

1. PROJECT DESCRIPTION

1.1 INTRODUCTION

EASY ONLINE BANKING project describes the nature of transactions done through INTERNET easily. Transactions related to ADMIN and CLIENT can be easily identified and projected through EASY ONLINE BANKING. This system makes all our work easier.

It is an online web application where the client can transact money from one account to another account to different banks. There are two different phases are used in this application. One is for the admin login and operations and other sides are for the client login and usages. The client can register their account with the use of his account number and pin.

With this registration, the user can login to a single account and can maintain or transfer money to many accounts and different banks. the client can also view his transaction details and deposit details. The admin can modify the client data and bank balance whereas the client can only view his account details. With one single login, the client can maintain all his bank details. There is no need of separate login for a different bank or account.

Importance of EOB

Easy Online Banking has become important nowadays in our day to day life. Also, it enables the customers who are unable to transact with BANK in direct. It plays a vital role in all banks with all customers. The project “EASY ONLINE BANKING” is an online web application where the client can transact money from one account to another account in spite of account maintained in different banks.

Easy Online Banking ensures the customer – banker relationship in a smooth way. It projects the nature of account maintained by the customer and highlights the amount transacted within few minutes. It states the Goodwill of both the customer and the banker as and when transacted. On the whole, EOB is an important tool for a customer to transact when he is far away from his native place.

Even fresher's who are about to open an account in any bank at any time can widespread his knowledge about Easy Online Banking and make use of it in his near future.

1.2 EXISTING SYSTEM

This system is purely computerized. As the name, itself specifies as **online** banking the transactions can be routed through internet only. Here, the customer who has an account in any bank only can make use of it. Otherwise it is of no use. The customers can sign in and start the transaction if he remembers his account number and its PIN number. It is mandatory for all the customers to keep in touch with the bankers randomly to know the actual status of the bank, updated net banking procedure etc.

Disadvantages of EOB

- User Interface is not wider to full page
- Fully Localized
- Does not send any notification to mail
- Data Analytics are not used
- Search process is not sufficient

1.2.1 HARDWARE SPECIFICATIONS

System : Intel Pentium

Hard Disk : 320GB

RAM : 2GB

OS : Windows 8.1

1.2.2 SOFTWARE SPECIFICATIONS

Front End

Language : JSP,

IDE : Net beans 8.0, Dreamweaver CC 2014 and
Edge Animate CC 2014.

Server : Apache Tomcat 8.0.3.

Back End

Database : Mongo Db. 3.2.3,

IDE : 3T Mongo Chef 3.5.1.

1.3 PROPOSED SYSTEM

The proposed system is highly modified via the user interface and also by the user interaction. Here the Java Mail API is used to connect to our Gmail port and send a registration conformation message. This helps to authenticate the right user and block the unauthorized user to use your login.

This helps to transact the amount from different database to various accounts from the savings account. The user can also view their transaction and deposit details with the help of the Data visualization.

The Dashboard helps to interact with the user and can analyst their account balance, account transaction and their account deposit amount details. The dashboard is very useful to understand the flow of amount and to track the status of their account details

Advantages of EOB

- API's like Java Mail API, Visulaize.js, Google Map API are used
- Partially Localized
- Send registration notification to mail
- Data Analytics are used
- Connects to Google Map
- Search process is more sufficient
- More comfortable for all persons
- User Interface is wider to full page
- Timely reachable and
- Most innovative process for the people to easily transact with banks at any time.
- All time availability
- Safely transferable.

1.2.1 HARDWARE SPECIFICATIONS

System : Intel Pentium

Hard Disk : 320GB

RAM : 2GB

OS : Windows 8.1

1.2.2 SOFTWARE SPECIFICATIONS

Front End

Language : JSP,

IDE : Net beans 8.0, Dreamweaver CC 2014 and
Edge Animate CC 2014.

Server : Apache Tomcat 8.0.3.

Back End

Database : Mongo Db. 3.2.3,

IDE : 3T Mongo Chef 3.5.1.

Bigdata Visualization

Server : JasperReports Server 6.3,

IDE : Jaspersoft Studio 6.3.

Supporting Files

Mongo dB Connector	: mongo-java-driver-3.2.2.jar,
Java Mail API	: mail.jar, smtp.jar, mailapi.jar, pop3.jar
Bigdata Visualization	: Visualize.js is built in JasperReports

1.2.3 SOFTWARE DESCRIPTION

MongoDB: (from *humongous*) is a free and open-source cross-platform document-oriented database program. Classified as a NoSQL database program, MongoDB uses JSON-like documents with schemas. MongoDB is developed by MongoDB Inc. and is free and open-source, published under a combination of the GNU Affero General Public License and the Apache License.

Like other NoSQL databases, MongoDB supports dynamic schema design, allowing the documents in a collection to have different fields and structures. The database uses a document storage and data interchange format called BSON, which provides a binary representation of JSON-like documents. Automatic sharding enables data in a collection to be distributed across multiple systems for horizontal scalability as data volumes increase.

MongoDB was created by Dwight Merriman and Eliot Horowitz, who had encountered development and scalability issues with traditional relational database approaches while building Web applications at DoubleClick, an Internet advertising company that is now owned by Google Inc. According to Merriman, the name of the database was derived from the word *humongous* to represent the idea of supporting large amounts of data. Merriman and Horowitz helped form 10Gen Inc. in 2007 to commercialize MongoDB and related software. The company was renamed MongoDB Inc. in 2013.

The database was released to open source in 2009 and is available under the terms of the Free Software Foundation's GNU AGPL Version 3.0 commercial license. At the time of this writing, among other users, the insurance company MetLife is using MongoDB for customer service applications, the website Craigslist is using it for archiving data, the CERN physics lab is using it for data aggregation and discovery and the The New York Times newspaper is using MongoDB to support a form-building application for photo submissions.

Java Mail API: The JavaMail API provides a platform-independent and protocol-independent framework to build mail and messaging applications. The JavaMail API provides a set of abstract classes defining objects that comprise a mail system. It is an optional package (standard extension) for reading, composing, and sending electronic messages.

JavaMail provides elements that are used to construct an interface to a messaging system, including system components and interfaces. While this specification does not define any specific implementation, JavaMail does include several classes that implement RFC822 and MIME Internet messaging standards. These classes are delivered as part of the JavaMail class package.

Following are some of the protocols supported in JavaMail API:

- **SMTP:** Acronym for **Simple Mail Transfer Protocol**. It provides a mechanism to deliver email.
- **POP:** Acronym for **Post Office Protocol**. POP is the mechanism most people on the Internet use to get their mail. It defines support for a single mailbox for each user. RFC 1939 defines this protocol.
- **IMAP:** Acronym for **Internet Message Access Protocol**. It is an advanced protocol for receiving messages. It provides support for multiple mailbox for each user, in addition to, mailbox can be shared by multiple users. It is defined in RFC 2060.
- **MIME:** Acronym for **Multipurpose Internet Mail Extensions**. . It is not a mail transfer protocol. Instead, it defines the content of what is transferred: the format of the messages, attachments, and so on. There are many different documents that take effect here: RFC 822, RFC 2045, RFC 2046, and RFC 2047. As a user of the JavaMail API, you usually don't need to worry about these formats. However, these formats do exist and are used by your programs.
- **NNTP and Others:** There are many protocols that are provided by third-party providers. Some of them are Network News Transfer Protocol (NNTP), Secure Multipurpose Internet Mail Extensions (S/MIME) etc.

Google API: It is a set of application programming interfaces (APIs) developed by Google which allow communication with Google_Services and their integration to other services. Examples of these include Search, Gmail, Translate or Google Maps. Third-party apps can use these APIs to take advantage of or extend the functionality of the existing services.

The APIs provide functionality like analytics, machine learning as a service (the Prediction API) or access to user data (when permission to read the data is given). Another important example is an embedded Google map on a website, which can be achieved using the Static maps API, Places API or Google Earth API

Visualize.js: It is a **JavaScript** API framework used to embed JasperReport Server reports & visualizations inside web applications. ... Developers can quickly and easily call backend JasperReports Server services using REST calls for common functions such as user authentication, data connectivity, repository services and more.

JasperReports Server: JasperReports Server is a stand-alone and embeddable reporting server. It provides reporting and analytics that can be embedded into a web or mobile application as well as operate as a central information hub for the enterprise by delivering mission critical information on a real-time or scheduled basis to the browser, mobile device, printer, or email inbox in a variety of file formats. JasperReports Server is optimized to share, secure, and centrally manage your Jaspersoft reports and analytic views.

Flexible Server Architecture

- Semantic layer for relational and non-relational data sources - BI Editions only
- On-premises, virtualized, or Cloud (SaaS & PaaS) deployment options
- Open standards, REST and SOAP-based web service architecture simplifies mobile and web application integration

Centralized Repository

- Connectivity to existing identity management systems to centralize and secure reports and analysis views
- Report access and usage auditing for compliance
- Granular security access down to the cell and column level

Ad hoc Reporting

- Web-based, drag-and-drop report designer creates interactive reports for dashboards, email distribution, or within a web application
- Metadata layer masks complex data descriptions with simplified business user-friendly names
- Web-based reports provide rich, interactive reports with drill down, filtering, animated charting, and more

Dashboards

- Web-based, drag-and-drop dashboard designer
- Single report and dashboard-level parameters drive user interaction
- Free-form layout designer for customized dashboard design

2. LOGICAL DEVELOPMENT

2.1 DFD

A two-dimensional diagram that explains how data is processed and transferred in a system. The graphical depiction identifies each source of data and how it interacts with other data sources to reach a common output. Individuals seeking to draft a data flow diagram must (1) identify external inputs and outputs, (2) determine how the inputs and outputs relate to each other, and (3) explain with graphics how these connections relate and what they result in. This type of diagram helps business development and design teams visualize how data is processed and identify or improve certain aspects.

It is also known as bubble charts. DFD is a designing tool used in the top-down approach to Systems Design. This context-level DFD is next "exploded", to produce a Level 1 DFD that shows some of the detail of the system being modeled. The Level 1 DFD shows how the system is divided into sub-systems (processes), each of which deals with one or more of the data flows to or from an external agent, and which together provide all of the functionality of the system as a whole. It also identifies internal data stores that must be present in order for the system to do its job, and shows the flow of data between the various parts of the system.

Data flow diagrams are one of the three essential perspectives of the structured-systems analysis and design method SSADM. The sponsor of a project and the end users will need to be briefed and consulted throughout all stages of a system's evolution. With a data flow diagram, users are able to visualize how the system will operate, what the system will accomplish, and how the system will be implemented. The old system's dataflow diagrams can be drawn up and compared with the new system's data flow diagrams to draw comparisons to implement a more efficient system.

