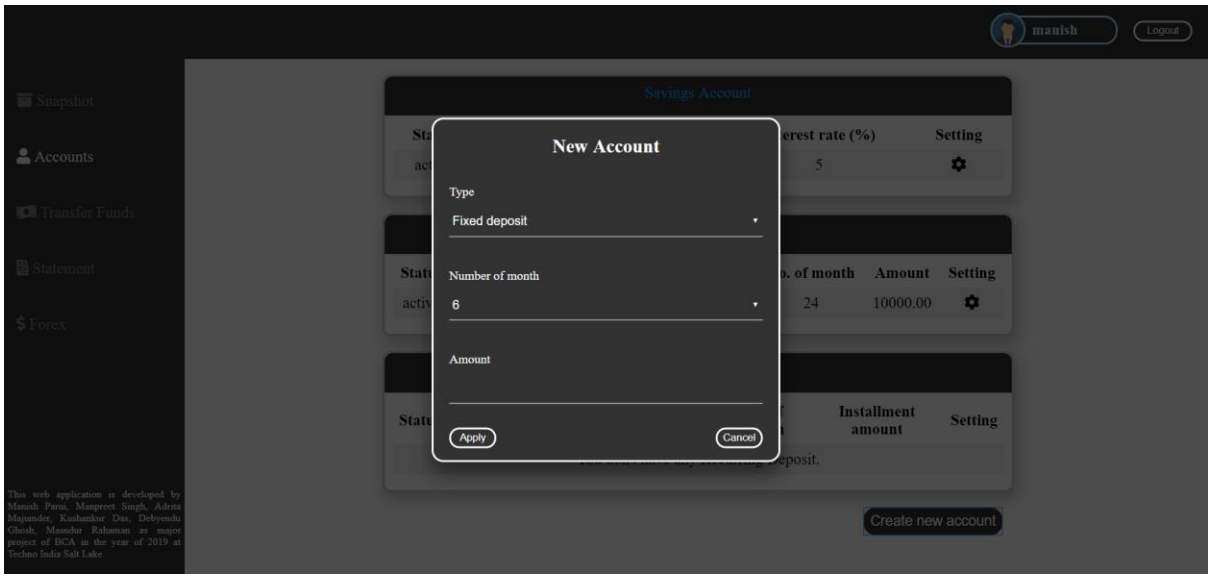

Interface 2.1.1



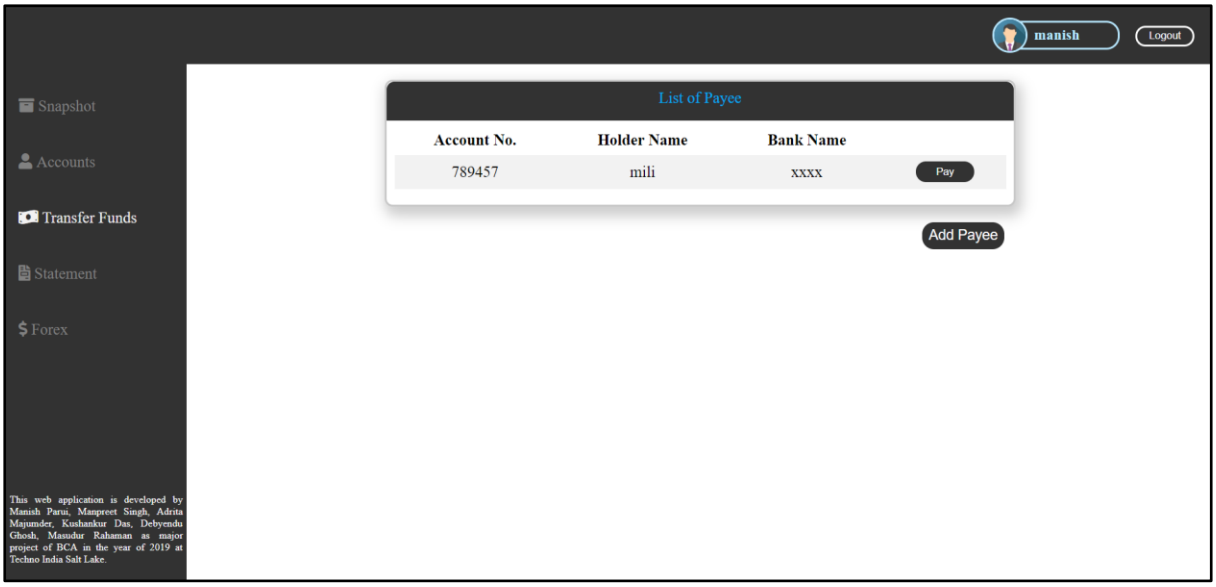

Interface 2.1.2



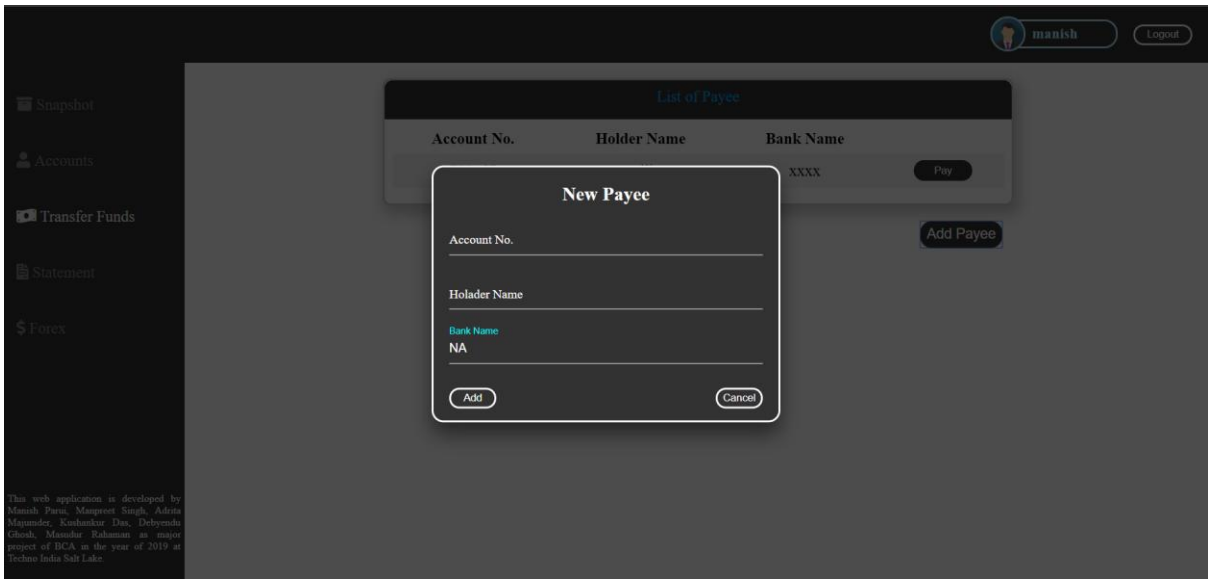

Interface 2.2.0



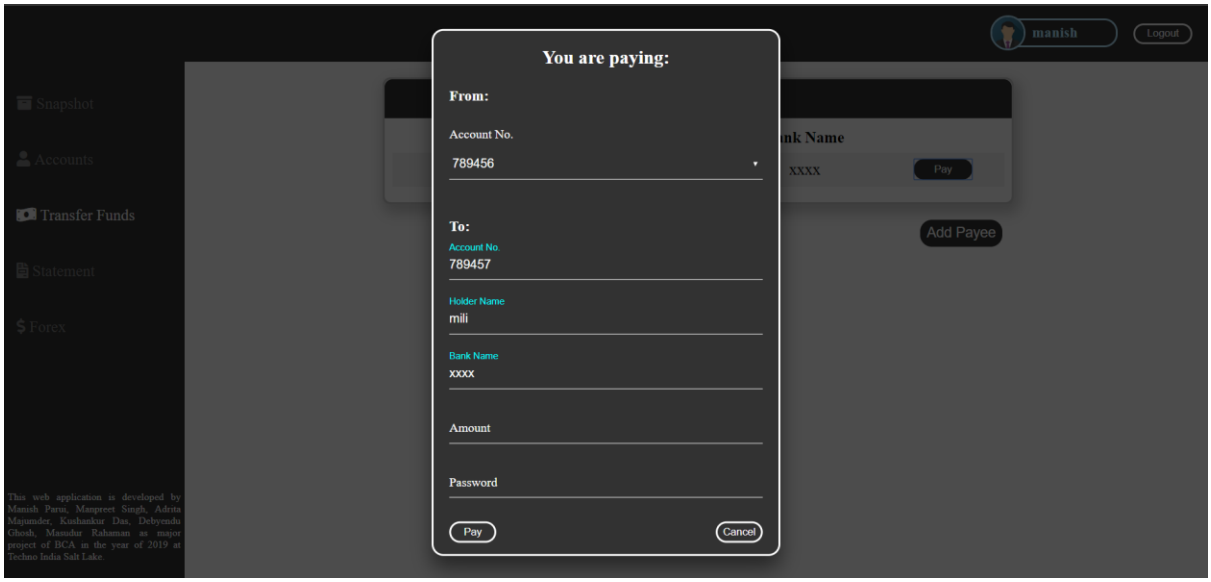

Interface 2.2.1



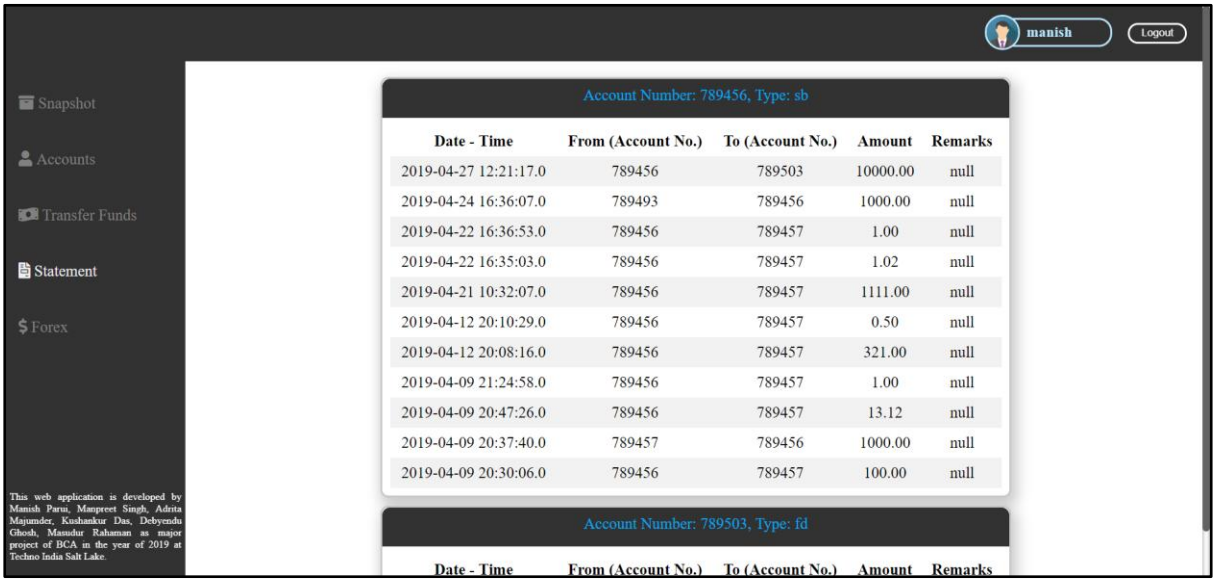

Interface 2.3.0




Interface 2.3.1

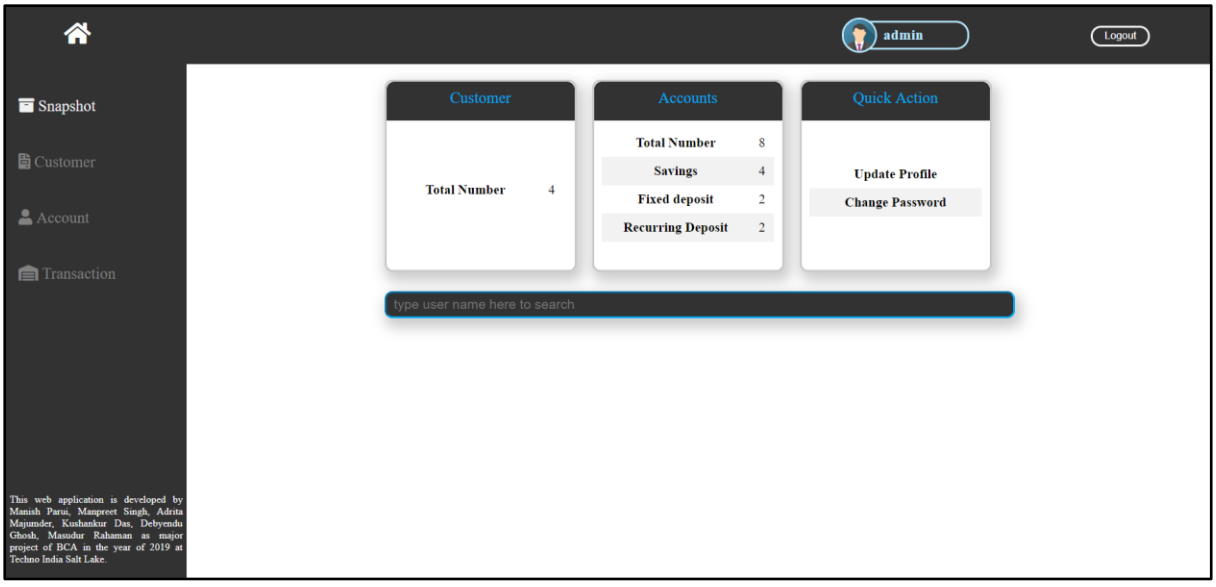



Interface 2.3.2

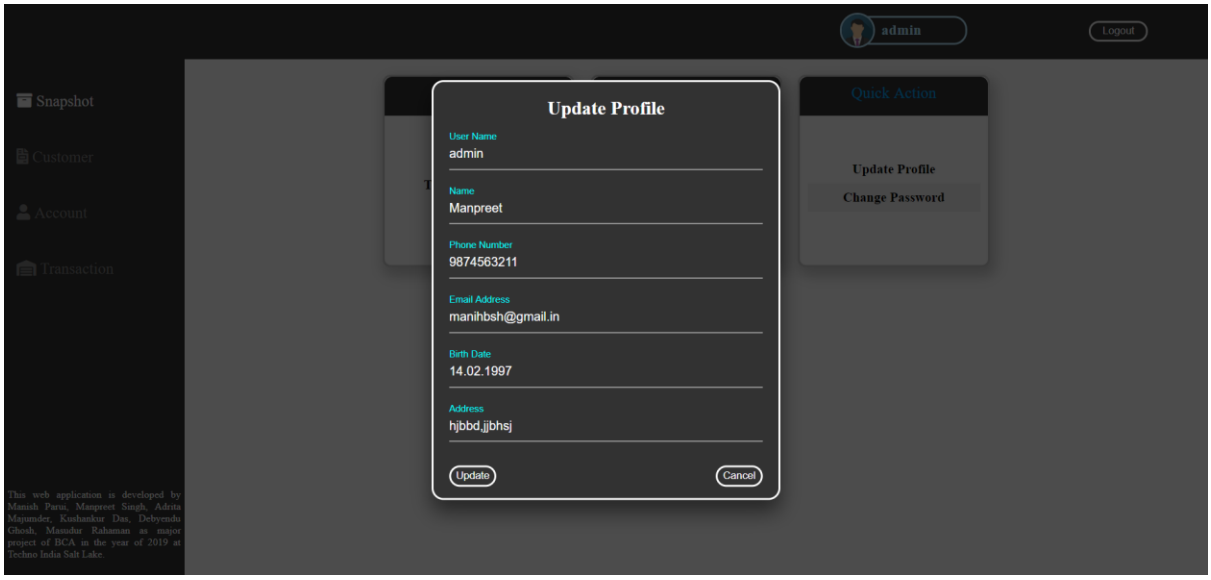



Interface 2.4.0

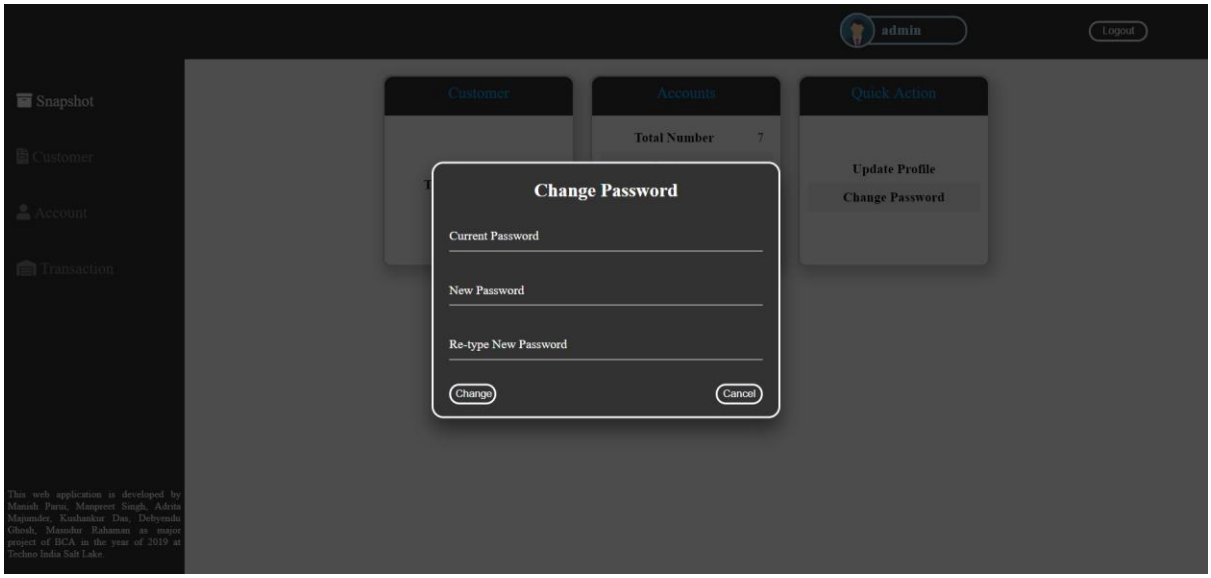
Interface 2.5.0



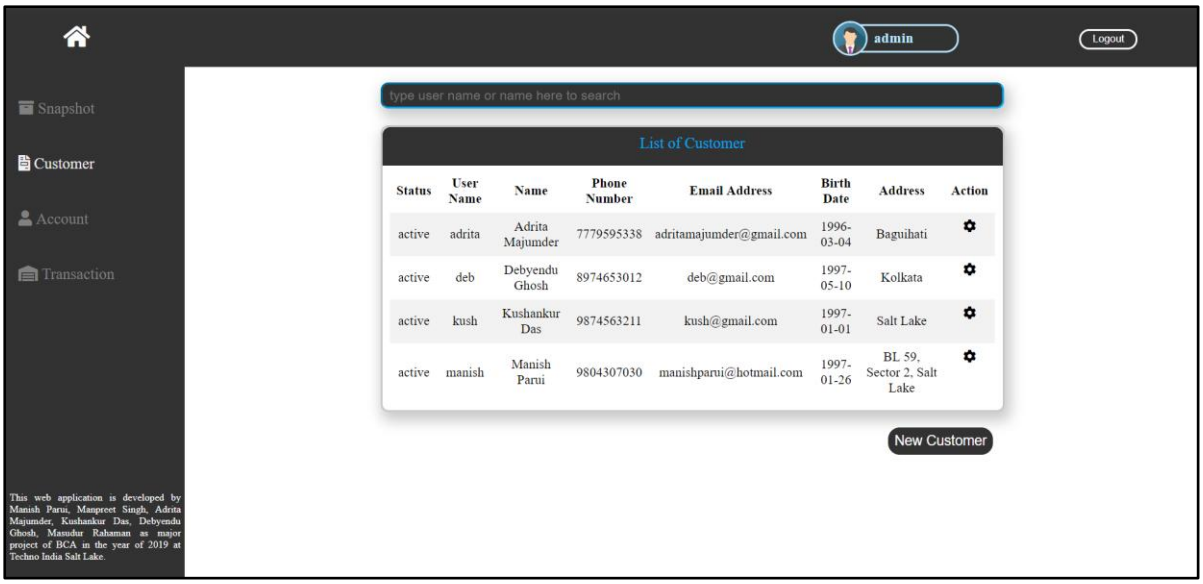

Interface 3.1.0



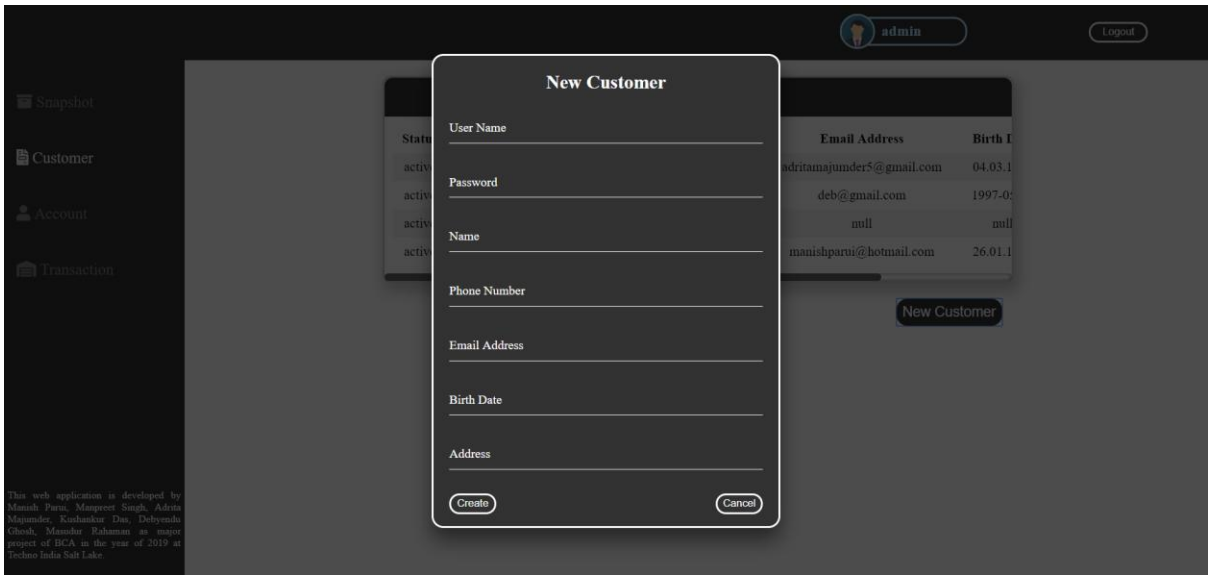

Interface 3.1.1



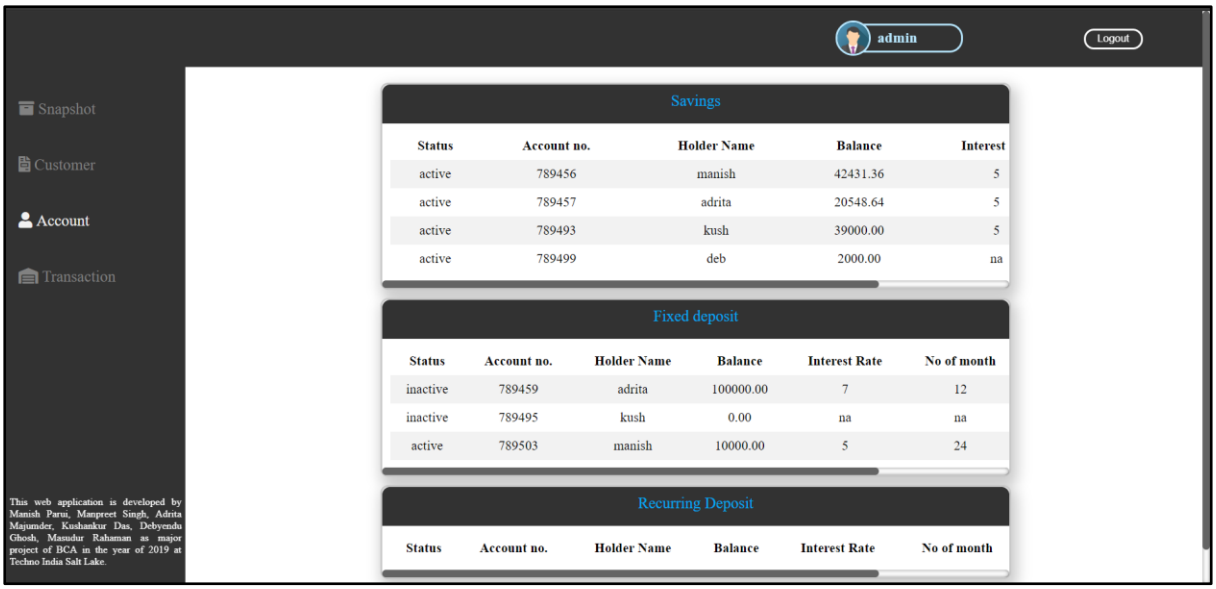

Interface 3.1.2



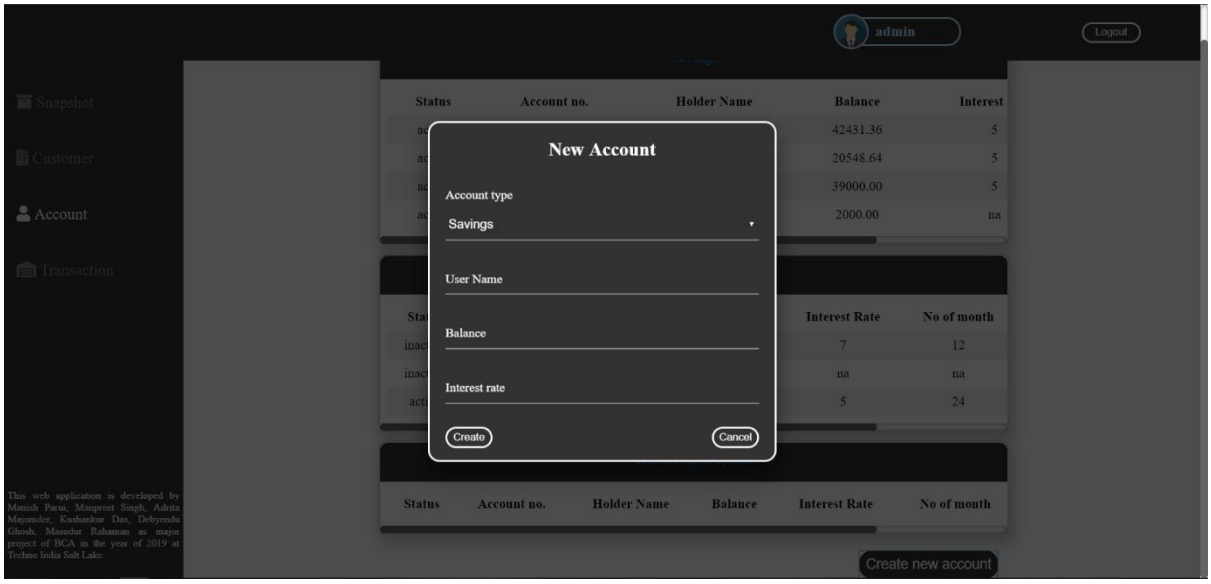

Interface 3.2.0



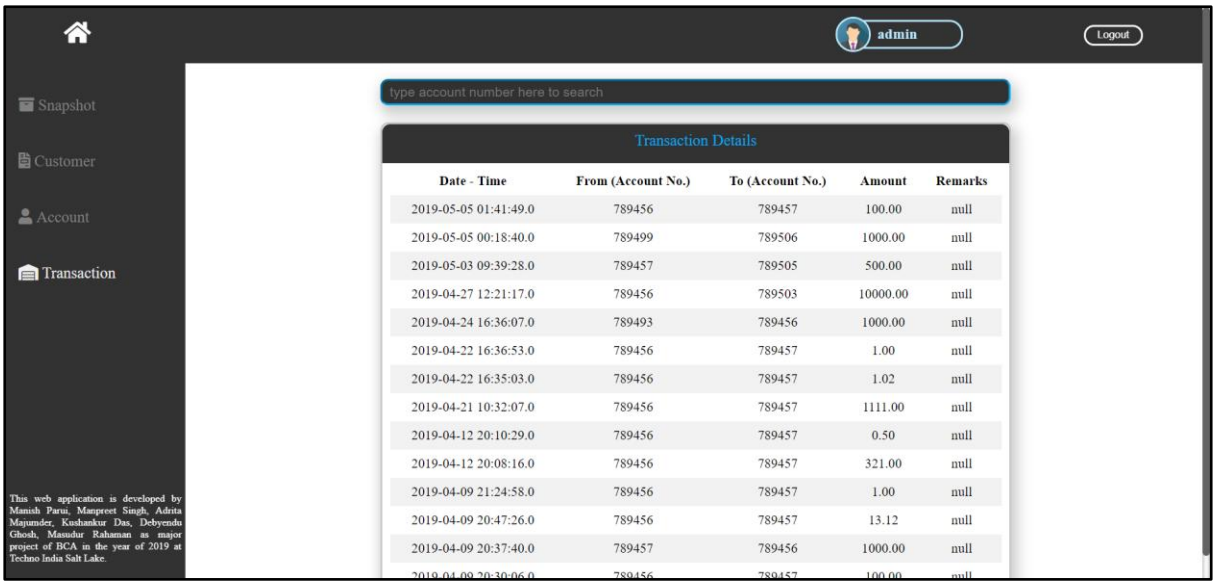

Interface 3.2.1




Interface 3.3.0




Interface 3.3.1




Interface 3.4.0

# 5.3 Procedural Design

Class name: AccountDAO

Purpose:Provides access to IB's MySQL database's Account table

Member function:

1. Name of the function: latestAccno
Argument list: String, String
Return type: String
Purpose: retrieve last created account number.
Algorithm: NA

2. Name of the function: isActive
Argument list: String
Return type: String
Purpose: retrieve account status.
Algorithm: NA

3. Name of the function: deactiveOne
Argument list: String,
Return type: void
Purpose: deactivated account
Algorithm: NA

4. Name of the function: retrieveData
Argument list: String, String
Return type: ResultSet
Purpose: retrieve account information.
Algorithm: NA

5. Name of the function: retrieveBalance
Argument list: String
Return type: float
Purpose: retrieve account balance.
Algorithm: NA

6. Name of the function: insertData
Argument list: String, String, float, float, String
Return type: void
Purpose: create new account.
Algorithm: NA

7. Name of the function: updateBalance

Argument list: String, float
Return type: void
Purpose: update account balance
Algorithm: NA

Class name: AdminDAO

Purpose:Provides access to IB's MySQL database's Admin table

Member function:

1. Name of the function: retrieveData
Argument list: NA
Return type: ResultSet
Purpose: retrieve admin information.
Algorithm: NA

2. Name of the function: updateData
Argument list: String, String, String,String, String, String
Return type: void
Purpose: update admin details.
Algorithm: NA

Class name: CustomerDAO

Purpose:Provides access to IB's MySQL database's Customer table

Member function:

1. Name of the function: deactivate
Argument list: String
Return type: void
Purpose: deactivate user profile.
Algorithm: NA

2. Name of the function: insertData
Argument list: String, String, String, String, String, String
Return type: void
Purpose: create new user profile
Algorithm: NA

3. Name of the function: retrieveData
Argument list: NA
Return type: ResultSet
Purpose: retrieve customer information.
Algorithm: NA

4. Name of the function: updateData
Argument list: String, String, String, String, String, String
Return type: String
Purpose: update customer profile
Algorithm: NA

Class name: LoginDAO

Purpose:Provides access to IB's MySQL database's Login table

Member function:

1. Name of the function:retrieveData
Argument list: String
Return type: ResultSet
Purpose: retrieve login details
Algorithm: NA

2. Name of the function: insertData
Argument list: String, String
Return type: void
Purpose: create new login.
Algorithm: NA

3. Name of the function: updatePassword
Argument list: String, String
Return type: void
Purpose: update password
Algorithm: NA

Class name: PayeeDAO

Purpose:Provides access to IB's MySQL database's Payee table

Member function:

1. Name of the function:retrieveData
Argument list: String
Return type: ResultSet
Purpose: retrieve payee information
Algorithm: NA

2. Name of the function: insertData
Argument list: String, String, String
Return type: void
Purpose: create new payee

Algorithm: NA

Class name: TransactionDAO

Purpose:Provides access to IB's MySQL database's Transaction table

Member function:

1. Name of the function:retrieveData
Argument list: String
Return type: ResultSet
Purpose: retrieve transaction information
Algorithm: NA

2. Name of the function: insertData
Argument list: String, String, String
Return type: void
Purpose: insert transaction information
Algorithm: NA

Class name: MySQLConnector

Purpose: provide connection to database.

Member function:

1. Name of the function: Connector
Argument list: NA
Return type: Connection
Purpose: provide connection to database.
Algorithm: NA

Class name: Servlet

Purpose: retrieve information, process them and send response.

Member function:

1. Name of the function: doPost
Argument list: HttpServletRequest, HttpServletResponse
Return type: void
Purpose: retrieve information, process them and send response.
Algorithm: NA

**Note:** *above class (Class name: Servlet) is a generic procedural design of a servlet which can be applied to any other servlet.*

# 6. CODING STANDARD FOLLOWED AND ASSUMPTIONS

## CODING STANDARD

The main goal of the recommendation is to improve readability and thereby the understanding and the maintainability and the general quality of the code. It is impossible to cover all the specific cases in a general guide and the programmer should be flexible.

## NAMING CONVENTION

Names representing packages should be in all lower case mypackage, com.company.application.ui
Package naming convention used by Sun for the Java core packages. The initial package name representing the domain name must be in lower case.

Names representing types must be nouns and written in mixed case starting with upper case