2.1.1 Login Module

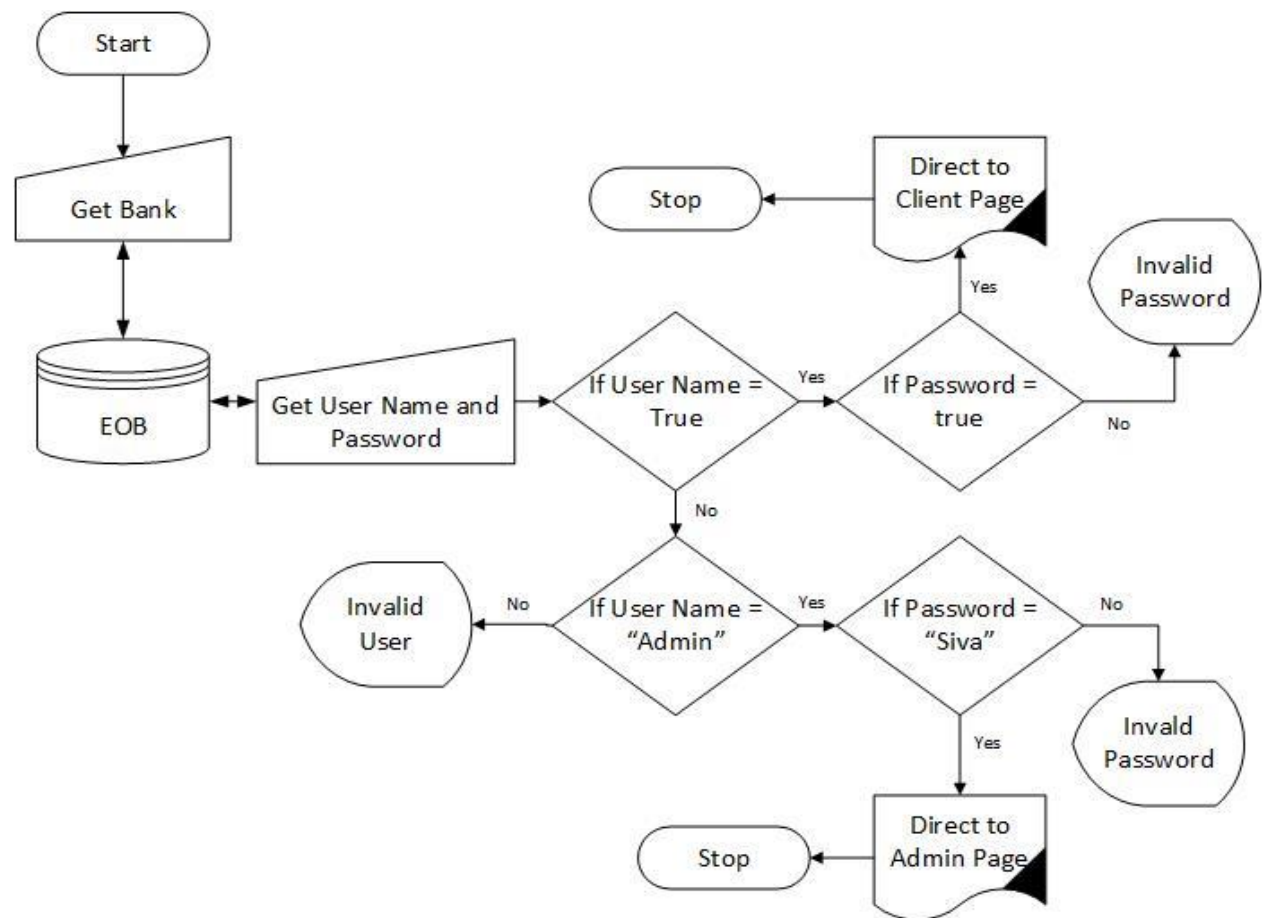


Fig: 2.1.1

2.1.2 Registration Module

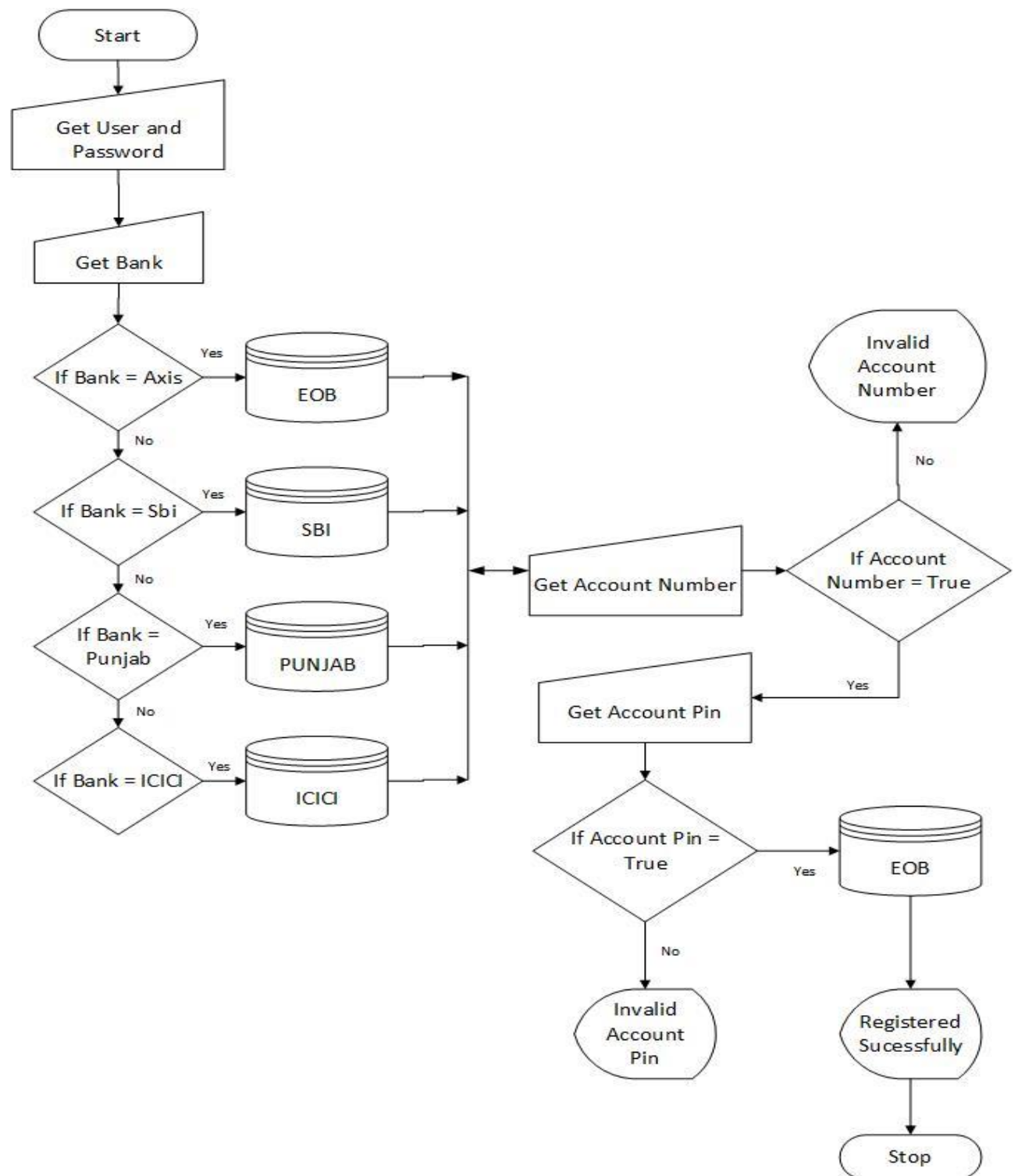


Fig: 2.1.2

2.1.3 Transaction Module

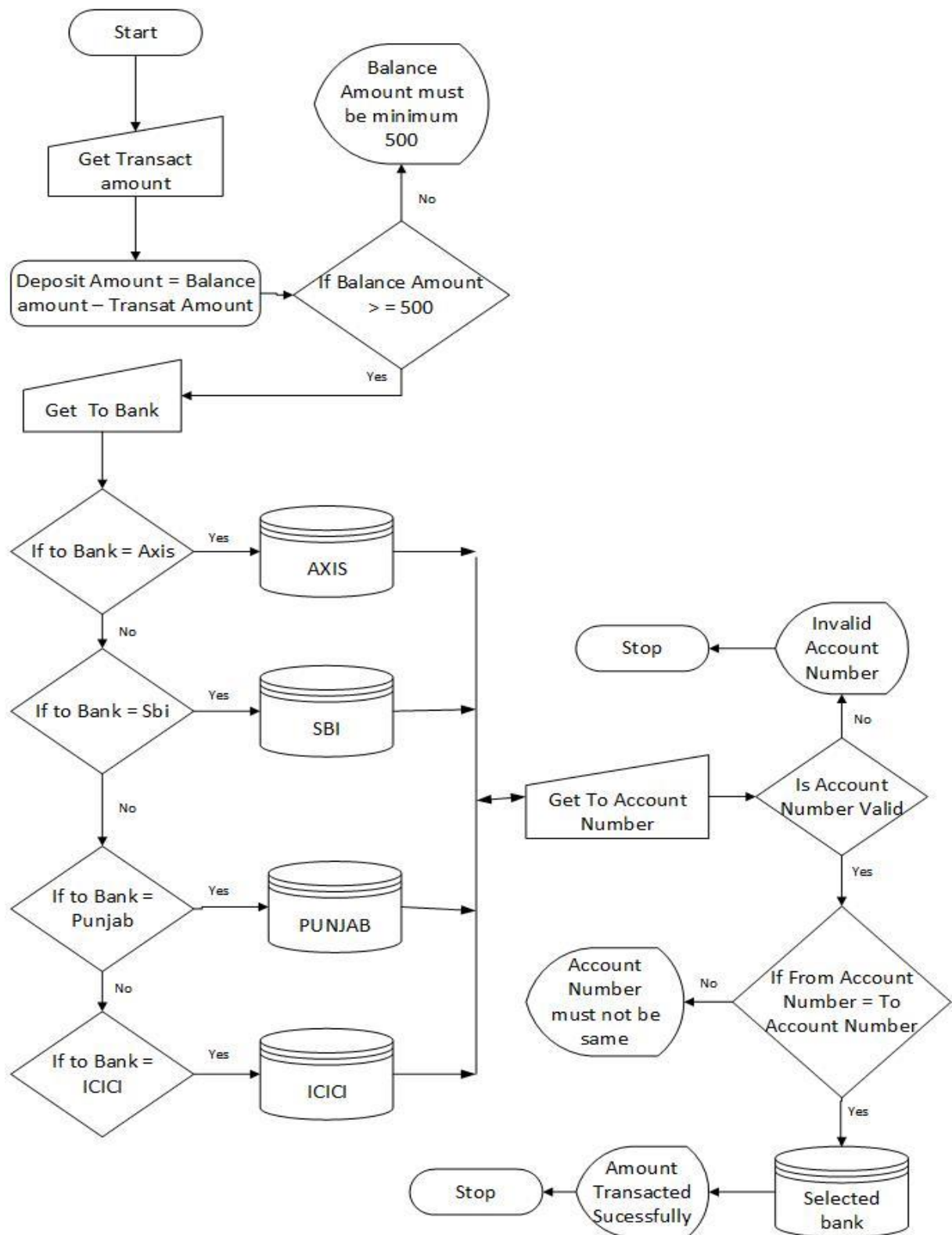


Fig: 2.1.3

2.2 ARCHITECTURAL DIAGRAM

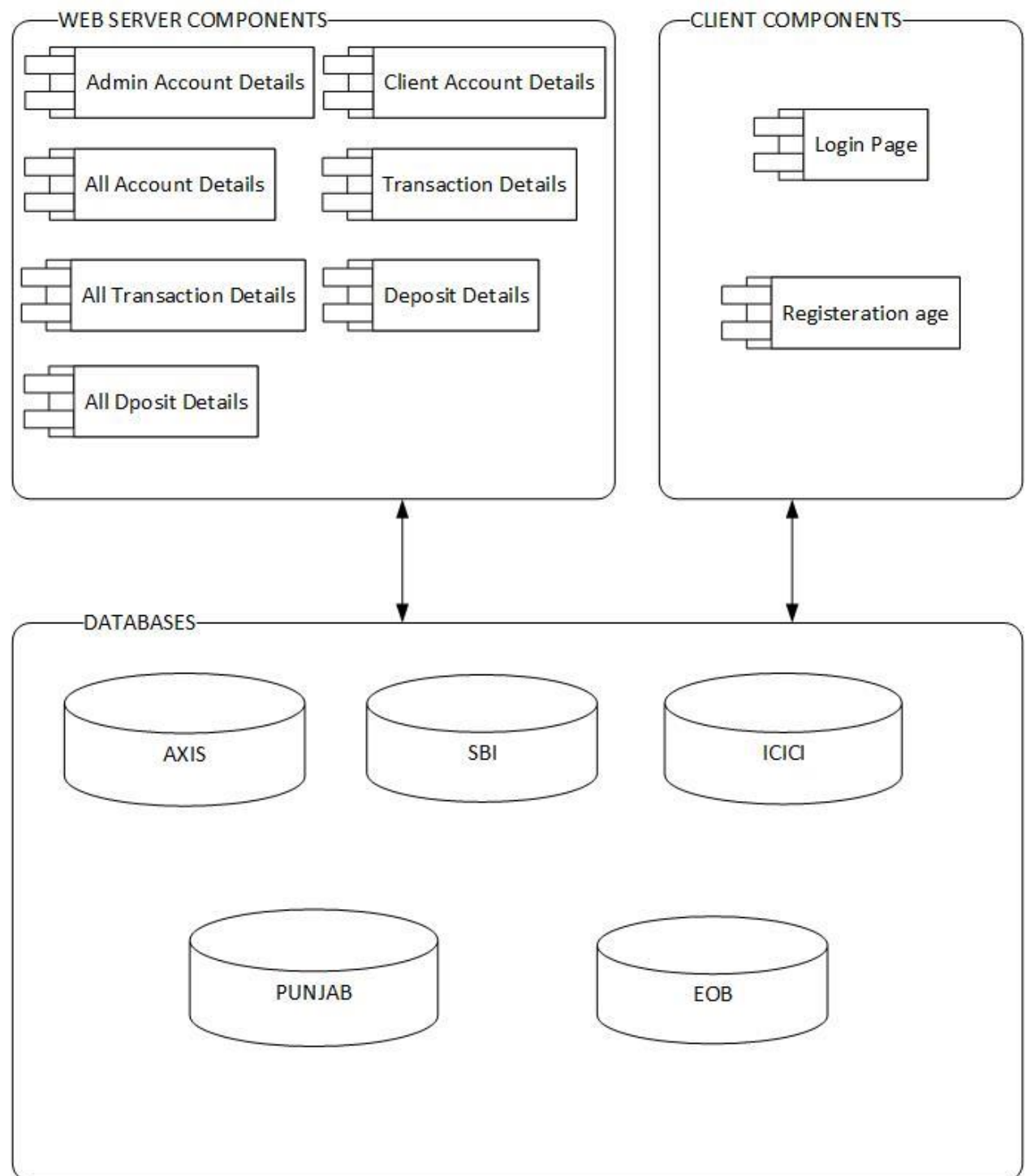


Fig: 2.2

3. DATABASE DESIGN

3.1 DOCUMENT DESIGN

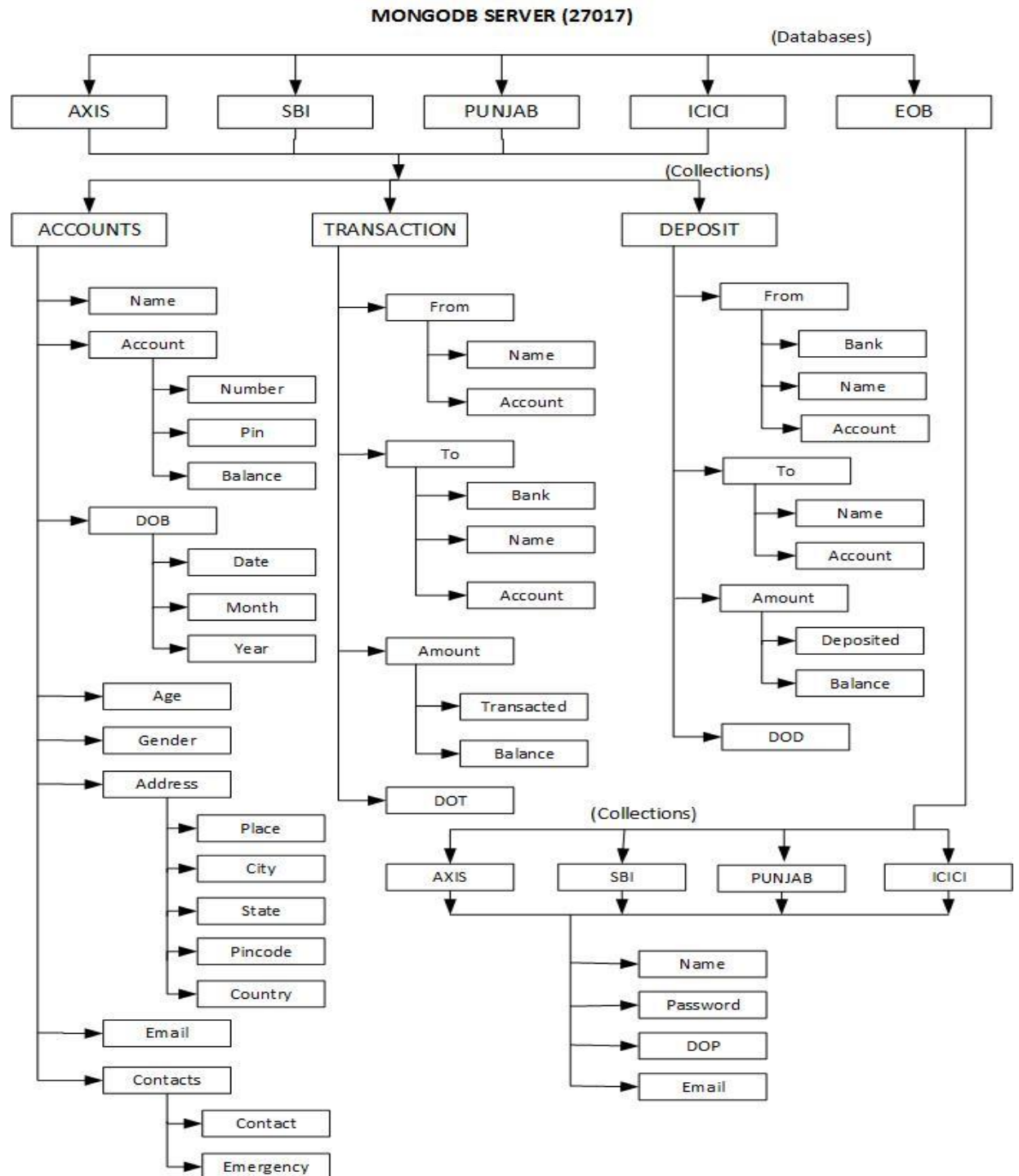


Fig: 3.1

4. PROGRAM DESIGN

4.1. MODULES DESCRIPTION

The online banking is an easy way to transact money from one account to other.