Line, Audio System Common practice in the Java development community and also the type naming convention used by Sun for the Java core packages.

**Variable names must be in mixed case starting with lower case** Makes variables easy to distinguish from types, and effectively resolves potential naming collision as in the declaration Line.

**Names representing constants (final variables) must be all uppercase using underscore to separate words**
e.g. MAX_ITERATIONS, COLOR_RED

Names representing methods must be verbs and written in mixed case starting with lower case getName(), computeTotalWidth()
Common practice in the Java development community and also the naming convention used by Sun for the Java core packages. This is identical to variable names, but methods in Java are already distinguishable from variables by their specific form.
Private class variables should have underscore suffix class
Person {private String name_; ...}
Apart from its name and its type, the scope of a variable is its most important feature. Indicating class scope by using underscore makes it easy to distinguish class variables

from local scratch variables. This is important because class variables are considered to have higher significance than method variables, and should be treated with special care by the programmer.
A side effect of the underscore naming convention is that it nicely resolves the problem of finding reasonable variable names for setter methods: void setName(String name){ name_ = name;
}
An issue is whether the underscore should be added as a prefix or as a suffix. Both practices are commonly used, but the latter is recommended because it seems to best preserve the readability of the name.

**Generic variables should have the same name as their type** void setTopic(Topic topic) // NOT: void setTopic(Topic value)
// NOT: void setTopic(Topic aTopic) //
NOT: void setTopic(Topic t)
void connect(Database database) // NOT: void connect(Database db)
// NOT: void connect(Database oracleDB)
It reduces complexity by reducing the number of terms and names used. Also makes it easy to deduce the type given a variable name only.

Variables with a large scope should have long names; variables with a small scope can have short names
Scratch variables used for temporary storage or indices are best kept short. A programmer reading such variables should be able to assume that its value is not used outside a few lines of code. Common scratch variables for integers are i, j, k, m, n and for characters' c and d.

**The terms get/set must be used where an attribute is accessed directly** employee.getName(); employee.setName(name); matrix.getElement(2, 4); matrix.setElement(2, 4, value);
Common practice in the Java community and the convention used by Sun for the Java core packages.