This web application is made up of two tier architecture. The one is the Design part which contains the html5 and css3 design core and other is the business part which connects to the database and process the information. This web application has two phases. One is for the Administrator and other is for the Client users. It web forms of this project are

- 4.1.1 Welcome Page
- 4.1.2 Login
- 4.1.3 Registration
- 4.1.4 Admin Page
- 4.1.5 Client Details
- 4.1.6 Transaction
- 4.1.7 Client Transaction Details
- 4.1.8 Client Deposit Details
- 4.1.9 All Transaction Details
- 4.1.10 on Details
- 4.1.11 All Deposit Details
- 4.1.12 All Account Details

4.1.1 Welcome Page

This page represents the details of how to sign in and sign up related to our transaction. It also states how to go about for a new person to open an account on his name and his PIN number will be registered immediately for his/her reference. Also, a person already registered can also sign in and continue his/her transaction without any delay.

4.1.2 Login Page

This page is used to login by both the Client user and the Administrator. With the help of the validation and a valid authentication the page is directed to Client page or admin page. If the user is not registered, then he/she cannot login for the online banking.

4.1.3 Registration Page

In this page the Client user can register for the online banking. To register for online banking the user must have an account on a bank. The person who has no bank account cannot register for the online banking. The user gets an Email notification for their successful registration. This helps to prevent the unauthorized user to use your email address. The user who have already registered using the mail id cannot register again with the same mail id.

4.1.4 Admin Page

The Admin page can view or edit the client details whereas the client detail page can only view their details and also can update their personal details. The admin can insert, search, update and delete any of the account details form any bank. This page uses the Google Map API which helps to fill the Address details of the user. The Google map API autocompletes the address field by accessing the google map databases. This helps to keep the track of the user location.

4.1.5 Client details Page

The client details page is for only the client users. The client detail page can only view their details and also can update their personal details. The client doesn't have a privilege to edit the account details. But they edit their personal details like name, age, date of birth, address etc... They also have the Google Map API services to fill their address field.

4.1.6 Transaction

This is where the transaction takes place. First the user gets authenticated and can transfer their amount via this application. The user cannot transfer them to their own account. Their account balance must be minimum 500 /-.

4.1.7 Client Transaction details

In the transaction page the client can transact the money from one account to another. In the Transaction details page the user can view their particular transaction details. Also, he can update the availability of amount in his account in few minutes

4.1.8 Client Deposit details

Here the client can deposit any amount of money as and when he thinks. He can be safe in all corners when he has the possibility of depositing the amount at all times. Deposit of money gives him more confidence that he has saved some amount for his future. Here the admin can view the account deposit information using the data visualization i.e. Admin can view the details in the form of different types of charts. This helps to analyze the report for the future predictions.

4.1.9 All transaction details

Transactions of all types through online banking makes the work of any person feel comfortable and acceptable because his time is consumed rather than standing in queue in Bank and make his transactions complete. Here also the admin can view the account deposit information using the data visualization i.e. Admin can view the details in the form of different types of charts. This helps to analyze the report for the future predictions.

4.1.10 All deposit details

All kinds of deposit, whether Time Deposit, Fixed Deposit or Recurring Deposit or Savings through salary can be transacted through NET and can assess the nature of deposit to be maintained simultaneously.

4.1.11 All Account Details Page

Transactions related to cash or kind can be processed through NET BANKING from the place where we stay. This becomes easier for a customer or client to transact confidently and cautiously with more belief and safety corners.

5. TESTING

Testing is a series of different tests that whose primary purpose is to fully exercise the computer based system. Although each test has a different purpose, all work should verify that all system element has been properly integrated and performed allocated function. Testing is the process of checking whether the developed system works according to the actual requirement and objectives of the system.

The philosophy behind testing is to find the errors. A good test is one that has a high probability of finding an undiscovered error. A successful test is one that uncovers the undiscovered error. Test cases are devised with this purpose in mind. A test case is a set of data that the system will process as an input.

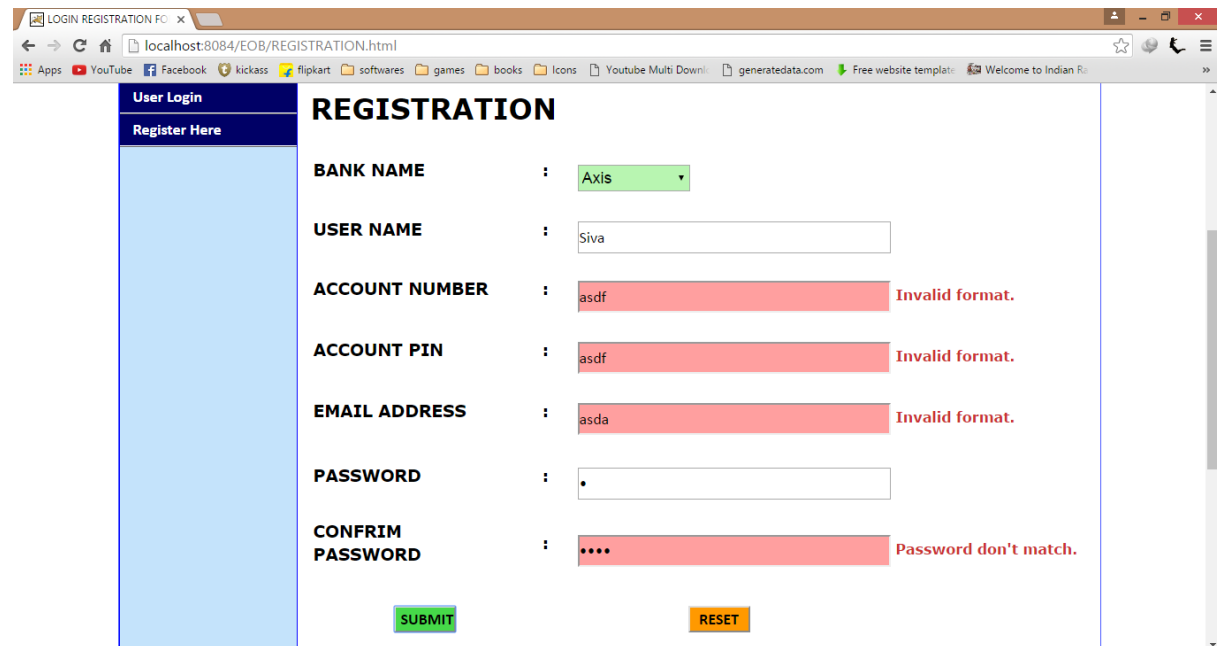
TYPES OF TESTING

- Unit testing
- Integration testing
- Validation testing
- Output testing
- User acceptance testing

UNIT TESTING

All modules were tested and individually as soon as they were completed and were checked for their correct functionality.

Test result



The screenshot shows a web browser window with the address bar displaying 'localhost:8084/EOB/REGISTRATION.html'. The page has a dark blue sidebar on the left with 'User Login' and 'Register Here' links. The main content area is titled 'REGISTRATION' and contains a form with the following fields and validation messages:

Field	Value	Validation Message
BANK NAME	Axis	
USER NAME	Siva	
ACCOUNT NUMBER	asdf	Invalid format.
ACCOUNT PIN	asdf	Invalid format.
EMAIL ADDRESS	asda	Invalid format.
PASSWORD	.	
CONFIRM PASSWORD	Password don't match.

At the bottom of the form, there are two buttons: 'SUBMIT' (green) and 'RESET' (orange).

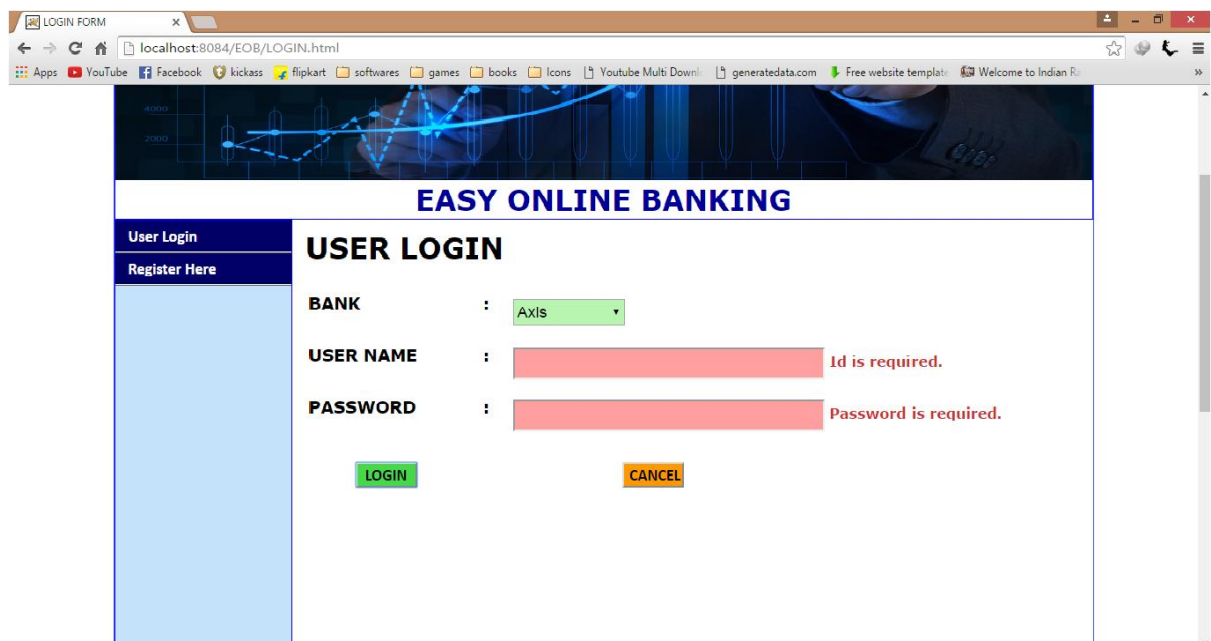
Fig: 5.1

INTEGRATION TESTING

The entire project was split into small program; each of these single programs gives a frame as an output. These programs were tested individually; at last all these programs were combined together by creating another program where all these constructors were used. It gives a lot of problem by not functioning in an integrated manner.

The user interface testing is important since the user has to declare that the arrangements made in frames are convenient and it is satisfied. When the frames were given for the test, the end user gave suggestion. Based on their suggestions the frames were modified and put into practice.

Test result



The screenshot displays a web browser window with the address bar showing 'localhost:8084/EOB/LOGIN.html'. The page features a header with a blue background and a line graph. Below the header, the title 'EASY ONLINE BANKING' is centered. On the left side, there is a vertical navigation menu with two links: 'User Login' and 'Register Here'. The main content area is titled 'USER LOGIN' and contains three input fields: 'BANK' (a dropdown menu with 'Axis' selected), 'USER NAME' (a text input field with a red border and the message 'Id is required.'), and 'PASSWORD' (a text input field with a red border and the message 'Password is required.'). At the bottom of the form, there are two buttons: 'LOGIN' (green) and 'CANCEL' (orange).

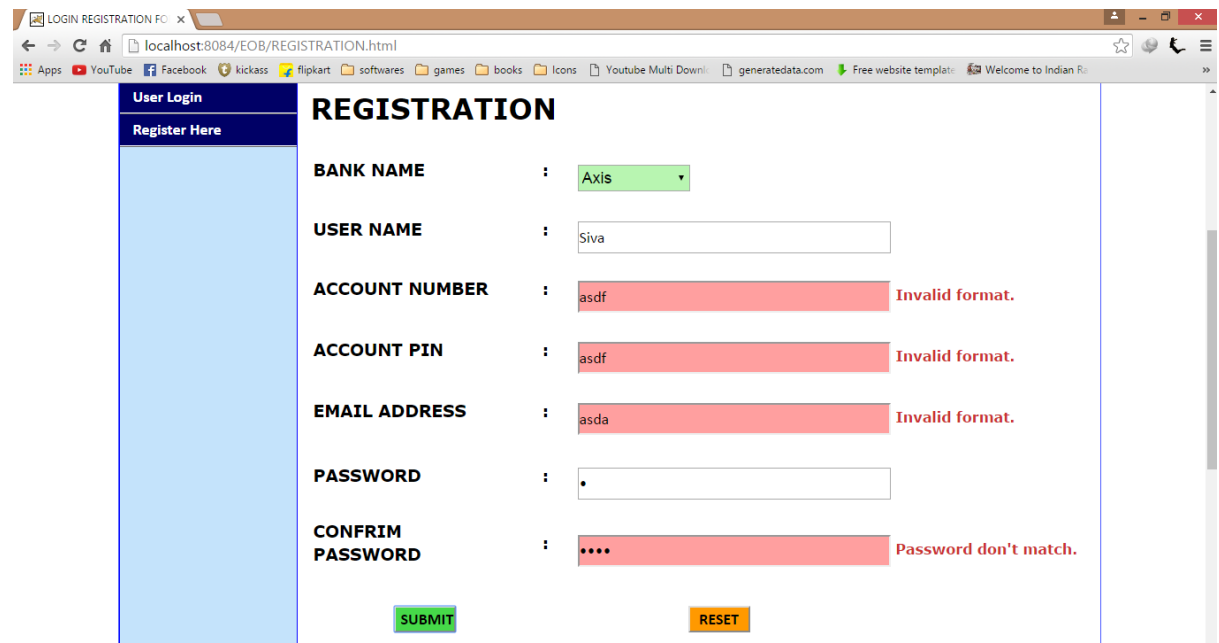
Fig: 5.2

VALIDATION TESTING

It is said that validation is successful when the software functions in a systematic manner that can be reasonably accepted by the customers. This type of testing is very important because it is the only way to check whether the requirements given by user have been completely fulfilled.

The input given to various forms are validated effectively. The validated input is given for all modules. Each module is tested independently.

Test result



The screenshot shows a web browser window with the address bar displaying 'localhost:8084/EOB/REGISTRATION.html'. The page has a dark blue sidebar on the left with 'User Login' and 'Register Here' links. The main content area is titled 'REGISTRATION' and contains a form with the following fields and validation messages:

Field	Value	Validation Message
BANK NAME	Axis	
USER NAME	Siva	
ACCOUNT NUMBER	asdf	Invalid format.
ACCOUNT PIN	asdf	Invalid format.
EMAIL ADDRESS	asda	Invalid format.
PASSWORD	.	
CONFRIM PASSWORD	Password don't match.

At the bottom of the form, there are two buttons: 'SUBMIT' (green) and 'RESET' (orange).

Fig: 5.3

OUTPUT TESTING

After performing the unit testing the next step is output testing of the proposed system. Since the system cannot be useful if it does not produce the required output. Asking the user about the format in which the system is required tests the output displayed or generated by the system under consideration. Here the output format is considered in two ways. One is on screen and another one is printed format. The output format on the screen is found to be corrected as the format was designed in the system phase according to the user needs. And for the hardcopy the output comes according to the specifications requested by the user.

USER ACCEPTANCE TESTING

In these testing procedures, the project is given to the customer to test whether all requirements have been fulfilled and after the user is fully satisfied. The project is perfectly ready. If the user makes request for any change and if they found any errors those all errors has to be taken into consideration and to be correct it to make a project a perfect project.