The is prefix should be used for boolean variables and methods isSet, isVisible, isFinished, isFound, isOpen
This is the naming convention for boolean methods and variables used by Sun for the Java core packages. Using the is prefix solves a common problem of choosing bad Boolean names like status or flag. isStatus or isFlag simply doesn't fit, and the programmer is forced to chose more meaningful names.

**The term find can be used in methods where something is looked up** vertex.findNearestVertex(); matrix.findSmallestElement();
node.findShortestPath(Node destinationNode);

Give the reader the immediate clue that this is a simple look up method with a minimum of computations involved. Consistent use of the term enhances readability. The n prefix should be used for variables representing a number of objects. E.g. nPoints, nLines etc.

## The Iterator variables should be called i, j, k etc.

for (Iterator i = points.iterator(); i.hasNext(); ) { : }
for (int i = 0; i < nTables; i++) { : }
The notation is taken from mathematics where it is an established convention for indicating iterators. Variables named j, k etc. should be used for nested loops only.

Negated boolean variable names must be avoided
bool isError; // NOT: isNoError bool isFound;
// NOT: isNotFound
The problem arise when the logical not operator is used and double negative arises. It is not immediately apparent what !isNotError means.

User Defined Exception classes should be suffixed with Exception class
AccessException extends Exception { : }
Exception classes are really not part of the main design of the program, and naming them like this makes them stand out relative to the other classes. This standard is followed by Sun in the basic Java library.

Functions (methods returning an object) should be named after what they return and procedures (void methods) after what they do
Increase readability. Makes it clear what the unit should do and especially all the things it is not supposed to do. This again makes it easier to keep the code clean of side effects.

## Java source files should have the extension .java

Special characters like TAB and page break must be avoided
These characters are bound to cause problem for editors, printers, terminal emulators or debuggers when used in a multi-programmer, multi-platform environment.

Variables should be initialized where they are declared and they should be declared in the smallest scope possible
This ensures that variables are valid at any time. Sometimes it is impossible to initialize a variable to a valid value where it is declared. In these cases it should be left uninitialized rather than initialized to some phony value.

Variables must never have dual meaning
Enhances readability by ensuring all concepts are represented uniquely. Reduce chance of error by side effects.

Floating point constants should always be written with a digit before the decimal point double total = 0.5; // NOT: double total = .5;
The 0.5 is a lot more readable than .5; there is no way it can be mixed with the integer.

## Arrays should be declared with their brackets next to the type

double[] vertex; // NOT: double vertex[]; int[] count; // NOT: int count[]; public static void main(String[] arguments) public double[] computeVertex()
The reason for is twofold. First, the array-ness is a feature of the class, not the variable. Second, when returning an array from a method, it is not possible to have the brackets with other than the type (as shown in the last example).

# 6. TESTING

| Module | Input | Output | Test Condition | Result |
|---|---|---|---|---|
| Login | Enter Login Id and Password | Logging into Customer or Admin Mode if Id or Password is correct or showing an error message: "Wrong Id or Password" | If the application able to verify the password from database if correct then redirect the page to home or admin-home according to account type else prevent user from login. | Passed |
| Register (CUSTOMER) | Enter Name, Phone Number, Address and other details to sign up for the banking as per procedure. | Account will be crated if each and every detail is given correctly and user name is available else an error message is displayed. | If the application able to check each and every details provided by the customer are correct in every aspect and user name is available else prevent user from register. | Passed |
| Snapshots (CUSTOMER)<br>1. Update Profile<br>2. Change Password | 1. Name, Phone Number, Email Address, Birth Date, Address.<br><br>2. Current Password, New Password. | 1. Profile is updated if given Phone Number, Email Address, Birth Date are correct then profile is updated else prevent user from updating.<br><br>2. Password is updated if the current password matches. | 1. If the application able to detect the error on specific fields like: Phone Number, Email address, Birth Date, or not<br>2. If the application able to update password while current password matched or not. | 1. Passed<br>2. Passed |
| Accounts (CUSTOMER)<br>3. Close account<br>4. Create New account | 1. No Input required<br>2. Account type, Number of Month, Amount | 1. Account is closed except Saving account.<br>2. Account created if customer's saving account have sufficient balance else | 1. If the application able to close the account except saving account.<br>2. If the application able to create new account if there are sufficient balance | 1. Passed<br>2. Passed |

| | | | | |
|---|---|---|---|---|
| | | prevent customer from creating one. | on customer saving account. | |
| Transfer Fund (CUSTOMER)<br>1. Pay<br>2. Add Payee | 1. Amount, Password.<br>2. Account number, holder name. | 1. Customer can able to transfer fund if they have sufficient balance on their saving account and password is correct.<br>2. Payee added if the account number is correct. | 1. If the application able to transfer fund if there are sufficient balance on customer's saving account and password is correct else prevent user from transfer fund.<br>2. If the application able to add payee or not | 1. Passed<br>2. Passed |
| Snapshots (ADMIN)<br>1. Update Profile<br>2. Change password | 1.Name, Phone Number, Email Address, Birth Date, Address.<br>2.Current Password, New Password. | 1. Profile is updated if given Phone Number, Email Address, Birth Date are correct then profile is updated else prevent user from updating.<br>2. Password is updated if the current password matches. | 1. If the application able to detect the error on specific fields like: Phone Number, Email address, Birth Date, or not<br>2. If the application able to update password while current password matched or not. | 1.Passed<br>2. Passed |
| Customer (ADMIN) New customer profile creation | User name, Password, Name, Phone Number, Email Address, Birth Date, Address. | New user profile is created if the user name entered by the user is not hold by anyone else, else prevent user from creating user profile. | If the application is able to create new customer profile or prevent user from create new customer profile or not. | Passed |
| Accounts (ADMIN) New Account creation | Account type, User name, Balance (only for savings), Interest rate, Amount (only for Fixed and recurring deposit), No of Month (only for Fixed and recurring deposit). | Account created if details which are provided by user are correct else prevent user from creating account. | If the application is able to create new account or prevent user from create new account or not. | Passed |

# 8. FUTURE SCOPE OF THE PROJECT

The followings are just a sample of future opportunities that could be implemented:

1. This project can be updated in future to provide loan features.
2. Virtual cards services can be implemented in future.
3. This project can be updated in future to provide bills payment functionality.
4. Changing of password can be done using user phone number or email address.
5. We can implement 2-factor authentication for login which can bring more security.
6. We can include real-time time chat services to improve user experience and fix their problem instantly.
7. This project can be updated in future to provide multi-currency transaction.

# 9. CONCLUSION

Banking systems have been with us for as long as people have been using money. Banks and other financial institutions provide security for individuals, businesses and governments, alike.

In general, what banks do is pretty easy to figure out. For the average person banks accept deposits, make loans, provide a safe place for money and valuables, and act as payment agents between merchants and banks.

Banks are quite important to the economy and are involved in such economic activities as issuing money, settling payments, credit intermediation, maturity transformation and money creation in the form of fractional reserve banking.

Now whereas in point of online banking system it is an electronic payment system that gives flexibility to many users.It also reduces work stress of a banker. Internet banking software provides personal and corporate banking services offering features such as viewing accountbalances, obtaining statements, checking recent transaction and making payments. Access is usually through a secure web site using a username and password, but security is a key consideration in internet banking and many banks also offer two factor authentication using a username and password feature.

# 10. REFERENCES and BIBLOGRAPHY

https://www.w3schools.com/html/default.asp

https://www.w3schools.com/css/default.asp

https://www.w3schools.com/js/default.asp

https://www.javatpoint.com/java-tutorial

https://www.tutorialspoint.com/html_online_training/index.asp

https://www.tutorialspoint.com/css_online_training/index.asp

https://en.wikipedia.org/wiki/Cascading_Style_Sheets

# 11.APPENDIX

**Index page (index.jsp)**

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
   pageEncoding="ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
	<link rel="stylesheet" type="text/css" href="index/index.css">
	<link rel="stylesheet" type="text/css" href="index/masud-login.css">
	<link rel="stylesheet" type="text/css" href="index/pur.css">
	<link rel="stylesheet" type="text/css" href="common/customscrollbar.css">
	<link rel="stylesheet" type="text/css" href="index/snackbar.css">
	<link rel="stylesheet" type="text/css" href="common/fontawesome-free-5.7.2-web/css/all.min.css">
	<script type="text/javascript" src="index/snackbar.js"></script>
	<script src="index/index.js"></script>
	<meta name="viewport" content="width=device-width, initial-scale=1">
	<title>welcome</title>
</head>
<body>
	<!--header-->
	<div class="header">
		<table class="menu">
			<tr>
				<td class="logo"></td>
				<td class="menu-item"><a href="">Explore Product</a></td>
				<td class="menu-item"><a href="">Make Payment</a></td>
				<td class="menu-item"><a href="">Personal Banking</a></td>
				<td class="menu-item" onclick="regctnr()">Register</td>
				<td class="login-btn-ctnr">
					<button class="btn" onclick="loginctnr()">Login</button>
				</td>
			</tr>
		</table>
	</div>
```

```
<!--header-end-->
<!--popup-login-->
<div id="pul" class="pul">
	<div class="login-ctnr">
		<h2>Login</h2>
		<form action="LoginServ" method="post">
			<div class="inputbox">
				<input type="text" name="uname" required>
				<label>User Name</label>
			</div>
			<div class="inputbox">
				<input type="Password" name="pass" required>
				<label>Password</label>
			</div>
			<input class="btn btn-login" type="Submit" value="Login">
			<button class="btn btn-cancel"
onclick="loginctnrcancel()">Cancel</button>
		</form>
	</div>
</div>
<!--popup-login-end-->
<!--popup-registration-->
<div id="pur" class="pur">
	<div class="pur-ctnr">
		<h2>New Account Registration</h2>
		<form action="RegServ" method="post">
			<div class="inputbox">
				<input type="text" name="regUname" placeholder="i.e. jhon"
required>
				<label>User Name</label>
			</div>
			<div class="inputbox">
				<input type="Password" name="regPass" required>
				<label>Password</label>
			</div>
			<div class="inputbox">
```

```
                                        <input type="text" name="regName" placeholder="i.e. Jhon Doe"
required>
                                        <label>Name</label>
                                </div>
                                <div class="inputbox">
                                        <input type="number" name="regPhno" placeholder="i.e.
9874563210" min="6000000000" max="9999999999" required>
                                        <label>Phone Number</label>
                                </div>
                                <div class="inputbox">
                                        <input type="email" name="regEmail" placeholder="i.e.
jhon@mail.com" required>
                                        <label>Email Address</label>
                                </div>
                                <div class="inputbox">
                                        <input type="date" name="regDob" placeholder="i.e. 26-01-1997"
required>
                                        <label>Birth Date</label>
                                </div>
                                <div class="inputbox">
                                        <input type="text" name="regAddress" placeholder="i.e. Address
line 1, address line 2" required>
                                        <label>Address</label>
                                </div>
                                <input class="purbtn btn-register" type="Submit" name=""
value="Register">
                                <button class="purbtn btn-cancel"
onclick="regctnrcancel()">Cancel</button>
                        </form>
                </div>
        </div>
        <!--popup-registration-end-->
        <!--social-link-->
        <div class="social-link">
                <a href="https://www.facebook.com/"><i class="fab fa-facebook-square social-link-
i"></i></a>
                <a href="https://twitter.com/"><i class="fab fa-twitter-square social-link-i"></i></a>
                <a href="https://www.instagram.com/"><i class="fab fa-instagram social-link-i"></i></a>
```

```
                <a href="https://www.linkedin.com/"><i class="fab fa-linkedin social-link-i"></i></a>

                <a href=""><i class="fab fa-google-play social-link-i"></i></a>
        </div>
        <!--social-link-end-->
        <!--main-->
        <div class="body">
                <!--banner-->
                <div class="inbk">
                        <div class="banner-login-btn-ctnr">
                                <h1>Internet Banking</h1>
                                <p>Banking at your finger tip</p>
                                <button class="banner-btn" onclick="loginctnr()">Login</button>
                                <button class="banner-btn" onclick="regctnr()">Register</button>
                        </div>
                </div>
                <!--banner-end-->
        </div>
        <!--main-end-->
</body>
</html>
```

**Home page (home.jsp)**