Test result

The screenshot displays a web browser window with the address bar showing 'localhost:8084/EOB/REGISTRATION.html'. The page features a navigation bar with 'User Login' and 'Register Here' links. The main content area is titled 'REGISTRATION' and contains a form with the following fields and values:

Field	Value
BANK NAME	Axis
USER NAME	Siva
ACCOUNT NUMBER	1408
ACCOUNT PIN	2222
EMAIL ADDRESS	sdfs@gmail.com
PASSWORD	[Masked]
CONFRIM PASSWORD	[Masked]

At the bottom of the form, there are two buttons: 'SUBMIT' (green) and 'RESET' (orange).

Fig: 5.4

6. CONCLUSION

This project titled “**EASY ONLINE BANKING**” is a website where all the job goers as well as business people can get benefitted. The main aim of this project is to make and feel easy transactions with the bankers as and when possible and necessary. Whenever a person likes to transact / deposit any amount, EOB assures 100% safety and comfortable. Money, Time, Security and Easiness plays a vital role in Net Banking Process.

FUTURE ENHANCEMENT

The user will be informed every day with new updates by message. Can select them by test. Then the whole working functionality can be developed in an efficient way.

7. REFERENCES

Following books and eBooks are used to complete this project reports.

- www.youtube.com
- www.mongodb.org
- www.google.com
- www.community.jaspersoft.com/project/jasperreports-server
-

8. APPENDIX

8.1 SOURCE CODE

Logincheck.jsp

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>
<% @page import="com.mongodb.BasicDBObject"%>
<% @page import="com.mongodb.DBObject"%>
<% @page import="com.mongodb.DB" %>
<% @page import="com.mongodb.DBCollection" %>
<% @page import="com.mongodb.DBCursor" %>
<% @page import="com.mongodb.MongoClient" %>
<% @page import="com.mongodb.MongoException" %>
<% @page import="java.util.Date"%>
<% @page import="java.net.UnknownHostException" %>
<% @page import="java.io.*" %>

<%! String bank;
    String accno;
    String pswd;
    String uname;

    String daccno;
    String dpswd;

    int i=0;
%>
<%! DB db; DBCollection coll; MongoClient mongo; BasicDBObject document;%>
<%
    session=request.getSession();
    try
    {
```

```

bank=request.getParameter("Bname");
accno=request.getParameter("ACCNONUMBER");
pswd=request.getParameter("PASSWORD");

mongo = new MongoClient("localhost", 27017);
db = mongo.getDB("EOB");
coll = db.getCollection(bank);

BasicDBObject search = new BasicDBObject();
search.put("Account Number",accno);

DBCursor cursor = coll.find(search);

if(cursor!=null)
{
while (cursor.hasNext())
{
DBObject cur = cursor.next();
uname=cur.get("Name").toString();
daccno=cur.get("Account Number").toString();
dpswd=cur.get("Password").toString();
mongo.close();
}
}
else
{
mongo.close();
session.setAttribute("msg", "INVALID ACCOUNT");
response.sendRedirect("AuthenticationMessage.jsp");
}
if(accno.equals(daccno))
{
if(pswd.equals(dpswd))
{

```

```

        mongo.close();
        session.setAttribute("accno", accno);
        session.setAttribute("user", uname);
        session.setAttribute("bank", bank);
        response.sendRedirect("DETAILS.jsp");
    }
    else
    {
        mongo.close();
        session.setAttribute("msg", "INVALID PASSWORD");
        response.sendRedirect("AuthenticationMessage.jsp");
    }
}
else if("Admin".equals(accno) && "Siva".equals(pswd))
{
    mongo.close();
    session.setAttribute("page", "Admin");
    response.sendRedirect("ACCOUNT DETAILS.jsp");
}
else
{
    mongo.close();
    session.setAttribute("msg", "INVALID USER");
    response.sendRedirect("AuthenticationMessage.jsp");
}
}
catch(Exception e)
{
    mongo.close();
    out.println(e);
}
%>

```

Register.jsp

```
<% @page contentType="text/html" pageEncoding="UTF-8"%>
```



```

<% @page import="com.mongodb.BasicDBObject"%>
<% @page import="com.mongodb.DBObject"%>
<% @page import="com.mongodb.DB" %>
<% @page import="com.mongodb.DBCollection" %>
<% @page import="com.mongodb.DBCursor" %>
<% @page import="com.mongodb.MongoClient" %>
<% @page import="com.mongodb.MongoException" %>
<% @page import="java.util.Date"%>
<% @page import="java.net.UnknownHostException" %>

<%! DB db; DBCollection coll; MongoClient mongo; BasicDBObject document;%>

<%! String bank;
    String name;
    String acno;
    String acpin;
    String eid;
    String pswd;

    String dname;
    String dacno;
    String dacpin;
    String deid;
    String dpswd;

%>
<%
    try
    {
        bank=request.getParameter("Bname");
        name=request.getParameter("NAME");
        acno=request.getParameter("ACNUMBER");
        acpin=request.getParameter("ACPIN");
        eid=request.getParameter("EID");
        pswd=request.getParameter("PASSWORD");

```

```

try
{
    mongo = new MongoClient("localhost", 27017);
    db = mongo.getDB(bank);
    coll = db.getCollection("ACCOUNTS");
}
catch(Exception e)
{
    out.print(e);
}

BasicDBObject search = new BasicDBObject();
search.put("Account.Number",acno);

DBCursor cursor = coll.find(search);

if(cursor!=null)
{
    while (cursor.hasNext())
    {
        DBObject cur = cursor.next();

        dname=cur.get("Name").toString();
        DBObject account = (BasicDBObject) cur.get("Account");
        dacno=account.get("Number").toString();
        dacpin=account.get("Pin").toString();
        mongo.close();

    }
}
else
{
    mongo.close();
    session.setAttribute("msg", "INVALID ACCOUNT");
}

```

```

        response.sendRedirect("AuthenticationMessage.jsp");
    }
}
catch(Exception e)
{
    mongo.close();
    out.println(e);
}
try
{

if(name.equals(dname))
{
    if(acno.equals(dacno))
    {
        if(acpin.equals(dacpin))
        {
            try
            {
                mongo = new MongoClient("localhost", 27017);
                db= mongo.getDB("EOB");
                coll= db.getCollection(bank);
            }
            catch(Exception e)
            {
                out.print(e);
            }

            document = new BasicDBObject();
            document.put("Name",name);
            document.put("Account Number",acno);
            document.put("Password",pswd);
            document.put("DOP",new Date());
            document.put("Email",eid);

```

```

try
{
coll.insert(document);
mongo.close();
//session.setAttribute("msg", "REGISTERED Sucessfully");
//session.setAttribute("tomail", eid);
//response.sendRedirect("RegisterAuthentication.jsp");
%>

<jsp:forward page="mail">
    <jsp:param name="to" value="<%= eid%>"></jsp:param>
</jsp:forward>

<%

}
catch(Exception e)
{
mongo.close();
session.setAttribute("msg", "USER ALREADY REGISTERED");
response.sendRedirect("AuthenticationMessage.jsp");
}
}
else
{
mongo.close();
session.setAttribute("msg", "INVALID ACCOUNT PIN");
response.sendRedirect("AuthenticationMessage.jsp");
}
}
else
{
mongo.close();
session.setAttribute("msg", "INVALID ACCOUNT NUMBER");

```

```

        response.sendRedirect("AuthenticationMessage.jsp");
    }

}

else
{
    mongo.close();
    session.setAttribute("msg", "INVALID USER");
    response.sendRedirect("AuthenticationMessage.jsp");
}

}

catch(Exception e)
{
    mongo.close();
    out.println(e);
}

%>

```

AccountDetails.jsp

```

<% @page import="java.io.*" %>
<% @page import="java.util.Calendar" %>
<% @page import="com.mongodb.BasicDBObject"%>
<% @page import="com.mongodb.DBObject"%>
<% @page import="com.mongodb.DB" %>
<% @page import="com.mongodb.DBCollection" %>
<% @page import="com.mongodb.DBCursor" %>
<% @page import="com.mongodb.MongoClient" %>
<% @page import="com.mongodb.MongoException" %>
<% @page import="java.util.Date"%>
<% @page import="java.net.UnknownHostException" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />

```

```

<title>ACCOUNT DETAILS</title>

<link href="CSS/STYLES3.css" rel="stylesheet" type="text/css" /><style
type="text/css">

</style>
<link href="jQueryAssets/SpryValidationSelect.css" rel="stylesheet" type="text/css"
/>
<script src="jQueryAssets/SpryValidationSelect.js" type="text/javascript"></script>
<script type="text/javascript">
function Age(){ //v2.0
    var a=document.getElementById('YEAR').value;
    var nowdate=new Date();
    var nowyr=nowdate.getFullYear();
    var i=nowyr-a;
    document.ACCOUNT_DETAILS.AGE.value = i;
}
</script>
</head>
<%! String getpage;%>
<%try
{
    getpage=session.getAttribute("page").toString();
}
catch(Exception e)
{
    response.sendRedirect("LOGIN.html");
}
%>
<%! DB db; DBCollection coll; MongoClient mongo; BasicDBObject document;
DBCursor cursor;%>
<%!
    String name;
    String btn=null;

```

```

String bank="";
String acno="";
String acpin="";
String blncamnt="";
String uname="";
String date;
String month;
String year;
String age;
String gender;
String addr="";
String city="";
String pncd="";
String cntry="";
String state="";
String cntcno="";
String emgcntcno="";
String eid="";
String dnum="";
String route="";
%>
<%
try
{
    btn=request.getParameter("SEARCH");
    if(btn!=null)
    {
        bank=request.getParameter("Bname");
        acno=request.getParameter("ACCNONUMBR");

        try
        {
            mongo = new MongoClient("localhost", 27017);
            db = mongo.getDB(bank);

```

```

        coll = db.getCollection("ACCOUNTS");
    }
    catch(Exception e)
    {
        mongo.close();
        session.setAttribute("msg", "PLEASE SELECT THE BANK");
        response.sendRedirect("AdminMessage.jsp");
    }

    BasicDBObject search = new BasicDBObject();
    search.put("Account.Number",acno);
    try
    {
        cursor = coll.find(search);
    }
    catch(Exception e)
    {
        mongo.close();
        session.setAttribute("msg", "INVALID USER NAME.TRY AGAIN");
        response.sendRedirect("AdminMessage.jsp");
    }
    if(cursor==null)
    {
        mongo.close();
        session.setAttribute("msg", "INVALID USER NAME.TRY AGAIN");
        response.sendRedirect("AdminMessage.jsp");
    }
    else if(cursor!=null)
    {
        while (cursor.hasNext())
        {
            DBObject cur = cursor.next();

            uname=cur.get("Name").toString();

```



```

age=cur.get("Age").toString();
eid=cur.get("Email").toString();

DBObject account = (BasicDBObject) cur.get("Account");
acno=account.get("Number").toString();
acpin=account.get("Pin").toString();
blncamnt=account.get("Balance").toString();

DBObject address = (BasicDBObject) cur.get("Address");
        addrs=address.get("Place").toString();
        dnum=address.get("Door").toString();
route=address.get("Route").toString();
city=address.get("City").toString();
state=address.get("State").toString();
pncd=address.get("Pincode").toString();
cntry=address.get("Country").toString();

DBObject contact = (BasicDBObject) cur.get("Contacts");
cntcno=contact.get("Contact").toString();
emgcntcno=contact.get("Emergency").toString();

        mongo.close();
    }
}
}
}
catch(Exception e)
{
    mongo.close();
    session.setAttribute("msg", "TTRY AGAIN");
    response.sendRedirect("AdminMessage.jsp");
}

%>

```



```

<form action="ACCOUNT DETAILS.jsp" method="post"
name="ACCOUNT_DETAILS" target="_self" id="ACCOUNT_DETAILS">
  <table width="1320" height="1375" border="0">
    <tr>
      <td height="61"><h3>BANK NAME</h3></td>
      <td><h3>:</h3></td>
      <td><span id="spryselect1">
        <select name="Bname" required="required" id="Bname">
          <option value="" selected="selected">Select Bank</option>
          <option value="AXIS">Axis</option>
          <option value="SBI">Sbi</option>
          <option value="PUNJAB">Punjab</option>
          <option value="ICICI">ICICI</option>
        </select>
        <span class="selectInvalidMsg">Please select a valid item.</span><span
class="selectRequiredMsg">Please select an item.</span></span></td>
      </tr>
      <tr>
        <td height="61" colspan="3"><div align="center" class="HEAD">ACCOUNT
INFORMATION</div></td>
      </tr>
      <tr>
        <td height="61"><h3>ACCOUNT NUMBER</h3></td>
        <td><h3>:</h3></td>
        <td>
          <input name="ACCNONUMBR" type="number" required="required"
class="CONTACT" value="<%=acno%>" id="ACCNONUMBR" />
        </td>
      </tr>
      <tr>
        <td width="287" height="61"><h3>ACCOUNT PIN</h3></td>
        <td width="40"><h3>:</h3></td>
        <td width="921">

```

```

        <input name="ACCNOPIN" type="number" required="required"
value="<%=acpin%>" class="CONTACT" id="ACCNOPIN" min="1000" /></td>
    </tr>
    <tr>
        <td height="61"><h3>BALANCE AMOUNT</h3></td>
        <td><h3>:</h3></td>
        <td><input name="BLNCAMNT" type="number" required="required"
value="<%=blncamnt%>" class="CONTACT" id="BLNCAMNT"/></td>
    </tr>
    <tr>
        <td height="65" colspan="3"><div align="center"
class="HEAD">PERSONAL INFORMATION</div></td>
    </tr>
    <tr>
        <td height="65"><h3>USER NAME</h3></td>
        <td><h3>:</h3></td>
        <td><input name="UNAME" type="text" required="required" class="FIELD"
id="UNAME" autocomplete="off" value="<%= unname %>" /></td>
    </tr>
    <tr>
        <td height="53"><h3>DATE OF BIRTH</h3></td>
        <td><h3>:</h3></td>
        <td><table width="559" height="52" border="0">
            <tr>
                <td width="153"><select name="DATE" required="required"
class="D_O_B" id="DATE">
                    <option selected="selected">DATE</option>
                    <%
                                for(int i=1;i<=31;i++)
                                {
                                    %>
                                <option value="<%= i %>"><%= i %></option>
                                <% } %>
                                </select></td>

```

```

        <td width="182"><select name="MONTH" required="required"
class="D_O_B" id="MONTH">
    <option selected="selected">MONTH</option>
    <option value="JANUARY">JANUARY</option>
    <option value="FEBRUARY">FEBRUARY</option>
    <option value="MARCH">MARCH</option>
    <option value="APRIL">APRIL</option>
    <option value="MAY">MAY</option>
    <option value="JUNE">JUNE</option>
    <option value="JULY">JULY</option>
    <option value="AUGUST">AUGUST</option>
    <option value="SEPTEMBER">SEPTEMBER</option>
    <option value="OCTOBER">OCTOBER</option>
    <option value="NOVEMBER">NOVEMBER</option>
    <option value="DECEMBER">DECEMBER</option>
</select></td>

    <td width="210"><select name="YEAR" required="required"
class="D_O_B" id="YEAR" onchange="Age()" >
    <option selected="selected">YEAR</option>