```
<%@page import="DAO.CustomerDAO"%>
<%@page import="DAO.TransactionDAO"%>
<%@page import="DAO.PayeeDAO"%>
<%@page import="DAO.AccountsDAO"%>
<%@page import="java.sql.ResultSet"%>
<%@page import="java.sql.SQLException"%>
<%@page import="java.sql.PreparedStatement"%>
<%@page import="javax.websocket.Session"%>
<%@page import="nclass.MySQLConnector"%>
<%@page import="java.sql.Connection"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
   pageEncoding="ISO-8859-1"%>
<%
```

```
		response.setHeader("Cache-Control", "no-cache, no-store, must-revalidate");
		response.setHeader("Pragma", "no-cache");
		response.setHeader("Expires", "0");
		 if(session.getAttribute("uname") == null){
			response.sendRedirect("index.jsp");
		} else if(session.getAttribute("type").equals("a")) {
			session.removeAttribute("uname");
			session.invalidate();
			response.sendRedirect("index.jsp");
		} else {
%>
<%
		String uname = session.getAttribute("uname").toString();
		Connection con = null;
		AccountsDAO adao = new AccountsDAO();
		TransactionDAO tdao = new TransactionDAO();
		CustomerDAO cdao = new CustomerDAO();
%>
<!DOCTYPE html>
<html>
<head>
		<link rel="stylesheet" type="text/css" href="home/home.css">
		<link rel="stylesheet" type="text/css" href="common/messagebox.css">
		<link rel="stylesheet" type="text/css" href="home/comingsoon.css">
		<link rel="stylesheet" type="text/css" href="home/popup.css">
		<link rel="stylesheet" type="text/css" href="home/hovermenu.css">
		<link rel="stylesheet" type="text/css" href="common/customscrollbar.css">
		<link rel="stylesheet" type="text/css" href="common/credit.css">
		<link rel="stylesheet" type="text/css" href="common/fontawesome-free-5.7.2-web/css/all.min.css">
		<script src="home/home.js"></script>
		<script src="home/comingsoon.js"></script>
		<script type="text/javascript" src="home/popup.js"></script>
		<script type="text/javascript" src="common/common.js"></script>

		<title>home</title>
</head>
```

```
<body onresize="mainLeft(); mainHeight()" onload="mainLeft(); mainHeight()">
	<!--header-->
	<div class="header">
		<div class="home click" id="homelogofa" onclick="home()"><i class="fas fa-home"></i></div>
		<div class="acc-div">
			<img src="common/resources/m_avatar.png">
			<h3><%=session.getAttribute("uname") %></h3>
		</div>
		<button id="logout-btn" class="logout-btn" onclick="ModalOverlay()">Logout</button>
	</div>
	<!--header-end-->
	<!--modal-overlay-logout-message-box-->
	<div id="modal-overlay" class="modal-overlay">
		<div class="messagebox">
			<p>Do you really wanna logout?</p><br>
			<form action="LogoutServ" method="post">
				<input class="messagebox-btn-left" type="Submit" value="Logout">
			</form>
			<button class="messagebox-btn-right" onclick="CloseModalOverlay()">Cancel</button>
		</div>
	</div>
	<!--modal-overlay-logout-message-box-end-->
	<!--sidebar-->
	<div id="sidebar" class="sidebar">
		<br>
		<br>
		<a href="#snapshot" id="snap" class="sidebar-a-temp" onclick="SnapshotVisible()"><i class="fas fa-archive"></i> Snapshot</a>
		<br>
		<br>
		<a href="#account" id="acc" class="sidebar-a" onclick="AccountsVisible()"><i class="fas fa-user"></i> Accounts</a>
		<br>
		<br>
		<a href="#tf" id="tff" class="sidebar-a" onclick="TFVisible()"><i class="fas fa-money-bill-wave"></i>
```

```
                Transfer Funds</a>
                <br>
                <br>
                <a href="#pb" id="pbb" class="sidebar-a" onclick="PBVisible()"><i class="fas fa-file-
invoice"></i> Statement</a>
                <br>
                <br>
                <a href="#forex" id="for" class="sidebar-a" onclick="ForexVisible()"><i class="fas fa-dollar-
sign"></i> Forex</a>
                <div class="cr">
                        <p id="cr" >This web application is developed by Kushankur Das major project of
BCA in the year of 2019 at Techno India Salt Lake.</p>
                </div>
        </div>
        <!--sidebar-end-->
        <!--main-->
        <!--snapshot-->
        <div id="main" class="main">
                <div id="snapshot" class="container-temp">
                        <div class="card last-trans">
                                <div class="card-header">
                                        Last Five Transaction
                                </div>
                                <div class="card-body last-trans-body">
                                        <table class="striped-table">
                                                <tr>
                                                        <th>Date Time</th>
                                                        <th>From</th>
                                                        <th>To</th>
                                                        <th>Amount</th>
                                                </tr>
                                                <%
                        ResultSet sbacnotrrs = adao.retrieveAcno(uname);
                                                        String acnotr = null;
                                                        while(sbacnotrrs.next()){
                                                                acnotr = sbacnotrrs.getString(1);
                                                                break;
```

```
                                        }
                                        sbacnotrrs.close();
                                        int x5 = tdao.countData(acnotr);
                                        if(x5 == 0){
                                    %>
                                            <tr><td colspan="4">You don't have
any transaction yet.</td></tr>
                                    <%
                                        } else{
                                ResultSet transrs = tdao.retrive5Data(acnotr);
                                            while(transrs.next()){
                                    %>
                                                <tr>

        <td><%=transrs.getString("datetime") %></td>

        <td><%=transrs.getString("from") %></td>

        <td><%=transrs.getString("to") %></td>

        <td><%=transrs.getString("amount") %></td>
                                                </tr>
                                    <%
                                            }
                                            transrs.close();
                                        }
                                    %>

                                </table>
                            </div>
                        </div>
                        <div class="card balance">
                            <div class="card-header">
                                Balance
                            </div>
                            <div class="card-body balance-body">
                                <%=adao.retrieveBalance(acnotr) %>
                            </div>
```

```
                </div>


                <div class="card quick-request">
                        <div class="card-header">
                                Quick Action
                        </div>
                        <div class="card-body quick-request-body">
                                <table class="striped-table">
                                        <tr><td class="click"
onclick="openUpdateProfile()">Update Profile</td></tr>
                                        <tr><td class="click"
onclick="openChangePassword()">Change Password</td></tr>
                                </table>
                        </div>
                </div>
        </div>
        <!--update-profile-->
        <table class="displaynone" id="updateProfileData">
                <tr>
        <%
                ResultSet updateprofrs = cdao.retrieveData(uname);
                while(updateprofrs.next()){
        %>
                        <td><%=updateprofrs.getString("holdername") %></td>
                        <td><%=updateprofrs.getString("phno") %></td>
                        <td><%=updateprofrs.getString("email") %></td>
                        <td><%=updateprofrs.getString("dob") %></td>
                        <td><%=updateprofrs.getString("address") %></td>
        <%
                        break;
                }
                updateprofrs.close();
        %>
                </tr>
        </table>
        <div id="updateprof" class="pop-overlay">
                <div class="pop">
```

```
                                <h2>Update Profile</h2>
                                <form action="UpdateProfileServ" method="post">
                                        <div class="inputbox">
                                                <input type="text" name="uprrrUname" readonly
value=<%=uname %>>
                                                <label>User Name</label>
                                        </div>
                                        <div class="inputbox">
                                                <input type="text" name="uprrrName" id="uprrrName"
required>
                                                <label>Name</label>
                                        </div>
                                        <div class="inputbox">
                                                <input type="number" name="uprrrPhno" id="uprrrPhno"
min="6000000000" max="9999999999" required>
                                                <label>Phone Number</label>
                                        </div>
                                        <div class="inputbox">
                                                <input type="email" name="uprrrEmail" id="uprrrEmail"
required>
                                                <label>Email Address</label>
                                        </div>
                                        <div class="inputbox">
                                                <input type="date" name="uprrrDob" id="uprrrDob"
required>
                                                <label>Birth Date</label>
                                        </div>
                                        <div class="inputbox">
                                                <input type="text" name="uprrrAddress"
id="uprrrAddress" required>
                                                <label>Address</label>
                                        </div>
                                        <input class="pop-btn-left" type="Submit" value="Update">
                                        <button type="button" class="pop-btn-right"
onclick="closeUpdateProfile()">Cancel</button>
                                </form>
                        </div>
                </div>
                <!--update-profile-end-->
```

```
<!--change-password-popup-->
<div id="chnagepass" class="pop-overlay">
	<div class="pop">
		<h2>Change Password</h2>
		<form action="ChangePassServ" method="post">
			<div class="inputbox">
				<input type="password" name="oldPass" required>
				<label>Current Password</label>
			</div>
			<div class="inputbox">
				<input type="password" name="newPass" required>
				<label>New Password</label>
			</div>
			<input type="hidden" name="cphjuname" value=<%=uname %>>
			<input class="pop-btn-left" type="Submit" value="Change">
			<button type="button" class="pop-btn-right"
onclick="closeChangePassword()">Cancel</button>
		</form>
	</div>
</div>
<!--change-password-end-->
<!--snapshot-end-->
<!--accounts-->
<div id="accounts" class="container">
	<div class="card accounts-lists">
		<div class="card-header">
			Savings Account
		</div>
		<div class="card-body accounts-lists-body">
			<table class="striped-table">
				<tr>
					<th>Status</th>
					<th>Account No.</th>
					<th>Balance</th>
					<th>Interest rate (%)</th>
					<th>Opening</th>
```

```
                                        <th>Setting</th>
                                    </tr>
                                    <%
                                        int sbcount = adao.countData("sb", uname);
                                        if(sbcount == 0){
                                    %>
                <tr><td colspan="6">You don't have any Savings Account.</td></tr>
                                    <%
                                        } else{
                                            ResultSet sbrs = adao.retriveData("sb",
uname);
                                            while (sbrs.next()){
                                    %>
                                            <tr>

            <td><%=sbrs.getString("status") %></td>
            <td><%=sbrs.getString("acno") %></td>
            <td><%=sbrs.getString("balance") %></td>
            <td><%=sbrs.getString("rate") %></td>
            <td><%=sbrs.getString("opening") %></td>
                                                <td class="tooltip">
                                                <i class="fas fa-cog
click"></i>
                                                <span
class="tooltiptext">
                                                    <span
class="click" onclick="openAccStatement()">

            View Statement
                                                    </span>
                                                </span>
                                            </td>
                                        </tr>
                                    <%      }
                                            sbrs.close();
                                        }
                                    %>
                                </table>
```

```
                </div>
            </div>
            <div class="card accounts-lists">
                <div class="card-header">
                    Fixed deposit
                </div>
                <div class="card-body accounts-lists-body">
                    <table class="striped-table">
                        <tr>
                            <th>Status</th>
                            <th>Account no.</th>
                            <th>Balance</th>
                            <th>Interest rate (%)</th>
                            <th>No. of month</th>
                            <th>Amount</th>
                            <th>Expiry</th>
                            <th>Setting</th>
                        </tr>
                        <%
                            int fdcount = adao.countData("fd", uname);
                            if(fdcount == 0){
                        %>
            <tr><td colspan="8">You don't have any Fixed deposit.</td></tr>
                        <%
                            } else{
                    ResultSet fdrs = adao.retriveData("fd", uname);
                                    while (fdrs.next()){
                        %>
                                    <tr>
<td><%=fdrs.getString("status") %></td>
<td><%=fdrs.getString("acno") %></td>
<td><%=fdrs.getString("balance") %></td>
<td><%=fdrs.getString("rate") %></td>
<td><%=fdrs.getString("nomonth") %></td>
<td><%=fdrs.getString("installment") %></td>
```