    <%
        Calendar cal=Calendar.getInstance();
        int y=cal.get(Calendar.YEAR);
        for (int j=1990;j<=y;j++)
        {
            %>

            <option value="<%= j %>"><%= j %></option>
        } %>
    </select></td>

</tr>
</table></td>

</tr>
<tr>
    <td height="59"><h3>AGE</h3></td>
    <td><h3>:</h3></td>

```



```

        </td>
    </tr>
    <tr>
        <td height="59"><h3>STREET ROUTE </h3></td>
        <td><h3>:</h3></td>
        <td><textarea name="ROUTE" cols="45" rows="5" required="required"
class="TEXTAREA" id="route"><%=route%></textarea></td>
    </tr>
    <tr>
        <td height="59"><h3>CITY</h3></td>
        <td><h3>:</h3></td>
        <td><input name="CITY" type="text" required="required" class="CONTACT"
id="locality" value="<%=city%>" readonly="readonly" />
    </td>
    </tr>
    <tr>
        <td height="59"><h3>STATE</h3></td>
        <td><h3>:</h3></td>
        <td>
            <input name="STATE" type="text" required="required" class="CONTACT"
id="administrative_area_level_1" value="<%=state%>" readonly="readonly" />
        </td>
    </tr>
    <tr>
        <td height="59"><h3>ZIP CODE</h3></td>
        <td><h3>:</h3></td>
        <td><input name="PCODE" type="number" required="required"
value="<%=pncd%>" class="CONTACT" id="postal_code" /></td>
    </tr>
    <tr>
        <td height="59"><h3>COUNTRY</h3></td>
        <td><h3>:</h3></td>
        <td>

```

```

        <input name="COUNTRY" type="text" required="required"
class="CONTACT" id="country" value="<%=cntry%>" readonly="readonly" />
    </td>
</tr>
<tr>
    <td height="59"><h3>CONTACT NUMBER</h3></td>
    <td><h3>:</h3></td>
    <td><input name="CNUMBER" type="number" required="required"
class="CONTACT" id="CNUMBER" max="9999999999" min="1000000"
value="<%=cntcno%>" /></td>
</tr>
<tr>
    <td height="59"><h3>EMERGENCY CONTACT</h3></td>
    <td><h3>:</h3></td>
    <td><input name="ENUMBER" type="number" class="CONTACT"
value="<%=emgcntcno%>" id="ENUMBER" max="9999999999"
min="1000000"/></td>
</tr>
<tr>
    <td height="59"><h3>EMAIL ADDRESS</h3></td>
    <td><h3>:</h3></td>
    <td><input name="EADDRESS" type="email" required="required"
value="<%= eid %>" class="EMAIL" id="EADDRESS" autocomplete="off" /></td>
</tr>
<script>
var autocomplete;
var componentForm = {
    street_number: 'short_name',
    route: 'long_name',
    locality: 'long_name',
    administrative_area_level_1: 'long_name',
    country: 'long_name',
    postal_code: 'short_name'
};

```



```

function initAutocomplete() {
    autocomplete = new google.maps.places.Autocomplete(
(document.getElementById('autocomplete')),
    {types: ['geocode']}));
    autocomplete.addListener('place_changed', fillInAddress);
}

function fillInAddress() {
    var place = autocomplete.getPlace();

    for (var component in componentForm) {
        document.getElementById(component).value = "";
        document.getElementById(component).disabled = false;
    }

    for (var i = 0; i < place.address_components.length; i++) {
        var addressType = place.address_components[i].types[0];
        if (componentForm[addressType]) {
            var val = place.address_components[i][componentForm[addressType]];
            document.getElementById(addressType).value = val;
        }
    }
}

function geolocate() {
    if (navigator.geolocation) {
        navigator.geolocation.getCurrentPosition(function(position) {
            var geolocation = {
                lat: position.coords.latitude,
                lng: position.coords.longitude
            };
            var circle = new google.maps.Circle({
                center: geolocation,
                radius: position.coords.accuracy
            });

```

```

        autocomplete.setBounds(circle.getBounds());
    });
}
}
</script>
<script
src="https://maps.googleapis.com/maps/api/js?key=AIzaSyDuQMJQ_SkJv0Z-
jQFRuqnvplx9UYzExbw&libraries=places&callback=initAutocomplete"
async defer></script>
<tr>
<td height="74" colspan="3"><table width="1272" height="88" border="0">
<tr>
<td width="249"><div align="center">
<input name="Button" type="submit" class="SUBMIT" id="Button"
form="ACCOUNT_DETAILS" formaction="dbwrk.jsp" formmethod="post"
value="INSERT" />
</div></td>
<td width="249"><div align="center">
<input name="SEARCH" type="submit" formnovalidate="formnovalidate"
class="VIEW" id="SEARCH" form="ACCOUNT_DETAILS"
formaction="ACCOUNT_DETAILS.jsp" formmethod="post" value="SEARCH" />
</div></td>
<td width="249"><div align="center">
<input name="Button" type="submit" class="UPDATE" id="Button"
form="ACCOUNT_DETAILS" formaction="dbwrk.jsp" formmethod="post"
value="UPDATE" />
</div></td>
<td width="249"><div align="center">
<input name="CALCEL" type="reset" class="CANCEL" id="CALCEL"
form="ACCOUNT_DETAILS" value="CANCLE" />
</div></td>
<td width="254"><div align="center" id="END">

```

```

        <input name="Button" type="submit" formnovalidate="formnovalidate"
class="DELETE" id="Button" form="ACCOUNT_DETAILS"
formaction="dbwrk.jsp" formmethod="post" value="DELETE" />
    </div></td>
</tr>
</table></td>
</tr>
</table>
</form>
<p>&nbsp;      </p>
<!-- end .content --></div>
<div class="footer">
    <p>&nbsp;</p>
    <!-- end .container --></div>
</div>
<script type="text/javascript">
var spryselect1 = new Spry.Widget.ValidationSelect("spryselect1", {invalidValue:"-
1"});
</script>
</body>
</html>

```

AllAccountDetail.jsp

```

<% @page import="java.io.*" %>
<% @page import="java.util.Calendar" %>
<% @page import="com.mongodb.BasicDBObject"%>
<% @page import="com.mongodb.DBObject"%>
<% @page import="com.mongodb.DB" %>
<% @page import="com.mongodb.DBCollection" %>
<% @page import="com.mongodb.DBCursor" %>
<% @page import="com.mongodb.MongoClient" %>
<% @page import="com.mongodb.MongoException" %>
<% @page import="java.util.Date"%>
<% @page import="java.net.UnknownHostException" %>

```

```

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>ACCOUNT DETAILS</title>

<link href="CSS/STYLES4.css" rel="stylesheet" type="text/css" /><style
type="text/css">

</style>
<script type="text/javascript">

</script>
</head>
<%! String getpage;%>
<%try
{
    getpage=session.getAttribute("page").toString();
}
catch(Exception e)
{
    response.sendRedirect("LOGIN.html");
}
%>
<%! DB db; DBCollection coll; MongoClient mongo; BasicDBObject document;%>
<%!
    String bank;
    String name;
    String acno;
    String acpin;
    String blncamnt;
    String date;
    String month;

```



```

<!-- end .content --></div>
<table width="100%" height="167" border="1" id="DBACCOUNTS">
  <tbody>
    <tr class="HEAD">
      <td width="7%" align="center" valign="middle"><h6>NAME</h6></td>
      <td width="8%" align="center" valign="middle"><h6>ACCOUNT
NUMBER</h6></td>
      <td width="8%" align="center" valign="middle"><h6>ACCOUNT
PIN</h6></td>
      <td width="8%" align="center" valign="middle"><h6>BALANCE
AMOUNT</h6></td>
      <td width="7%" align="center" valign="middle"><h6>DOB</h6></td>
      <td width="5%" align="center" valign="middle"><h6>AGE</h6></td>
      <td width="7%" align="center" valign="middle"><h6>GENDER</h6></td>
      <td width="22%" align="center" valign="middle"><h6>ADDRESS</h6>
      <td width="9%" align="center" valign="middle"><h6>EMAIL</h6>
      <td width="10%" align="center" valign="middle"><h6>CONTACT</h6>
      <td width="9%" align="center" valign="middle"><h6>EMERGENCY
CONTACT</h6>
    </td>
  </tr>

  <% try
    {
      if(request.getParameter("Button")!=null)
      {
        bank=request.getParameter("Bname");

        try
        {
          mongo = new MongoClient("localhost", 27017);
          db = mongo.getDB(bank);
          coll = db.getCollection("ACCOUNTS");
        }
        catch(Exception e)

```

```

{
mongo.close();
session.setAttribute("msg", "PLEASE SELECT THE BANK");
response.sendRedirect("AdminMessage.jsp");
}

DBCursor cursor = coll.find();
if(cursor!=null)
{
while (cursor.hasNext())
{
DBObject cur = cursor.next();

name=cur.get("Name").toString();
age=cur.get("Age").toString();
eid=cur.get("Email").toString();
gender=cur.get("Gender").toString();

DBObject account = (BasicDBObject) cur.get("Account");
acno=account.get("Number").toString();
acpin=account.get("Pin").toString();
blncamnt=account.get("Balance").toString();

DBObject dob= (BasicDBObject) cur.get("DOB");
date=dob.get("Date").toString();
month=dob.get("Month").toString();
year=dob.get("Year").toString();

Dateofbirth=date+"-"+month+"-"+year;

DBObject address = (BasicDBObject) cur.get("Address");
addrs=address.get("Place").toString();
dnum=address.get("Door").toString();

```



```

        route=address.get("Route").toString();
        city=address.get("City").toString();
        state=address.get("State").toString();
        pncd=address.get("Pincode").toString();
        cntry=address.get("Country").toString();

        addr=addr+"-"+dnum+"-"+route+"-"+city+"-
"+state+"-"+pncd+"-"+cntry;

```

```

        DBObject contact = (BasicDBObject) cur.get("Contacts");
        cntcno=contact.get("Contact").toString();
        emgcntcno=contact.get("Emergency").toString();

        %>

<tr>
<td style="font-size: small"><%= name %></td>
<td style="font-size: small"><%= acno %></td>
<td style="font-size: small"><%= acpin %></td>
<td style="font-size: small"><%= blncamnt %></td>
<td style="font-size: small"><%= Dateofbirth %></td>
<td style="font-size: small"><%= age %></td>
<td style="font-size: small"><%= gender %></td>
<td style="font-size: small"><%= addr %></td>
<td style="font-size: small"><%= eid %></td>
<td style="font-size: small"><%= cntcno %></td>
<td style="font-size: small"><%= emgcntcno %></td>
</tr>

        <% }

    }
else
{
    mongo.close();
    session.setAttribute("msg", "INVALID USER NAME.TRY AGAIN");
    response.sendRedirect("ClientMessage.jsp");
}

```

```

        }
    }
    catch(Exception e)
    {
        mongo.close();
        session.setAttribute("msg", "THE USER NAME OR ACCOUNT NUMBER
ALREADY EXIST.TRY AGAIN");
        response.sendRedirect("ClientMessage.jsp");
    }

%>
</tbody>
</table>
</body>
</html>

```

Clntwrk.jsp

```

<% @page import="com.mongodbDBObject"%>
<% @page contentType="text/html" pageEncoding="UTF-8"%>
<% @page import="com.mongodb.BasicDBObject"%>
<% @page import="com.mongodb.DB" %>
<% @page import="com.mongodb.DBCollection" %>
<% @page import="com.mongodb.DBCursor" %>
<% @page import="com.mongodb.MongoClient" %>
<% @page import="com.mongodb.MongoException" %>
<% @page import="java.util.Date"%>
<% @page import="java.net.UnknownHostException" %>

<%! DB db; DBCollection coll; MongoClient mongo; BasicDBObject document;%>

<%! String btn;
    String bank;
    String acno;
    String acpin;
    int blncamnt;

```

String name;
int date;
String month;
int year;
int age;
String gender;
String addr;
String city;
int pncd;
String cntry;
String state;
String cntcno;
String emgcntcno;
String eid;
String dnum;
String route;

String fbank;
String fsacno;
int facno;
int samnt;
int tamnt;
int bamnt;

String dname;
String dacpin;
String dacno;
String sacpin;
String sacno;
String dblnc;
String tuser;
int iblnc;
int addamnt;
int iacno;

Date dtdate;

```
%>
<%
    try
    {
        btn=request.getParameter("Button");
        bank=request.getParameter("Bname");
        name=request.getParameter("UNAME");

        if("UPDATE".equals(btn))
        {
            acno=request.getParameter("ACCNONUMBR");
            acpin=request.getParameter("ACCNOPIN");
            blncamnt=Integer.parseInt(request.getParameter("BLNCAMNT"));
            date=Integer.parseInt(request.getParameter("DATE"));
            month=request.getParameter("MONTH");
            year=Integer.parseInt(request.getParameter("YEAR"));
            age=Integer.parseInt(request.getParameter("AGE"));
            gender=request.getParameter("GENDER");
            addrs=request.getParameter("HADDRESS");
            dnum=request.getParameter("DNUM");
            route=request.getParameter("ROUTE");
            city=request.getParameter("CITY");
            state=request.getParameter("STATE");
            pncd=Integer.parseInt(request.getParameter("PCODE"));
            cntry=request.getParameter("COUNTRY");
            cntcno=request.getParameter("CNUMBER");
            emgcntcno=request.getParameter("ENUMBER");
            eid=request.getParameter("EADDRESS");
        }
        else if("TRANSACT".equals(btn))
        {
```

```

fbank=request.getParameter("FBname");

    fsacno=request.getParameter("FACNUMBER");
    facno=Integer.parseInt(fsacno);

    sacpin=request.getParameter("ACPIN");
    samnt=Integer.parseInt(request.getParameter("SAMOUNT"));
    tamnt=Integer.parseInt(request.getParameter("TAMOUNT"));
    bamnt=Integer.parseInt(request.getParameter("BAMOUNT"));
    sacno=request.getParameter("TACNUMBER");

}

}
catch(Exception e)
{
    out.println("test");
    out.print(e);
}

if("UPDATE".equals(btn))
{

    try
    {
        mongo = new MongoClient("localhost", 27017);
        db = mongo.getDB(bank);
        coll = db.getCollection("ACCOUNTS");
    }
    catch(Exception e)
    {
        mongo.close();
        out.print("DB Error");
    }
}

```

```

        out.print(e);
    }

    BasicDBObject account = new BasicDBObject();
        account.put("Number",acno);
        account.put("Pin", acpin);
        account.put("Balance",blncamnt);

    BasicDBObject dob = new BasicDBObject();
        dob.put("Date",date);
        dob.put("Month", month);
        dob.put("Year",year);

    BasicDBObject contact = new BasicDBObject();
        contact.put("Contact",cntcno);
        contact.put("Emergency", emgcntcno);

    BasicDBObject address = new BasicDBObject();
        address.put("Place",addrs);
        address.put("City", city);
        address.put("State", state);
        address.put("Pincode",pncd);
        address.put("Country",cntry);

    document = new BasicDBObject();
        document.put("Name", name);
        document.put("Account", account);
        document.put("DOB", dob);
        document.put("Age", age);
        document.put("Gender", gender);
        document.put("Address", address);
        document.put("Email", eid);
        document.put("Contacts", contact);

```

```

        try
        {
            BasicDBObject search = new BasicDBObject();
            search.put("Name",name);
            coll.update(search,document);
            mongo.close();
            session.setAttribute("msg", "SUCESSFULLY UPDATED");
            response.sendRedirect("ClientMessage.jsp");
        }
        catch(Exception e)
        {
            mongo.close();
            session.setAttribute("msg", "THE USER NAME OR ACCOUNT NUMBER
ALREADY EXIST.TRY AGAIN");
            response.sendRedirect("ClientMessage.jsp");
        }

    }
    else if("TRANSACTION".equals(btn))
    {

        try
        {
            mongo = new MongoClient("localhost", 27017);
            db = mongo.getDB(fbank);
            coll = db.getCollection("ACCOUNTS");
        }
        catch(Exception e)
        {
            mongo.close();
            out.print(e);
        }
    }

```

```

BasicDBObject search = new BasicDBObject();
search.put("Account.Number",fsacno);

DBCursor cursor = coll.find(search);
if(cursor!=null)
{
    while (cursor.hasNext())
    {
        DBObject cur = cursor.next();

        dname=cur.get("Name").toString();

        DBObject account = (BasicDBObject) cur.get("Account");
        dacno=account.get("Number").toString();
        dacpin=account.get("Pin").toString();
        mongo.close();
    }
}
else
{
    mongo.close();
    session.setAttribute("msg", "INVALID USER NAME.TRY AGAIN");
    response.sendRedirect("ClientMessage.jsp");
}

if(fsacno.equals(dacno))
{
    if(sacpin.equals(dacpin))
    {
        try
        {
            mongo = new MongoClient("localhost", 27017);
            db = mongo.getDB(bank);
            coll = db.getCollection("ACCOUNTS");

```



```

    }
    catch(Exception e)
    {
        mongo.close();
        out.print("Db error inside transact");
        out.print(e);
    }
    try
    {
        iacno=Integer.parseInt(sacno);
        BasicDBObject tosearch = new BasicDBObject();
        tosearch.put("Account.Number",sacno);

        DBCursor cursor2 = coll.find(tosearch);
        if(cursor2!=null)
        {
            while (cursor2.hasNext())
            {

                DBObject cur2 = cursor2.next();
                tuser=cur2.get("Name").toString();
                DBObject account2 = (BasicDBObject) cur2.get("Account");
                dacno=account2.get("Number").toString();
                dblnc=account2.get("Balance").toString();
                mongo.close();

            }
        }
        else
        {
            mongo.close();
            session.setAttribute("msg", "INVALID ACCOUNT NUMBER.TRY
AGAIN");

```

```

        response.sendRedirect("ClientMessage.jsp");
    }

    iblnc=Integer.parseInt(dblnc);
    addamnt=iblnc+tamnt;

    if(sacno.equals(dacno))
    {
        if(facno!=iacno)
        {
            if(bamnt>=500)
            {
                try
                {
                    mongo = new MongoClient("localhost", 27017);
                    db = mongo.getDB(fbank);
                    coll=db.getCollection("ACCOUNTS");
                }
                catch(Exception e)
                {
                    mongo.close();
                    out.print("Db error inside blnc check");
                    out.print(e);
                }
            }
            try
            {
                BasicDBObject balance = new BasicDBObject();
                balance.put("Account.Balance",bamnt);

                BasicDBObject setbalance = new BasicDBObject();
                setbalance.put("$set",balance);

                BasicDBObject frmacnt = new BasicDBObject();
                frmacnt.put("Account.Number",fsacno);
            }
            catch(Exception e)
            {
                out.print("Db error inside frmacnt check");
                out.print(e);
            }
        }
    }
}

```

```

coll.update(frmacnt,setbalance);
mongo.close();
}
catch(Exception e)
{
    mongo.close();
    out.print("update error 1");
    out.println(e);
}

try
{
    mongo = new MongoClient("localhost", 27017);
    db = mongo.getDB(bank);
    coll=db.getCollection("ACCOUNTS");
}
catch(Exception e)
{
    mongo.close();
    out.print("db error aftr update 1");
    out.print(e);
}

try
{
    BasicDBObject balance = new BasicDBObject();
    balance.put("Account.Balance",addamnt);

    BasicDBObject setbalance = new BasicDBObject();
    setbalance.put("$set",balance);

    BasicDBObject frmacnt = new BasicDBObject();
    frmacnt.put("Account.Number",sacno);

```

```

coll.update(frmacnt,setbalance);
mongo.close();
}
catch(Exception e)
{
    mongo.close();
    out.print("update error 2");
    out.println(e);
}

dtdate=new Date();
try
{
    mongo = new MongoClient("localhost", 27017);
    db = mongo.getDB(fbank);
    coll=db.getCollection("TRANSACTION");
}
catch(Exception e)
{
    mongo.close();
    out.print("db error aftr update error 2");
    out.print(e);
}

try
{
    BasicDBObject from = new BasicDBObject();
    from.put("Name",name);
    from.put("Account",fsacno);

    BasicDBObject to = new BasicDBObject();
    to.put("Bank",bank);
    to.put("Name",tuser);
    to.put("Account",sacno);

```

```

BasicDBObject amount = new BasicDBObject();
amount.put("Transacted",tamnt);
amount.put("Balance",bamnt);

document = new BasicDBObject();
document.put("From",from);
document.put("To",to);
document.put("Amount",amount);
document.put("DOT",dtdate);

coll.insert(document);

mongo.close();
}
catch(Exception e)
{
    mongo.close();
    out.print("transaction insert error");
    out.println(e);
}

try
{
    mongo = new MongoClient("localhost", 27017);
    db = mongo.getDB(bank);
    coll=db.getCollection("DEPOSIT");
}
catch(Exception e)
{
    mongo.close();
    out.print(e);
}
try

```

```

{
    BasicDBObject from = new BasicDBObject();
    from.put("Bank",fbank);
    from.put("Name",name);
    from.put("Account",fsacno);

    BasicDBObject to = new BasicDBObject();
    to.put("Name",tuser);
    to.put("Account",sacno);

    BasicDBObject amount = new BasicDBObject();
    amount.put("Deposited",tamnt);
    amount.put("Balance",addamnt);

    document = new BasicDBObject();
    document.put("From",from);
    document.put("To",to);
    document.put("Amount",amount);
    document.put("DOD",dtdate);

    coll.insert(document);

    mongo.close();
}
catch(Exception e)
{
    mongo.close();
    out.print("deposit error");
    out.println(e);
}

mongo.close();
session.setAttribute("msg", "AMOUNT SUCESSFULLY
TRANSACTIONED");

```

```

        response.sendRedirect("ClientMessage.jsp");
    }
    else
    {
        mongo.close();
        session.setAttribute("msg", "YOUR BALANCE AMOUNT MUST
BE ATLEAST 500/-");
        response.sendRedirect("ClientMessage.jsp");
    }
}
else
{
    mongo.close();
    session.setAttribute("msg", "TO ACCOUNT NUMBER CAN NOT BE
SAME AS FROM ACCOUNT NUMBER. TRY AGAIN");
    response.sendRedirect("ClientMessage.jsp");
}

}
else
{
    mongo.close();
    session.setAttribute("msg", "INVALID TO ACCOUNT NUMBER");
    response.sendRedirect("ClientMessage.jsp");
}
}

catch(Exception e)
{
    out.print("overall error");
    out.println(e);
}

else
{

```

```

        mongo.close();
        session.setAttribute("msg", "INVALID ACCOUNT PIN");
        response.sendRedirect("ClientMessage.jsp");
    }

}

else
{
    mongo.close();
    session.setAttribute("msg", "INVALID USER");
    response.sendRedirect("ClientMessage.jsp");
}

}

%>

```

Dbwrk.jsp

```

<% @page contentType="text/html" pageEncoding="UTF-8"%>
<% @page import="com.mongodb.BasicDBObject"%>
<% @page import="com.mongodb.DB" %>
<% @page import="com.mongodb.DBCollection" %>
<% @page import="com.mongodb.DBCursor" %>
<% @page import="com.mongodb.MongoClient" %>
<% @page import="com.mongodb.MongoException" %>
<% @page import="java.util.Date"%>
<% @page import="java.net.UnknownHostException" %>

<%! DB db; DBCollection coll; MongoClient mongo; BasicDBObject document;%>

<%! String btn;
    String bank;
    String acno;
    String acpin;

```



```

int blncamnt;
String name;
int date;
String month;
int year;
int age;
String gender;
String addrs;
String city;
int pncd;
String cntry;
String state;
String cntcno;
String emgcntcno;
String eid;
String dnum;
String route;
%>
<%

```

```

try
{
btn=request.getParameter("Button");
bank=request.getParameter("Bname");
name=request.getParameter("UNAME");

if("INSERT".equals(btn)||"UPDATE".equals(btn))
{
acno=request.getParameter("ACCNONUMBR");
acpin=request.getParameter("ACCNOPIN");
blncamnt=Integer.parseInt(request.getParameter("BLNCAMNT"));
date=Integer.parseInt(request.getParameter("DATE"));
month=request.getParameter("MONTH");
year=Integer.parseInt(request.getParameter("YEAR"));

```

```

age=Integer.parseInt(request.getParameter("AGE"));
gender=request.getParameter("GENDER");
addrs=request.getParameter("HADDRESS");
dnum=request.getParameter("DNUM");
route=request.getParameter("ROUTE");
city=request.getParameter("CITY");
state=request.getParameter("STATE");
pncd=Integer.parseInt(request.getParameter("PCODE"));
cntry=request.getParameter("COUNTRY");
cntcno=request.getParameter("CNUMBER");
emgcntcno=request.getParameter("ENUMBER");
eid=request.getParameter("EADDRESS");
}

}

catch(Exception e)
{
    out.print(e);
}

try
{
    mongo = new MongoClient("localhost", 27017);
    db = mongo.getDB(bank);
    coll = db.getCollection("ACCOUNTS");
}
catch(Exception e)
{
    out.print(e);
}

if("INSERT".equals(btn)||"UPDATE".equals(btn))
{

    BasicDBObject account = new BasicDBObject();

```

```

        account.put("Number",acno);
        account.put("Pin", acpin);
        account.put("Balance",blncamnt);

BasicDBObject dob = new BasicDBObject();
        dob.put("Date",date);
        dob.put("Month", month);
        dob.put("Year",year);

BasicDBObject contact = new BasicDBObject();
        contact.put("Contact",cntcno);
        contact.put("Emergency", emgcntcno);

BasicDBObject address = new BasicDBObject();
        address.put("Place",addrs);
        address.put("Door",dnum);
        address.put("Route",route);
        address.put("City", city);
        address.put("State", state);
        address.put("Pincode",pncd);
        address.put("Country",cntry);

document = new BasicDBObject();
        document.put("Name", name);
        document.put("Account", account);
        document.put("DOB", dob);
        document.put("Age", age);
        document.put("Gender", gender);
        document.put("Address", address);
        document.put("Email", eid);
        document.put("Contacts", contact);
}

```

```

if("INSERT".equals(btn))
{
    try
    {
        coll.insert(document);
        mongo.close();
        session.setAttribute("msg", "SUCESSFULLY INSERTED");
        response.sendRedirect("AdminMessage.jsp");
    }
    catch(Exception e)
    {
        mongo.close();
        session.setAttribute("msg", "THE USER NAME OR ACCOUNT NUMBER
ALREADY EXIST.TRY AGAIN");
        response.sendRedirect("AdminMessage.jsp");
    }
}
else if("UPDATE".equals(btn))
{
    try
    {
        BasicDBObject search = new BasicDBObject();
        search.put("Account.Number",acno);
        coll.update(search,document);
        mongo.close();
        session.setAttribute("msg", "SUCESSFULLY UPDATED");
        response.sendRedirect("AdminMessage.jsp");
    }
    catch(Exception e)
    {
        mongo.close();
        session.setAttribute("msg", "THE USER NAME OR ACCOUNT NUMBER
ALREADY EXIST.TRY AGAIN");
        response.sendRedirect("AdminMessage.jsp");
    }
}

```

```

    }
}
else if("DELETE".equals(btn))
{
    try
    {
        BasicDBObject search = new BasicDBObject();
        search.put("Account.Number",acno);
        coll.remove(search);
        mongo.close();
        session.setAttribute("msg", "SUCESSFULLY DELETED");
        response.sendRedirect("AdminMessage.jsp");
    }
    catch(Exception e)
    {
        mongo.close();
        session.setAttribute("msg", "THE USER NAME DOESNOT EXIST.TRY
AGAIN");
        response.sendRedirect("AdminMessage.jsp");
    }
}
%>

```

Deposit.jsp

```

<% @page import="java.io.*" %>
<% @page import="java.util.Calendar" %>
<% @page import="com.mongodb.BasicDBObject"%>
<% @page import="com.mongodb.DBObject"%>
<% @page import="com.mongodb.DB" %>
<% @page import="com.mongodb.DBCollection" %>
<% @page import="com.mongodb.DBCursor" %>
<% @page import="com.mongodb.MongoClient" %>

```

```

<% @page import="com.mongodb.MongoException" %>
<% @page import="java.util.Date"%>
<% @page import="java.net.UnknownHostException" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>DEPOSIT DETAILS</title>

<link href="CSS/STYLES4.css" rel="stylesheet" type="text/css" /><style
type="text/css">

</style>
<script type="text/javascript">
</script>
</head>
<%! String getpage;%>
<%try
{
    getpage=session.getAttribute("page").toString();
}
catch(Exception e)
{
    response.sendRedirect("LOGIN.html");
}
%>
<%! DB db; DBCollection coll; MongoClient mongo; BasicDBObject document;
DBCursor cursor;%>
<%!
    String bank;
    String fname;
    String facno;
    String fbank;

```

[illegible]


```

<td align="center" valign="middle"><h6>AMOUNT DEPOSITED</h6></td>
<td align="center" valign="middle"><h6>BALANCE AMOUNT</h6></td>
<td align="center" valign="middle"><h6>DATE OF DEPOSITION</h6></td>
</tr>

```

```

<% try
    {

        if(request.getParameter("Button")!=null)
        {
            bank=request.getParameter("Bname");

try
    {
        mongo = new MongoClient("localhost", 27017);
        db = mongo.getDB(bank);
        coll = db.getCollection("DEPOSIT");
    }
    catch(Exception e)
    {
        mongo.close();
        session.setAttribute("msg", "PLEASE SELECT THE BANK");
        response.sendRedirect("AdminMessage.jsp");
    }

    cursor = coll.find();
    if(cursor!=null)
    {
        while (cursor.hasNext())
        {
            DBObject cur = cursor.next();

            dod=cur.get("DOD").toString();

```

```
DBObject from = (BasicDBObject) cur.get("From");
fbank=from.get("Bank").toString();
fname=from.get("Name").toString();
facno=from.get("Account").toString();
```

```
DBObject to = (BasicDBObject) cur.get("To");
tname=to.get("Name").toString();
tacno=to.get("Account").toString();
```