```
<td><%=fdrs.getString("expiry") %></td>
                                                                    <td class="tooltip">
                                                                        <i class="fas fa-cog
click"></i>
                                                                        <span
class="tooltiptext">
                                                                            <span
class="click" onclick="openAccStatement()">
View Statement
                                                                            </span>
                                                                            <%
if(fdrs.getString("status").equals("active")){
                                                                            %>
<br>
<hr>
<span class="click" onclick="openAccReqClose(this)">
Close Account
</span>
                                                                            <%      }
%>
                                                                        </span>
                                                                    </td>
                                                                </tr>
                                                    <%              }
                                                                fdrs.close();
                                                            }
                                                    %>
                                                </table>
                                            </div>
                                        </div>
                                        <div class="card accounts-lists">
                                            <div class="card-header">
                                                Recurring Deposit
```

```
                    </div>
                    <div class="card-body accounts-lists-body">
                        <table class="striped-table">
                            <tr>
                                <th>Status</th>
                                <th>Account no.</th>
                                <th>Balance</th>
                                <th>Interest rate (%)</th>
                                <th>No. of month</th>
                                <th>Installment amount</th>
                                <th>Expiry</th>
                                <th>Setting</th>
                            </tr>
                            <%
                                int rdcount = adao.countData("rd", uname);
                                if(rdcount == 0){
                            %>
                        <tr><td colspan="8">You don't have any Recurring Deposit.</td></tr>
                            <%
                                } else{
                                    ResultSet rdrs = adao.retriveData("rd",
uname);
                                    while (rdrs.next()){
                            %>
                                    <tr>

        <td><%=rdrs.getString("status") %></td>
        <td><%=rdrs.getString("acno") %></td>
        <td><%=rdrs.getString("balance") %></td>
        <td><%=rdrs.getString("rate") %></td>
        <td><%=rdrs.getString("nomonth") %></td>
        <td><%=rdrs.getString("installment") %></td>
        <td><%=rdrs.getString("expiry") %></td>
                                        <td class="tooltip">
                                    <i class="fas fa-cog click"></i>
                                        <span
class="tooltiptext">
```

```
                                                                    <span
class="click" onclick="openAccStatement()">
        View Statement
                                                                </span>
                                                                    <%
        if(rdrs.getString("status").equals("active")){
                                                                    %>
        <br>
        <hr>
        <span class="click" onclick="openAccReqClose(this)">
                Close Account
        </span>
                                                                    <%      }
%>
                                                                </span>
                                                            </td>
                                                        </tr>
                                                <%              }
                                                    }
                                                %>
                                            </table>
                                        </div>
                                    </div>
<button type="button" class="btn-ana" onclick="openNewAccountPopup()">Create new account</button>
                    </div>
                    <!--account-statement-->
                    <div class="pop-overlay" id="accstatement">
                            <div class="pop">
<button type="button" class="pop-btn-right" onclick="closeAccStatement()">Cancel</button>
                            </div>
                    </div>
                    <!--account-statement-end-->
```

```
<!--account-request-close-->
<div class="pop-overlay" id="accreqclose">
    <div class="pop">
        <p>Do you really want to close your account: <span
id="accreqcloseacno"></span>?</p>
        <form action="AccCloseServ" method="post">
            <input id="accreqcloseaccnojs" type="hidden"
name="accreqcloseaccno">
            <input class="pop-btn-left" type="submit" value="Yes">
        </form>
        <button type="button" class="pop-btn-right"
onclick="closeAccReqClose()">Cancel</button>
    </div>
</div>
<!--account-request-close-end-->
<!--new-account-popup-->
<div id="newaccountpopup" class="pop-overlay">
    <div class="pop">
        <h2>New Account</h2>
        <form action="NewAccServ" method="post">
            <div class="inputbox">
                <label>Type</label><br><br>
                <select name="type">
                    <option value="fd">Fixed deposit</option>
                    <option value="rd">Recurring Deposit</option>
                </select>
            </div>
            <div class="inputbox">
                <label>Number of month</label><br><br>
                <select name="newacnomonth">
                    <option value="6">6</option>
                    <option value="12">12</option>
                    <option value="18">18</option>
                    <option value="24">24</option>
                </select>
            </div>
            <div class="inputbox">
```

```
                                        <label>Amount</label><br><br>
                                <input type="number" name="newacamt" required>


                                    </div>
                                <input type="hidden" name="uname" value=<%=uname %>>
                            <input class="pop-btn-left" type="Submit" name="" value="Apply">
        <button type="button" class="pop-btn-right" onclick="closeNewAccountPopup()">Cancel</button>
                                    </form>
                            </div>
                    </div>
                    <!--new-account-popup-end-->
                    <!--accounts-end-->
                    <!--transfer-funds-->
                    <div id="tf" class="container">
                            <div class="card lop">
                                    <div class="card-header">
                                            List of Payee
                                    </div>
                                    <div class="card-body lop-body">
                                            <table id="payeetable" class="striped-table">
                                                    <tr>
                                                            <th>Account No.</th>
                                                            <th>Holder Name</th>
                                                            <th>Bank Name</th>
                                                            <th></th>
                                                    </tr>
                                                    <%
                                                            PayeeDAO pdao = new PayeeDAO();
                                                            int lopCount = pdao.countData(uname);
                                                            if(lopCount == 0){
                                                    %>
                                                                    <tr><td colspan="4">You don't have
any payee.</td></tr>
                                                    <%
                                                            } else{
                                            ResultSet loprs = pdao.retrieveData(uname);
```

```
                                                    while(loprs.next()){
                                                %>
                                                        <tr
onclick="openPopUp(this)">
        <td><%=loprs.getString("acno")%></td>
        <td><%=loprs.getString("holdername")%></td>
        <td><%=loprs.getString("bankname")%></td>
                                                                <td><button
class="btn-lop-pay click">Pay</button></td>
                                                        </tr>
                                                <%
                                                    }
                                                }
                                                %>
                                        </table>
                                    </div>
                                </div>
        <button type="button" class="btn-ana" onclick="openPayeePopup()">Add Payee</button>
                </div>

                <!--add-payee-popup-->
                <div id="payeepopup" class="pop-overlay">
                        <div class="pop">
                                <h2>New Payee</h2>
                                <form action="AddPayeeServ" method="post">
                                        <div class="inputbox">
                                                <input type="text" name="payeeacno" required>
                                                <label>Account No.</label>
                                        </div>
                                        <div class="inputbox">
                                                <input type="text" name="payeehname" required>
                                                <label>Holader Name</label>
                                        </div>
                                        <div class="inputbox">
                                                <input type="text" name="payeebname" value="NA">
                                                <label>Bank Name</label>
```

```
                                        </div>
                                        <input type="hidden" name="payeeaddby" value=<%=uname %>>
                                        <input class="pop-btn-left" type="Submit" name="" value="Add">
<button type="button" class="pop-btn-right" onclick="closePayeePopup()">Cancel</button>
                                </form>
                        </div>
                </div>
                <!--add-payee-popup-end-->
                <!--transfer-funds-popup-->
                <div id="popup" class="pop-overlay">
                        <div class="pop">
                                <h2>You are paying:</h2>
                                <form action="FundTransferServ" method="post">
                                <h3 class="h3">From:</h3>
                                        <div class="inputbox">
                                                <label>Account No.</label><br><br>
                                                <select name="hsbacno">
                                                <%
                                                        ResultSet tfrs = adao.retrieveAcno(uname);
                                                        while(tfrs.next()){
                                                                String acnotff = tfrs.getString("acno");
                                                %>
    <option value=<%= tfrs.getString("acno") %>><%= tfrs.getString("acno") %></option>
                                                <%
                                                        }
                                                %>
                                                </select>
                                        </div>
                                        <h3 class="h3">To:</h3>
                                        <div class="inputbox">
                                                <input id="pacno" type="text" name="pacno" readonly>
                                                <label>Account No.</label>
                                        </div>
                                        <div class="inputbox">
                        <input id="phname" type="text" name="phname" readonly>
                                                <label>Holder Name</label>
```

```
                                        </div>
                                        <div class="inputbox">
                        <input id="pbname" type="text" name="pbname" readonly>
                                                <label>Bank Name</label>
                                        </div>
                                        <div class="inputbox">
                        <input id="pamount" type="text" name="pamount" required>
                                                <label>Amount</label>
                                        </div>
                                        <div class="inputbox">
                        <input id="ppass" type="Password" name="ppass" required>
                                                <label>Password</label>
                                        </div>
                                        <input type="hidden" name="puname" value=<%=uname %>>
                                        <input class="pop-btn-left" type="Submit" name="" value="Pay">
<button type="button" class="pop-btn-right" onclick="closePopUp()">Cancel</button>
                                </form>
                        </div>
        </div>
        <!--transfer-funds-popup-end-->
        <!--transfer-funds-end-->
        <!--statement-->
        <div id="pb" class="container">
        <%
                ResultSet allrs = adao.retrieveAllAcno(uname);
                while(allrs.next()){
                        String acno = allrs.getString("acno");
                        String type = allrs.getString("type");
        %>
                        <div class="card stat" id="<%=acno %>">
                                <div class="card-header">
                                        Account Number: <%=acno %>, Type: <%=type %>
                                </div>
                                <div class="card-body stat-body">
                                        <table class="striped-table">
                                                <tr>
```

```
                                        <th>Date - Time</th>
                                        <th>From (Account No.)</th>
                                        <th>To (Account No.)</th>
                                        <th>Amount</th>
                                        <th>Remarks</th>
                                    </tr>
                                    <%
                                        int x = tdao.countData(acno);
                                        if(x == 0){
                                    %>
<tr><td colspan="5">You don't have any transaction.</td></tr>
                                    <%
                                        } else{
ResultSet acnors = tdao.retriveData(acno);
                                            while(acnors.next()){
                                    %>
                                    <tr >
<td><%=acnors.getString("datetime") %></td>
<td><%=acnors.getString("from") %></td>
<td><%=acnors.getString("to") %></td>
<td><%=acnors.getString("amount") %></td>
<td><%=acnors.getString("remarks") %></td>
                                    </tr>
                                    <%
                                            }
                                            acnors.close();
                                        }
                                    %>
                                </table>
                            </div>
                        </div>
<%
        }
        allrs.close();
%>
</div>
```

```
                <!--statement-end-->
                <!--forex-->
                <div id="forex" class="container">
                        <div class="cs-middle">
                                <h1>COMING SOON</h1>
                                <p id="demo" style="font-size:30px"></p>
                        </div>
                </div>
                <!--forex-end-->
        </div>
        <!--main-end-->
        <script type="text/javascript">CusReload();</script>
        <script type="text/javascript">cr();</script>
</body>
</html>
<%}%>
```

**Admin home page(admin.jsp)**

```
<%@page import="DAO.LoginDAO"%>
<%@page import="DAO.AdminDAO"%>
<%@page import="DAO.TransactionDAO"%>
<%@page import="DAO.AccountsDAO"%>
<%@page import="java.sql.ResultSet"%>
<%@page import="DAO.CustomerDAO"%>
<%@page import="java.sql.Connection"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"    pageEncoding="ISO-8859-1"%>
response.setHeader("Cache-Control", "no-cache, no-store, must-revalidate");
        response.setHeader("Pragma", "no-cache");
        response.setHeader("Expires", "0");
          if(session.getAttribute("uname") == null){
                response.sendRedirect("index.jsp");
        } else if(session.getAttribute("type").equals("c")) {
                session.removeAttribute("uname");
                session.invalidate();
                response.sendRedirect("index.jsp");
        } else {
```

```
%>
<%
	String uname = session.getAttribute("uname").toString();
	Connection con = null;
	CustomerDAO cdao = new CustomerDAO();
	AccountsDAO adao = new AccountsDAO();
	TransactionDAO tdao = new TransactionDAO();
	AdminDAO amdao = new AdminDAO();
	LoginDAO ldao = new LoginDAO();
%>
<!DOCTYPE html>
<html>
<head>
	<link rel="stylesheet" type="text/css" href="admin/admin.css">
	<link rel="stylesheet" type="text/css" href="common/messagebox.css">
	<link rel="stylesheet" type="text/css" href="admin/comingsoon.css">
	<link rel="stylesheet" type="text/css" href="common/customscrollbar.css">
	<link rel="stylesheet" type="text/css" href="admin/popup.css">
	<link rel="stylesheet" type="text/css" href="admin/hovermenu.css">
	<link rel="stylesheet" type="text/css" href="common/credit.css">
	<link rel="stylesheet" type="text/css" href="common/fontawesome-free-5.7.2-web/css/all.min.css">
	<script type="text/javascript" src="admin/admin.js"></script>
	<script type="text/javascript" src="admin/popup.js"></script>
	<script type="text/javascript" src="admin/search.js"></script>
	<script type="text/javascript" src="common/common.js"></script>
	<title>admin home</title>
</head>
<body onresize="mainLeft(); mainHeight()" onload="mainLeft(); mainHeight()">
	<!--header-->
	<div class="header">
<div class="home click" id="homelogofa" onclick="home()"><i class="fas fa-home"></i></div>
		<div class="acc-div">
			<img src="common/resources/m_avatar.png">
			<h3><%= session.getAttribute("uname") %></h3>
		</div>
		<button id="logout-btn" class="logout-btn" onclick="ModalOverlay()">Logout</button>
```