```
DBObject amount = (BasicDBObject) cur.get("Amount");
damnt=amount.get("Deposited").toString();
bamnt=amount.get("Balance").toString();
```

```
%>
```

```
<tr>
```

```
<td style="font-size: small"><%= fbank %></td>
```

```
<td style="font-size: small"><%= fname %></td>
```

```
<td style="font-size: small"><%= facno %></td>
```

```
<td style="font-size: small"><%= tname %></td>
```

```
<td style="font-size: small"><%= tacno %></td>
```

```
<td style="font-size: small"><%= damnt %></td>
```

```
<td style="font-size: small"><%= bamnt %></td>
```

```
<td style="font-size: small"><%= dod %></td>
```

```
</tr>
```

```
<% }
```

```
}
```

```
else
```

```
{
```

```
mongo.close();
```

```
session.setAttribute("msg", "INVALID USER NAME.TRY AGAIN");
```

```
response.sendRedirect("AdminMessage.jsp");
```

```
}
```

```
}
```

```

    }
    catch(Exception e)
    {
        mongo.close();
        out.println(e);
        //session.setAttribute("msg", "THE USER NAME OR ACCOUNT NUMBER
ALREADY EXIST.TRY AGAIN");
        //response.sendRedirect("AdminMessage.jsp");
    }

%>
</tbody>
</table>

<!-- end .container --></div>
</body>
</html>

```

AdminTransactionDetails.jsp

```

<% @page import="java.io.*" %>
<% @page import="java.util.Calendar" %>
<% @page import="com.mongodb.BasicDBObject"%>
<% @page import="com.mongodb.DBObject"%>
<% @page import="com.mongodb.DB" %>
<% @page import="com.mongodb.DBCollection" %>
<% @page import="com.mongodb.DBCursor" %>
<% @page import="com.mongodb.MongoClient" %>
<% @page import="com.mongodb.MongoException" %>
<% @page import="java.util.Date"%>
<% @page import="java.net.UnknownHostException" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

```

```

<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>TRANSACTION DETAILS</title>

<link href="CSS/STYLES4.css" rel="stylesheet" type="text/css" /><style
type="text/css">

</style>
<script type="text/javascript">

</script>
</head>
<%! String getpage;%>
<%try
{
    getpage=session.getAttribute("page").toString();
}
catch(Exception e)
{
    response.sendRedirect("LOGIN.html");
}
%>
<%! DB db; DBCollection coll; MongoClient mongo; BasicDBObject document;
DBCursor cursor;%>
<%!
    String bank;
    String name;
    String fname;
    String facno;
    String tbank;
    String tname;
    String tacno;
    String tamnt;

```



```
<td align="center" valign="middle"><h6>DATE OF  
TRANSACTION</h6></td>
```

```
</tr>
```

```
<% try
```

```
{
```

```
    if(request.getParameter("Button")!=null)
```

```
    {
```

```
        bank=request.getParameter("Bname");
```

```
    try
```

```
    {
```

```
        mongo = new MongoClient("localhost", 27017);
```

```
        db = mongo.getDB(bank);
```

```
        coll = db.getCollection("TRANSACTION");
```

```
    }
```

```
    catch(Exception e)
```

```
    {
```

```
        mongo.close();
```

```
        session.setAttribute("msg", "PLEASE SELECT THE BANK");
```

```
        response.sendRedirect("AdminMessage.jsp");
```

```
    }
```

```
    cursor = coll.find();
```

```
    if(cursor!=null)
```

```
    {
```

```
        while (cursor.hasNext())
```

```
        {
```

```
            DBObject cur = cursor.next();
```

```
            dot=cur.get("DOT").toString();
```

```
            DBObject from = (BasicDBObject) cur.get("From");
```

```

fname=from.get("Name").toString();
facno=from.get("Account").toString();

```

```

DBObject to = (BasicDBObject) cur.get("To");
tbank=to.get("Bank").toString();
tname=to.get("Name").toString();
tacno=to.get("Account").toString();

```

```

DBObject amount = (BasicDBObject) cur.get("Amount");
tamnt=amount.get("Transacted").toString();
bamnt=amount.get("Balance").toString();

```

```

    %>

```

```

<tr>
<td style="font-size: small"><%= fname %></td>
<td style="font-size: small"><%= facno %></td>
<td style="font-size: small"><%= tbank %></td>
<td style="font-size: small"><%= tname %></td>
<td style="font-size: small"><%= tacno %></td>
<td style="font-size: small"><%= tamnt %></td>
<td style="font-size: small"><%= bamnt %></td>
<td style="font-size: small"><%= dot %></td>
</tr>

```

```

    <% }

```

```

}

```

```

else

```

```

{

```

```

    mongo.close();
    session.setAttribute("msg", "INVALID USER NAME.TRY AGAIN");
    response.sendRedirect("AdminMessage.jsp");
}

```

```

}

```

```

}

```



```

        catch(Exception e)
        {
            mongo.close();
            session.setAttribute("msg", "THE USER NAME OR ACCOUNT NUMBER
ALREADY EXIST.TRY AGAIN");
            response.sendRedirect("AddminMessage.jsp");
        }

%>
</tbody>
</table>

<!-- end .container --></div>
</body>
</html>

```

TransactionDetails.jsp

```

<% @page import="java.io.*" %>
<% @page import="java.util.Calendar" %>
<% @page import="com.mongodb.BasicDBObject"%>
<% @page import="com.mongodb.DBObject"%>
<% @page import="com.mongodb.DB" %>
<% @page import="com.mongodb.DBCollection" %>
<% @page import="com.mongodb.DBCursor" %>
<% @page import="com.mongodb.MongoClient" %>
<% @page import="com.mongodb.MongoException" %>
<% @page import="java.util.Date"%>
<% @page import="java.net.UnknownHostException" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">

```

```

<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>TRANSACTION DETAILS</title>

<link href="CSS/STYLE5.css" rel="stylesheet" type="text/css" /><style
type="text/css"></style>
<script type="text/javascript" src="http://127.0.0.1:8080/jasperserver-
pro/client/visualize.js"></script>
<script type="text/javascript">
visualize({
  auth: {
    name: "jasperadmin",
    password: "jasperadmin",
    organization: "organization_1"
  }
}, function (v) {
  //render report from provided resource
  v("#container").report({
    resource: "/public/Mydata/Reports/AXIS_Bank_Transaction_Report",
    error: handleError
  });
  //show error
  function handleError(err) {
    alert(err.message);
  }
});
</script>

</head>
<%! String user,facno,bank;%>
<%try
{
  facno=session.getAttribute("accno").toString();
  user=session.getAttribute("user").toString();

```



```

{
mongo.close();
session.setAttribute("msg", "PLEASE SELECT THE BANK");
response.sendRedirect("ClientMessage.jsp");
}

```

```

BasicDBObject search = new BasicDBObject();
search.put("From.Account",facno);

```

```

DBCursor cursor = coll.find(search);
if(cursor!=null)

```

```

{
    while (cursor.hasNext())
    {
        DBObject cur = cursor.next();

```

```

        dot=cur.get("DOT").toString();

```

```

        DBObject from = (BasicDBObject) cur.get("From");
        fname=from.get("Name").toString();
        facno=from.get("Account").toString();

```

```

        DBObject to = (BasicDBObject) cur.get("To");
        tbank=to.get("Bank").toString();
        tname=to.get("Name").toString();
        tacno=to.get("Account").toString();

```

```

        DBObject amount = (BasicDBObject) cur.get("Amount");
        tamnt=amount.get("Transacted").toString();
        bamnt=amount.get("Balance").toString();

```

```

        %>

```

```

<tr>

```

```

        <td style="font-size: small"><%= fname %></td>
        <td style="font-size: small"><%= facno %></td>
        <td style="font-size: small"><%= tbank %></td>
        <td style="font-size: small"><%= tname %></td>
        <td style="font-size: small"><%= tacno %></td>
        <td style="font-size: small"><%= tamnt %></td>
        <td style="font-size: small"><%= bamnt %></td>
        <td style="font-size: small"><%= dot %></td>
    </tr>

    <% }

    }
    else
    {
        mongo.close();
        session.setAttribute("msg", "INVALID USER NAME.TRY AGAIN");
        response.sendRedirect("ClientMessage.jsp");
    }
}
catch(Exception e)
{
    mongo.close();
    session.setAttribute("msg", "THE USER NAME OR ACCOUNT NUMBER
ALREADY EXIST.TRY AGAIN");
    response.sendRedirect("ClientMessage.jsp");
}

%>
</tbody>
</table>

<!-- end .container -->
</div>
<div class="footer">
<p>&nbsp;</p>

```

```
</div>
</body>
</html>
```

Transaction.jsp

```
<% @page import="java.io.*" %>
<% @page import="java.util.Calendar" %>
<% @page import="com.mongodb.BasicDBObject"%>
<% @page import="com.mongodb.DBObject"%>
<% @page import="com.mongodb.DB" %>
<% @page import="com.mongodb.DBCollection" %>
<% @page import="com.mongodb.DBCursor" %>
<% @page import="com.mongodb.MongoClient" %>
<% @page import="com.mongodb.MongoException" %>
<% @page import="java.util.Date"%>
<% @page import="java.net.UnknownHostException" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>TRANSACTION</title>
<link href="SpryAssets/SpryValidationTextField.css" rel="stylesheet" type="text/css"
/>
<link href="CSS/STYLES2.css" rel="stylesheet" type="text/css" /><style
type="text/css">

</style>
<link href="jQueryAssets/SpryValidationSelect.css" rel="stylesheet" type="text/css"
/>
<script src="SpryAssets/SpryValidationTextField.js" type="text/javascript"></script>
<script src="jQueryAssets/SpryValidationSelect.js" type="text/javascript"></script>
```

```

<script type="text/javascript">
function MM_preloadImages() { //v3.0
    var d=document; if(d.images){ if(!d.MM_p) d.MM_p=new Array();
        var i,j=d.MM_p.length,a=MM_preloadImages.arguments; for(i=0; i<a.length; i++)
            if (a[i].indexOf("#")!=0){ d.MM_p[j]=new Image; d.MM_p[j++].src=a[i];}}
    }

function blncamnt(){ //v2.0
    var tamnt=document.getElementById("TAMOUNT").value;
    var samnt=document.getElementById("SAMOUNT").value;
    var bamnt=samnt-tamnt;
    document.TRANSACTION.BAMOUNT.value = bamnt;
    }
function MM_popupMsg(msg) { //v1.0
    alert(msg);
}
</script>
</head>

    <%! String user,bank,facno;%>
<%try
{
    facno=session.getAttribute("accno").toString();
    user=session.getAttribute("user").toString();
    bank=session.getAttribute("bank").toString();
}
catch(Exception e)
{
    response.sendRedirect("LOGIN.html");
}
%>
<%! DB db; DBCollection coll; MongoClient mongo; BasicDBObject document;%>
<%!

```



```

String samnt;
String tamnt;
%>
<%
try
{
    try
    {
        mongo = new MongoClient("localhost", 27017);
        db = mongo.getDB(bank);
        coll = db.getCollection("ACCOUNTS");
    }
    catch(Exception e)
    {
        out.print(e);
    }

    BasicDBObject search = new BasicDBObject();
    search.put("Account.Number", facno);

    DBCursor cursor = coll.find(search);

    if(cursor!=null)
    {
        while (cursor.hasNext())
        {
            DBObject cur = cursor.next();

            user=cur.get("Name").toString();
            DBObject account = (BasicDBObject) cur.get("Account");
            facno=account.get("Number").toString();
            samnt=account.get("Balance").toString();
            mongo.close();
        }
    }
}

```

```
}  
else  
{  
    out.println("Invalid user or bank");  
}  
  
}  
  
catch(Exception e)  
{  
    mongo.close();  
    out.println(e);  
}  
  
%>  
  
<body>  
  
<div class="container">  
  
    <div class="header">EASY ONLINE BANKING</div>  
  
    <div class="sidebar1 ">  
  
        <ul class="nav">  
  
            <li><a href="DETAILS.jsp">Account Details</a></li>  
            <li><a href="TRANSACTION.jsp">Transaction </a></li>  
            <li><a href="DEPOSIT_DETAIL.jsp">Deposit Details</a></li>  
  
        </ul>  
  
        <p>&nbsp;</p>  
  
        <!-- end .sidebar1 --></div>  
  
    <div class="content">  
  
        <h1>TRANSACTION
```

```

        <input name="submit" type="submit" formnovalidate="formnovalidate"
class="DELETE" id="submit" form="viewall2" formaction="logout.jsp"
value="SIGN OUT" />
    </h1>
    <hr /><div id="viewall">
        <form id="viewall2" name="viewall2" method="post"
action="TRANSACTION_DETAILS.jsp">
            <input name="submit3" type="submit" class="SUBMIT" id="submit3"
form="viewall2" formaction="TRANSACTION_DETAILS.jsp"
formmethod="POST" value="VIEW ALL" />
        </form>
    </div>
    <form action="TRANSACTION.jsp" method="post" name="TRANSACTION"
target="_self" id="TRANSACTION">
        <table width="1390" height="712" border="0">
            <tr>
                <td height="61" colspan="3"><div align="center" class="HEAD">FROM
ACCOUNT</div></td>
            </tr>
            <tr>
                <td height="61"><h3>BANK NAME</h3></td>
                <td><h3>:</h3></td>
                <td><input name="FBname" type="text" class="CONTACT" id="FBname"
value="<%=bank%>" readonly="readonly" /></td>
            </tr>
            <tr>
                <td height="61"><h3>ACCOUNT USER</h3></td>
                <td><h3>:</h3></td>
                <td><label for="UNAME"></label>
                    <span id="SPRYID3">
                        <input name="UNAME" type="text" value="<%=user%>"
class="CONTACT" id="UNAME" readonly="readonly" />
                        <span class="textfieldRequiredMsg">Required.</span><span
class="textfieldInvalidFormatMsg">Only Integer is allowed.</span></span></td>

```

```

</tr>
<tr>
  <td height="61"><h3>ACCOUNT NUMBER</h3></td>
  <td><h3>:</h3></td>
  <td><label for="FACNUMBER"></label>
    <span id="SPRYID">
      <input name="FACNUMBER" type="text" value="<%=facno%>"
class="CONTACT" id="FACNUMBER" readonly="readonly" />
      <span class="textfieldRequiredMsg">Required.</span><span
class="textfieldInvalidFormatMsg">Only Integer is allowed.</span></span></td>
</tr>
<tr>
  <td width="278" height="61"><h3>ACCNOUNT PIN</h3></td>
  <td width="39"><h3>:</h3></td>
  <td width="571"><span id="SPRYCNAME">
    <label for="NAME2">
      <input name="ACPIN" type="password" class="CONTACT" id="ACPIN" />
    </label>
    <span class="textfieldRequiredMsg">PIN is Required.</span><span
class="textfieldInvalidFormatMsg">Only Integer is allowed..</span><span
class="textfieldMaxValueMsg">Only 4 number is allowed.</span></span></td>
</tr>
<tr>
  <td height="61"><h3>SAVINGS AMOUNT</h3></td>
  <td><h3>:</h3></td>
  <td><input name="SAMOUNT" type="number" value="<%=samnt%>"
class="CONTACT" id="SAMOUNT" readonly="readonly" /></td>
</tr>
<tr>
  <td height="61"><h3>TRANSACT AMOUNT</h3></td>
  <td><h3>:</h3></td>
  <td><span id="SPRYCNAME5">
    <label for="NAME4">

```