```
</div>
<!--header-end-->
<!--modal-overlay-logout-message-box-->
<div id="modal-overlay" class="modal-overlay">
	<div class="messagebox">
		<p>Do you really wanna logout?</p><br>
		<form action="LogoutServ" method="post">
			<input class="messagebox-btn-left" type="Submit" value="Logout">
		</form>
<button class="messagebox-btn-right" onclick="CloseModalOverlay()">Cancel</button>
	</div>
</div>
<!--modal-overlay-logout-message-box-end--><!--sidebar-->
<div id="sidebar" class="sidebar">
	<br>
	<br>
	<a href="#snapshot" id="snap" class="sidebar-a-temp" onclick="SnapshotVisible()"><i class="fas fa-archive"></i> Snapshot</a>
	<br>
	<br>
	<a href="#customer" id="cus" class="sidebar-a" onclick="CustomerVisible()"><i class="fas fa-file-invoice"></i> Customer</a>
	<br>
	<br>
	<a href="#account" id="acc" class="sidebar-a" onclick="AccountVisible()"><i class="fas fa-user"></i> Account</a>
	<br>
	<br>
	<a href="#report" id="rep" class="sidebar-a" onclick="ReportVisible()"><i class="fas fa-warehouse"></i> Transaction</a>
	<div class="cr">
		<p id="cr" >This web application is developed Kushankur Das as major project of BCA in the year of 2019 at Techno India Salt Lake.</p>
	</div>
</div>
<!--sidebar-end-->
<!--main-->
<div id="main" class="main">
```

```
<!--snapshot-->
<div id="snapshot" class="container-temp">
        <div class="nofcWarper">
        <div class="card nofc">
                <div class="card-header">
                        Customer
                </div>
                <div class="card-body nofb-body">
                        <table class="striped-table">
                                <tr>
                                        <th>Total Number</th>
                                        <td><%=cdao.countData() %></td>
                                </tr>
                        </table>
                </div>
        </div>
        <div class="card nofc">
                <div class="card-header">
                        Accounts
                </div>
                <div class="card-body nofc-body">
                        <table class="striped-table">
                                <tr>
                                        <th>Total Number</th>
                                        <td><%=adao.countTotal() %></td>
                                </tr>
                                <tr>
                                        <th>Savings</th>
                                        <td><%=adao.countTotal("sb") %></td>
                                </tr>
                                <tr>
                                        <th>Fixed deposit</th>
                                        <td><%=adao.countTotal("fd") %></td>
                                </tr>
                                <tr>
                                        <th>Recurring Deposit</th>
```

```
                                        <td><%=adao.countTotal("rd") %></td>
                                    </tr>
                                </table>
                            </div>
                        </div>
                        <div class="card nofc">
                            <div class="card-header">
                                Quick Action
                            </div>
                            <div class="card-body nofa-body">
                                <table class="striped-table">
                                    <tr><th class="click"
onclick="openUpdateProfile()">Update Profile</th></tr>
                                    <tr><th class="click"
onclick="openChangePassword()">Change Password</th></tr>
                                </table>
                            </div>
                        </div>
                        </div>
                        <!--update-profile-end-->
                        <table class="displaynone" id="updateProfileData">
                            <tr>
                        <%
                            ResultSet updateprofrs = amdao.retrieveData();
                            while(updateprofrs.next()){
                        %>
                                <td><%=updateprofrs.getString("name") %></td>
                                <td><%=updateprofrs.getString("phno") %></td>
                                <td><%=updateprofrs.getString("email") %></td>
                                <td><%=updateprofrs.getString("dob") %></td>
                                <td><%=updateprofrs.getString("address") %></td>
                        <%
                                break;
                            }
                            updateprofrs.close();
                        %>
                            </tr>
```

```
                </table>

                <div id="updateprof" class="pop-overlay">
                <div class="pop">
                        <h2>Update Profile</h2>
                        <form action="UpdateAdminServ" method="post">
                                <div class="inputbox">
        <input type="text" name="uprrrUname" readonly value=<%=uname %>>
                                        <label>User Name</label>
                                </div>
                                <div class="inputbox">
        <input type="text" name="uprrrName" id="uprrrName" required>
                                        <label>Name</label>
                                </div>
                                <div class="inputbox">
        <input type="text" name="uprrrPhno" id="uprrrPhno" required>
                                        <label>Phone Number</label>
                                </div>
                                <div class="inputbox">
        <input type="text" name="uprrrEmail" id="uprrrEmail" required>
                                        <label>Email Address</label>
                                </div>
                                <div class="inputbox">
        <input type="text" name="uprrrDob" id="uprrrDob" required>
                                        <label>Birth Date</label>
                                </div>
                                <div class="inputbox">
        <input type="text" name="uprrrAddress" id="uprrrAddress" required>
                                        <label>Address</label>
                                </div>
        <input class="pop-btn-left" type="Submit" value="Update">
      <button type="button" class="pop-btn-right"onclick="closeUpdateProfile()">Cancel</button>
                        </form>
                </div>
        </div>
        <!--update-profile-end-->
```

```
<!--change-password-popup-->
<div id="chnagepass" class="pop-overlay">
    <div class="pop">
        <h2>Change Password</h2>
        <form action="UpdateAdminPassServ" method="post">
            <div class="inputbox">
                <input type="password" name="oldPass" required>
                <label>Current Password</label>
            </div>
            <div class="inputbox">
                <input type="password" name="newPass" required>
                <label>New Password</label>
            </div>
            <input type="hidden" name="cphjuname" value=<%=uname %>>
            <input class="pop-btn-left" type="Submit" value="Change">
<button type="button" class="pop-btn-right" onclick="closeChangePassword()">Cancel</button>
        </form>
    </div>
</div>
<!--change-password-end-->
<!--search-box-->
<div class="search">
    <div class="search-container">
<input type="text" placeholder="type user name here to search" id="totalDetailSearchKey"
onkeyup="totalDetailSearch()">
    </div>
</div>
<!--search-box-end-->
<!--search-result-->
    <div class="card accounts-lists displaynone" id="searchResult">
        <div class="card-header">
            Search Result
        </div>
        <div class="card-body accounts-lists-body">
            <p><strong>---------- Customer ----------</strong></p>
            <table class="striped-table" id="searchResultCustomer">
```

```
                <tr>
                    <th>Status</th>
                    <th>User Name</th>
                    <th>Name</th>
                    <th>Phone Number</th>
                    <th>Email Address</th>
                    <th>Birth Date</th>
                    <th>Address</th>
                    <th>Action</th>
                </tr>
            </table>
            <hr>
            <hr>
            <p><strong>---------- Savings account ----------</strong></p>
            <table class="striped-table" id="searchResultSb">
                <tr>
                    <th>Status</th>
                    <th>Account no.</th>
                    <th>Holder Name</th>
                    <th>Balance</th>
                    <th>Interest rate</th>
                    <th>Opening</th>
                    <th>Setting</th>
                </tr>
            </table>
            <hr>
            <hr>
            <p><strong>---------- Fixed deposit account ----------</strong></p>
            <table class="striped-table" id="searchResultFd">
                <tr>
                    <th>Status</th>
                    <th>Account no.</th>
                    <th>Holder Name</th>
                    <th>Balance</th>
                    <th>Interest Rate</th>
                    <th>No of month</th>
```

```
                                        <th>Amount</th>
                                        <th>Expiry</th>
                                        <th>Setting</th>
                                    </tr>
                                </table>
                                <hr>
                                <hr>
                                <p><strong>---------- Recurring Deposit account ----------
</strong></p>
                                <table class="striped-table" id="searchResultRd">
                                    <tr>
                                        <th>Status</th>
                                        <th>Account no.</th>
                                        <th>Holder Name</th>
                                        <th>Balance</th>
                                        <th>Interest Rate</th>
                                        <th>No of month</th>
                                        <th>Amount</th>
                                        <th>Expiry</th>
                                        <th>Setting</th>
                                    </tr>
                                </table>
                            </div>
                        </div>
                        <!--search-result-end-->
                    </div>
                    <!--snapshot-end-->
                    <!--customer-->
                    <div id="customer" class="container">
                        <!--search-box-->
                        <div class="search">
                            <div class="search-container">
        <input type="text" placeholder="type user name or name here to search" id="customerdetailsearchkey"
onkeyup="customerDetailsSearch()">
                            </div>
                        </div>
                        <!--search-box-end-->
```

```
<div class="card accounts-lists">
    <div class="card-header">
        List of Customer
    </div>
    <div class="card-body accounts-lists-body">
        <div>
            <table class="striped-table" id="customerList">
                <tr>
                    <th>Status</th>
                    <th>User Name</th>
                    <th>Name</th>
                    <th>Phone Number</th>
                    <th>Email Address</th>
                    <th>Birth Date</th>
                    <th>Address</th>
                    <th>Action</th>
                </tr>
                <%
                    ResultSet csrs = cdao.retrieveData();
                    while(csrs.next()){
                %>
                    <tr>

<td><%=csrs.getString("status") %></td>
<td><%=csrs.getString("uname") %></td>
<td><%=csrs.getString("holdername") %></td>
<td><%=csrs.getString("phno") %></td>
<td><%=csrs.getString("email") %></td>
<td><%=csrs.getString("dob") %></td>
<td><%=csrs.getString("address") %></td>
                                <td class="tooltip">
                    <i class="fas fa-cog click"></i>
                    <span class="tooltiptext">
        <span class="click" onclick="openCsPopUp(this)">Update</span>
                                <%
```

```
        if(csrs.getString("status").equals("active")){
                                                                                %>

        <br>

        <hr>

        <span class="click" onclick="openDeleteUser(this)">Delete</span>
                                                                                <%      }
%>
                                                                                </span>
                                                                        </td>
                                                                </tr>
                                                        <%
                                                                }
                                                        %>
                                                </table>
                                        </div>
                                </div>
                        </div>
        <button type="button" class="btn-ana" onclick="openNewCusPopup()">New Customer</button>
                </div>
         <!--action-update-popup-->
                <div class="pop-overlay" id="cspopup">
                        <div class="pop">
                                <h2>Update Customer</h2>
                                <form action="UpdateCusServ" method="post">
                                        <div class="inputbox">
                                <input id="csuname" type="text" name="upUname" readonly>
                                <label>User name</label>
                                        </div>
                                        <div class="inputbox">
                                <input id="csname" type="text" name="upName" required>
                                <label>Name</label>
                                        </div>
                                        <div class="inputbox">
 <input id="csphno" type="number" name="upPhno" min="6000000000" max="9999999999" required>
```

```
                                        <label>Phone number</label>
                                    </div>
                                    <div class="inputbox">
                                <input id="csemail" type="email" name="upEmail" required>
                                        <label>Email address</label>
                                    </div>
                                    <div class="inputbox">
                                        <input id="csdob" type="date" name="upDob" required>
                                        <label>Birth date</label>
                                    </div>
                                    <div class="inputbox">
                                <input id="csaddress" type="text" name="upAddress" required>
                                        <label>Address</label>
                                    </div>
                                <input class="pop-btn-left" type="Submit" name="" value="Update">
<button type="button" class="pop-btn-right" onclick="closeCsPopUp()">Cancel</button>
                                </form>
                        </div>
                </div>
                <!--action-update-popup-end-->
                <!--action-delete-popup-->
                <div class="pop-overlay" id="csdelpo">
                        <div class="pop">
<p>Do you really want to delete user: <span id="closeuname"></span>?</p>
                                <form action="DelCusServ" method="post">
                                        <input id="csdeluname" type="hidden" name="csdeluname">
<input class="pop-btn-left" type="Submit" name="" value="Delete">
<button type="button" class="pop-btn-right" onclick="closeDeleteUser()">Cancel</button>
                                </form>
                        </div>
                </div>
                <!--action-delete-popup-end-->
                <!--new-cus-->
                <div id="newcus" class="pop-overlay">
                        <div class="pop">
                                <h2>New Customer</h2>
```

```
                    <form action="NewCusServ" method="post">
                        <div class="inputbox">
                            <input type="text" name="regUname" required>
                            <label>User Name</label>
                        </div>
                        <div class="inputbox">
                            <input type="password" name="regPass" required>
                            <label>Password</label>
                        </div>
                        <div class="inputbox">
                            <input type="text" name="regName" required>
                            <label>Name</label>
                        </div>
                        <div class="inputbox">
    <input type="number" min="6000000000" max="9999999999" name="regPhno" required>
                            <label>Phone Number</label>
                        </div>
                        <div class="inputbox">
                            <input type="email" name="regEmail" required>
                            <label>Email Address</label>
                        </div>
                        <div class="inputbox">
                            <input type="date" name="regDob" required>
                            <label>Birth Date</label>
                        </div>
                        <div class="inputbox">
                            <input type="text" name="regAddress" required>
                        <label>Address</label>
                        </div>
    <input class="pop-btn-left" type="Submit" name="" value="Create">
  <button type="button" class="pop-btn-right" onclick="closeNewCusPopup()">Cancel</button>
                    </form>
                </div>
            </div>
            <!--new-cus-end-->
            <!--customer-end-->
```

```
<!--accounts-->
<div id="account" class="container">
    <!--sb-->
    <div class="card accounts-lists">
        <div class="card-header">
            Savings
        </div>
        <div class="card-body accounts-lists-body">
        <table class="striped-table" id="sbList">
                <tr>
                    <th>Status</th>
                    <th>Account no.</th>
                    <th>Holder Name</th>
                    <th>Balance</th>
                    <th>Interest rate</th>
                    <th>Opening</th>
                    <th>Setting</th>
                </tr>
                <%
        ResultSet sbrs = adao.retrieveAllAccount("sb");
                    while(sbrs.next()){
                %>
                    <tr>

<td><%=sbrs.getString("status") %></td>
<td><%=sbrs.getString("acno") %></td>
<td><%=sbrs.getString("uname") %></td>
<td><%=sbrs.getString("balance") %></td>
<td><%=sbrs.getString("rate") %></td>
 <td><%=sbrs.getString("opening") %></td>
                        <td class="tooltip">
                    <i class="fas fa-cog click"></i>
                    <span class="tooltiptext">
    <span class="click" onclick="openAsPopUp(this)">Update Account</span>
                        <%
```

```
        if(sbrs.getString("status").equals("active")){
                                                                                %>

        <br>

        <hr>

        <span class="click" onclick="openDeleteAc(this)">Close Account</span>
                                                                                <%      }
%>
                                                                                </span>
                                                                        </td>
                                                                </tr>
                                                        <%
                                                                }
                                                        %>
                                                </table>


                                        </div>
                                </div>
                                <!--sb-end-->
                                <!--fd-->
                                <div class="card accounts-lists">
                                        <div class="card-header">
                                                Fixed deposit
                                        </div>
                                        <div class="card-body accounts-lists-body">
                                        <table class="striped-table" id="fdList">
                                                                <tr>
                                                                        <th>Status</th>
                                                                        <th>Account no.</th>
                                                                        <th>Holder Name</th>
                                                                        <th>Balance</th>
                                                                        <th>Interest Rate</th>
                                                                        <th>No of month</th>
                                                                        <th>Amount</th>
                                                                        <th>Expiry</th>
```

```
                                                    <th>Setting</th>
                                                </tr>
                                                <%
                                            ResultSet fdrs = adao.retrieveAllAccount("fd");
                                                    while(fdrs.next()){
                                                %>
                                                    <tr>

        <td><%=fdrs.getString("status") %></td>
        <td><%=fdrs.getString("acno") %></td>
        <td><%=fdrs.getString("uname") %></td>
        <td><%=fdrs.getString("balance") %></td>
        <td><%=fdrs.getString("rate") %></td>
        <td><%=fdrs.getString("nomonth") %></td>
       <td><%=fdrs.getString("installment") %></td>
        <td><%=fdrs.getString("expiry") %></td>
                                                            <td class="tooltip">
                                                <i class="fas fa-cog click"></i>
                                                <span class="tooltiptext">
                <span class="click" onclick="openAsPopUp(this)">Update Account</span>
                                                                                <%

        if(fdrs.getString("status").equals("active")){
                                                                                %>

        <br>

        <hr>

        <span class="click" onclick="openDeleteAc(this)">Close Account</span>
                                                                                <%      }
%>
                                                                        </span>
                                                            </td>
                                                    </tr>
                                                <%
                                                    }
                                                    fdrs.close();
```

```
                                        %>
                                    </table>
                            </div>
                    </div>
                    <!--fd-end-->
                    <!--rd-->
                    <div class="card accounts-lists">
                            <div class="card-header">
                                    Recurring Deposit
                            </div>
                            <div class="card-body accounts-lists-body">
                                    <div>
                                            <table class="striped-table" id="rdList">
                                                    <tr>
                                                            <th>Status</th>
                                                            <th>Account no.</th>
                                                            <th>Holder Name</th>
                                                            <th>Balance</th>
                                                            <th>Interest Rate</th>
                                                            <th>No of month</th>
                                                            <th>Amount</th>
                                                            <th>Expiry</th>
                                                            <th>Setting</th>
                                                    </tr>
                                                    <%
                            ResultSet rdrs = adao.retrieveAllAccount("rd");
                                                            while(rdrs.next()){
                                                    %>
                                                            <tr>

<td><%=rdrs.getString("status") %></td>
<td><%=rdrs.getString("acno") %></td>
<td><%=rdrs.getString("uname") %></td>
<td><%=rdrs.getString("balance") %></td>
<td><%=rdrs.getString("rate") %></td>
<td><%=rdrs.getString("nomonth") %></td>
```

```
<td><%=rdrs.getString("installment") %></td>
<td><%=rdrs.getString("expiry") %></td>
                                                    <td class="tooltip">
                                        <i class="fas fa-cog click"></i>
                                        <span class="tooltiptext">
        <span class="click" onclick="openAsPopUp(this)">Update Account</span>
                                                                <%

if(rdrs.getString("status").equals("active")){
                                                                %>

<br>
<hr>
<span class="click" onclick="openDeleteAc(this)">Close Account</span>
                                                                <%      }
%>
                                                        </span>
                                                    </td>
                                            </tr>
                                        <%
                                            }
                                            rdrs.close();
                                        %>
                                    </table>
                                </div>
                            </div>
                        </div>
                        <!--rd-end-->
<button type="button" class="btn-ana" onclick="openNewAccountPopup()">Create new account</button>
                </div>
                        <!--account-update-popup-->
                        <div class="pop-overlay" id="aspopup">
                            <div class="pop">
                                <h2>Account Update</h2>
                                <form action="UpdateAccServ" method="post">
                                    <div class="inputbox">
                            <input id="asacno" type="text" name="asacno" readonly>
```

```
                                        <label>Account No.</label>
                                    </div>
                                    <div class="inputbox">
                                <input id="asstat" type="text" name="asstat" required>
                                        <label>Status</label>
                                    </div>
                                    <div class="inputbox">
                                <input id="ashname" type="text" name="asuname" readonly>
                                        <label>User Name</label>
                                    </div>
                                    <div class="inputbox">
                                <input id="asbalance" type="text" name="asbalance" required>
                                        <label>Balance</label>
                                    </div>
                                    <div class="inputbox">
                                <input id="asrate" type="text" name="asrate" required>
                                        <label>Rate</label>
                                    </div>
                                <input class="pop-btn-left" type="Submit" value="Update">
<button type="button" class="pop-btn-right" onclick="closeAsPopUp()">Cancel</button>
                                    </form>
                                </div>
                            </div>
                            <!--account-update-popup-end-->
                            <!--action-delete-popup-->
                            <div class="pop-overlay" id="asdelpo">
                                <div class="pop">
                    <p>Do you really want to delete account: <span id="asacnoo"></span>?</p>
                                    <form action="DelAccServ" method="post">
                                <input id="asdelacno" type="hidden" name="asdelacno">
                            <input class="pop-btn-left" type="Submit" value="Close">
<button type="button" class="pop-btn-right" onclick="closeDeleteAc()">Cancel</button>
                                    </form>
                                </div>
                            </div>
                            <!--action-delete-popup-end-->
```

```
<!--new-account-popup-->
<div id="newaccountpopup" class="pop-overlay">
<div class="pop">
<h2>New Account</h2>
<form action="NewAccAdminServ" method="post">
<div class="inputbox">
<label>Account type</label><br><br>
<select id="newactype" name="newactype" onchange="newAcCreation()">
<option value="sb">Savings</option>
<option value="fd">Fixed deposit</option>
<option value="rd">Recurring Deposit</option>
</select>
</div>
<div class="inputbox">
<label>User Name</label><br><br>
<select name="newacuname">
<%
ResultSet activeCusUnamers = ldao.activeCusUname();
while(activeCusUnamers.next()){
%>
<option value=<%=activeCusUnamers.getString(1) %>><%=activeCusUnamers.getString(1) %></option>
<%
}
activeCusUnamers.close();
%>
</select>
</div>
<div class="inputbox" id="newacbalancediv">
<input id="newacbalance" type="number" name="newacbalance" value="2000" required>
<label>Balance</label>
</div>
<div class="inputbox">
<input id="newacrate" type="number" name="newacrate" required>
<label>Interest rate</label>
</div>
```

```
                        <div class="displaynone inputbox" id="newacinstallmentdiv">
                        <input type="number" name="newacamt" value="0" required>
                                                <label>Amount</label>
                                        </div>
                                <div class="displaynone inputbox" id="newacnomdiv">
                        <input type="number" name="newacnomonth" value="0" required>
                                                <label>No of Month</label>
                                        </div>
                        <input class="pop-btn-left" type="Submit" value="Create">
    <button type="button" class="pop-btn-right" onclick="closeNewAccountPopup()">Cancel</button>
                                        </form>
                                </div>
                        </div>
                        <!--new-account-popup-end-->
                <!--accounts-end-->
                <!--report-->
                <div id="report" class="container">
                        <!--search-box-->
                        <div class="search">
                                <div class="search-container">
<input type="text" placeholder="type account number here to search" id="transdetailsearchkey"
onkeyup="transDetailsSearch()">
                                </div>
                        </div>
                        <!--search-box-end-->
                        <div class="card accounts-lists">
                                <div class="card-header">
                                        Transaction Details
                                </div>
                                <div class="card-body accounts-lists-body">
                                        <table class="striped-table" id="transdetailtable">
                                                <tr>
                                                        <th>Date - Time</th>
                                                        <th>From (Account No.)</th>
                                                        <th>To (Account No.)</th>
                                                        <th>Amount</th>
```

```
                                        <th>Remarks</th>
                                    </tr>
                                    <%
                                        ResultSet alltranrs = tdao.retriveAllData();
                                        while(alltranrs.next()){
                                    %>
                                        <tr>
        <td><%=alltranrs.getString("datetime") %></td>
        <td><%=alltranrs.getString("from") %></td>
        <td><%=alltranrs.getString("to") %></td>
        <td><%=alltranrs.getString("amount") %></td>
        <td><%=alltranrs.getString("remarks") %></td>
                                        </tr>
                                    <%
                                        }
                                        alltranrs.close();
                                    %>

                                </table>
                            </div>
                        </div>
                    </div>
                    <!--report-end-->
        </div>
        <!--main-end-->
        <script type="text/javascript">CusReload();</script>
        <script type="text/javascript">cr();</script>
</body>
</html>
<%}%>
```