```

        <input name="TAMOUNT" type="text" class="CONTACT"
id="TAMOUNT" onblur="blncamnt()" />
    </label>
    <span class="textfieldRequiredMsg">Required.</span><span
class="textfieldInvalidFormatMsg">Only Integer is allowed.</span></span></td>
</tr>
<tr>
    <td height="61"><h3>BALANCE AMOUNT</h3></td>
    <td><h3>:</h3></td>
    <td><input name="BAMOUNT" type="number" required="required"
class="CONTACT" id="BAMOUNT" readonly="readonly" /></td>
</tr>
<tr>
    <td height="65" colspan="3"><div align="center" class="HEAD">TO
ACCOUNT</div></td>
</tr>
<tr>
    <td height="61"><h3>BANK NAME</h3></td>
    <td><h3>:</h3></td>
    <td><span id="spryselect1">
        <select name="Bname" required="required" id="Bname">
            <option>Select Bank</option>
            <option value="AXIS">Axis</option>
            <option value="SBI">Sbi</option>
            <option value="PUNJAB">Punjab</option>
            <option value="ICICI">ICICI</option>
        </select>
        <span class="selectRequiredMsg">Please select the bank.</span></span></td>
</tr>
<tr>
    <td height="61"><h3>ACCOUNT NUMBER</h3></td>
    <td><h3>:</h3></td>
    <td><label for="TACNUMBER"></label>
        <span id="SPRYID2">

```

```

        <input name="TACNUMBER" type="text" class="CONTACT"
id="TACNUMBER" />

        <span class="textfieldRequiredMsg">Required.</span><span
class="textfieldInvalidFormatMsg">Only Integer is allowed.</span></span></td>
    </tr>
</table>
</form>

<table width="901" height="88" border="0">
    <tr>
        <td><div align="center">
            <input name="Button" type="submit" class="SUBMIT" id="Button"
form="TRANSACTION" formaction="clntwrk.jsp" formmethod="POST"
value="TRANSACT" />
        </div></td>
        <td><div align="center">
            <input name="CALCEL" type="reset" class="CANCEL" id="CALCEL"
form="ACCOUNT_DETAILS" value="CANCLE" />
        </div></td>
    </tr>
</table>
<p>&nbsp;</p>
<!-- end .content --></div>
<div class="footer">
    <p>&nbsp;</p>
<!-- end .container --></div>
</div>
<script type="text/javascript">
var sprytextfield2 = new Spry.Widget.ValidationTextField("SPRYCNAME",
"integer", { validateOn:["change"]});
var sprytextfield3 = new Spry.Widget.ValidationTextField("SPRYID", "integer",
{isRequired:false});
var sprytextfield2 = new Spry.Widget.ValidationTextField("SPRYCNAME5",
"integer", { validateOn:["change"]});

```

```

var sprytextfield3 = new Spry.Widget.ValidationTextField("SPRYID2", "integer",
{ validateOn:["change"]});
var spryselect1 = new Spry.Widget.ValidationSelect("spryselect1",
{ validateOn:["change"]});
var sprytextfield3 = new Spry.Widget.ValidationTextField("SPRYID3", "none",
{isRequired:false});
</script>
</body>
</html>

```

AdminDepositDetails.jsp

```

<% @page import="java.io.*" %>
<% @page import="java.util.Calendar" %>
<% @page import="com.mongodb.BasicDBObject"%>
<% @page import="com.mongodb.DBObject"%>
<% @page import="com.mongodb.DB" %>
<% @page import="com.mongodb.DBCollection" %>
<% @page import="com.mongodb.DBCursor" %>
<% @page import="com.mongodb.MongoClient" %>
<% @page import="com.mongodb.MongoException" %>
<% @page import="java.util.Date"%>
<% @page import="java.net.UnknownHostException" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>DEPOSIT DETAILS</title>

<link href="CSS/STYLES4.css" rel="stylesheet" type="text/css" /><style
type="text/css">

```

```

</style>
<script type="text/javascript">
</script>
</head>
<%! String getpage;%>
<%try
{
    getpage=session.getAttribute("page").toString();
}
catch(Exception e)
{
    response.sendRedirect("LOGIN.html");
}
%>
<%! DB db; DBCollection coll; MongoClient mongo; BasicDBObject document;
DBCursor cursor;%>
<%!
    String bank;
    String fname;
    String facno;
    String fbank;
    String tname;
    String tacno;
    String damnt;
    String bamnt;
    String dod;
    %>

<body>
<div class="container">
    <div class="header">EASY ONLINE BANKING</div>
    <div class="sidebar1">
        <ul class="nav">
            <li><a href="ACCOUNT DETAILS.jsp">Account Details</a></li>

```



```

    <li><a href="TRANSACTION_DETAIL_ADMIN.jsp">Transaction
Details</a></li>

    <li class="active"><a href="DEPOSIT_DETAIL_ADMIN.jsp">Deposit
Details</a>    <!-- end .sidebar1 --></li>

</ul>

</div>

```

```

<li class="active"><a href="DEPOSIT_DETAIL_ADMIN.jsp">Deposit
Details</a> <!-- end .sidebar1 --></li>

</ul>

</div>

```

[illegible]

```
<form action="DEPOSIT_DETAIL_ADMIN.jsp" method="post"
name="DEPOSIT" target="_self" id="DEPOSIT">

<table width="902" height="132" border="0" align="center">

<tr>

<td height="61"><h3>BANK NAME</h3></td>

<td><h3>:</h3></td>

<td><select name="Bname" required="required" id="Bname">

<option value="" selected="selected">Select Bank</option>

<option value="AXIS">Axis</option>

<option value="SBI">Sbi</option>
```

[illegible]

```

{
    mongo = new MongoClient("localhost", 27017);
    db = mongo.getDB(bank);
    coll = db.getCollection("DEPOSIT");
}
catch(Exception e)
{
    mongo.close();
    session.setAttribute("msg", "PLEASE SELECT THE BANK");
    response.sendRedirect("AdminMessage.jsp");
}

cursor = coll.find();
if(cursor!=null)
{
    while (cursor.hasNext())
    {
        DBObject cur = cursor.next();

        dod=cur.get("DOD").toString();

        DBObject from = (BasicDBObject) cur.get("From");
        fbank=from.get("Bank").toString();
        fname=from.get("Name").toString();
        facno=from.get("Account").toString();

        DBObject to = (BasicDBObject) cur.get("To");
        tname=to.get("Name").toString();
        tacno=to.get("Account").toString();

        DBObject amount = (BasicDBObject) cur.get("Amount");
        damnt=amount.get("Deposited").toString();
        bamnt=amount.get("Balance").toString();
    }
}

```

```

%>

<tr>
<td style="font-size: small"><%= fbank %></td>
<td style="font-size: small"><%= fname %></td>
<td style="font-size: small"><%= facno %></td>
<td style="font-size: small"><%= tname %></td>
<td style="font-size: small"><%= tacno %></td>
<td style="font-size: small"><%= damnt %></td>
<td style="font-size: small"><%= bamnt %></td>
<td style="font-size: small"><%= dod %></td>
</tr>

<% }

}
else
{
    mongo.close();
    session.setAttribute("msg", "INVALID USER NAME.TRY AGAIN");
    response.sendRedirect("AdminMessage.jsp");
}

}

}
catch(Exception e)
{
    mongo.close();
    out.println(e);
    //session.setAttribute("msg", "THE USER NAME OR ACCOUNT NUMBER
ALREADY EXIST.TRY AGAIN");
    //response.sendRedirect("AdminMessage.jsp");
}

%>

</tbody>
</table>

```

```

        <!-- end .container --></div>
</body>
</html>

```

Details.jsp

```

<% @page import="java.io.*" %>
<% @page import="java.util.Calendar" %>
<% @page import="com.mongodb.BasicDBObject"%>
<% @page import="com.mongodb.DBObject"%>
<% @page import="com.mongodb.DB" %>
<% @page import="com.mongodb.DBCollection" %>
<% @page import="com.mongodb.DBCursor" %>
<% @page import="com.mongodb.MongoClient" %>
<% @page import="com.mongodb.MongoException" %>
<% @page import="java.util.Date"%>
<% @page import="java.net.UnknownHostException" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
<title>ACCOUNT DETAILS</title>

<link href="CSS/STYLES2.css" rel="stylesheet" type="text/css" /><style
type="text/css">

</style>
<script type="text/javascript">

function Age(){ //v2.0
    var a=document.getElementById('YEAR').value;

```

```

var nowdate=new Date();
var nowyr=nowdate.getFullYear();
var i=nowyr-a;
document.ACCOUNT_DETAILS.AGE.value = i;
}
</script>
</head>
<%! String accno,bank;%>
<%try
{
    accno=session.getAttribute("accno").toString();
    bank=session.getAttribute("bank").toString();
}
catch(Exception e)
{
    response.sendRedirect("LOGIN.html");
}
%>
<%! DB db; DBCollection coll; MongoClient mongo; BasicDBObject document;%>
<%!
    String user;
    String acno;
    String acpin;
    String blncamnt;
    String date;
    String month;
    String year;
    String age;
    String gender;
    String addrs;
    String dnum;
    String route;
    String city;
    String pncd;

```

```

String cntry;
String state;
String cntcno;
String emgcntcno;
String eid;
%>
<%
try
{
    try
    {
        mongo = new MongoClient("localhost", 27017);
        db = mongo.getDB(bank);
        coll = db.getCollection("ACCOUNTS");
    }
    catch(Exception e)
    {
        out.print(e);
    }

    BasicDBObject search = new BasicDBObject();
    search.put("Account.Number", accno);

    DBCursor cursor = coll.find(search);
    while (cursor.hasNext())
    {
        DBObject cur = cursor.next();

        user=cur.get("Name").toString();
        age=cur.get("Age").toString();
        eid=cur.get("Email").toString();

        DBObject account = (BasicDBObject) cur.get("Account");
        acno=account.get("Number").toString();

```

```

        acpin=account.get("Pin").toString();
        blncamnt=account.get("Balance").toString();

        DBObject address = (BasicDBObject) cur.get("Address");
        addrs=address.get("Place").toString();
        dnum=address.get("Door").toString();
        route=address.get("Route").toString();
        city=address.get("City").toString();
        state=address.get("State").toString();
        pncd=address.get("Pincode").toString();
        cntry=address.get("Country").toString();

        DBObject contact = (BasicDBObject) cur.get("Contacts");
        cntcno=contact.get("Contact").toString();
        emgcntcno=contact.get("Emergency").toString();
    }
}
catch(Exception e)
{
    out.println(e);
}

%>
<body>
<div class="container">
    <div class="header">EASY ONLINE BANKING</div>
    <div class="sidebar1">
        <ul class="nav">
            <li class="active"><a href="DETAILS.jsp">Account Details</a></li>
            <li><a href="TRANSACTION.jsp">Transaction</a></li>
            <li><a href="DEPOSIT_DETAIL.jsp">Deposit Details</a></li>
        </ul>
        <p>&nbsp;</p>
    <!-- end .sidebar1 --></div>

```


[illegible]

```

        <td><input name="UNAME" type="text" required="required" class="FIELD"
id="UNAME" autocomplete="off" value="<%= user %>" readonly="readonly"
/></td>

</tr>

<tr>

<td height="53"><h3>DATE OF BIRTH</h3></td>

<td><h3>:</h3></td>

<td><table width="559" height="52" border="0">

<tr>

<td width="153"><select name="DATE" required="required"
class="D_O_B" id="DATE">
<option selected="selected">DATE</option>
<%
                                for(int i=1;i<=31;i++)
                                {
                                    %>
<option value="<%= i %>"><%= i %></option>
<% } %>
</select></td>

<td width="182"><select name="MONTH" required="required"
class="D_O_B" id="MONTH">
<option selected="selected">MONTH</option>
<option value="JANUARY">JANUARY</option>
<option value="FEBRUARY">FEBRUARY</option>
<option value="MARCH">MARCH</option>
<option value="APRIL">APRIL</option>
<option value="MAY">MAY</option>
<option value="JUNE">JUNE</option>
<option value="JULY">JULY</option>
<option value="AUGUST">AUGUST</option>
<option value="SEPTEMBER">SEPTEMBER</option>
<option value="OCTOBER">OCTOBER</option>
<option value="NOVEMBER">NOVEMBER</option>
<option value="DECEMBER">DECEMBER</option>

```



```

<tr>
  <td height="61" colspan="3"><div align="center" class="HEAD">ACCOUNT
INFORMATION</div></td>
</tr>
<tr>
  <td height="61"><h3>ACCOUNT NUMBER</h3></td>
  <td><h3>:</h3></td>
  <td>
    <input name="ACCNONUMBR" type="number" required="required"
class="CONTACT" id="ACCNONUMBR" value="<%=acno%>"
readonly="readonly" />
  </td>
</tr>
<tr>
  <td width="278" height="61"><h3>ACCOUNT PIN</h3></td>
  <td width="39"><h3>:</h3></td>
  <td width="571">
    <input name="ACCNOPIN" type="number" required="required"
class="CONTACT" id="ACCNOPIN" min="1000" value="<%=acpin%>"
readonly="readonly" /></td>
</tr>
  <td height="61"><h3>BALANCE AMOUNT</h3></td>
  <td><h3>:</h3></td>
  <td><input name="BLNCAMNT" type="number" required="required"
class="CONTACT" id="BLNCAMNT" value="<%=blncamnt%>"
readonly="readonly"/></td>
</tr>
<tr>
  <td height="59" colspan="3"><div align="center" class="HEAD">CONTACT
INFORMATION</div></td>
</tr>
<tr>
  <td height="59"><h3> ADDRESS </h3></td>
  <td><h3>:</h3></td>

```

```

        <td><div id="locationField">
            <input name="HADDRESS" type="text" class="GEOADDRESS"
id="autocomplete" placeholder="Enter your address"
            onfocus="geolocate()" value="<%=addrs%>" />
        </input>
        </div></td>
    </tr>
    <tr>
        <td height="59"><h3>DOOR NUMBER </h3></td>
        <td><h3>:</h3></td>
        <td><input name="DNUM" type="text" required="required"
value="<%=dnum%>" class="CONTACT" id="street_number" /></td>
    </tr>
    <tr>
        <td height="59"><h3>STREET ROUTE </h3></td>
        <td><h3>:</h3></td>
        <td><textarea name="ROUTE" cols="45" rows="5" required="required"
class="TEXTAREA" id="route"><%=route%></textarea></td>
    </tr>
    <tr>
        <td height="59"><h3>CITY</h3></td>
        <td><h3>:</h3></td>
        <td><input name="CITY" type="text" required="required" class="CONTACT"
id="locality" value="<%=city%>" readonly="readonly" /></td>
    </tr>
    <tr>
        <td height="59"><h3>STATE</h3></td>
        <td><h3>:</h3></td>
        <td><input name="STATE" type="text" required="required"
class="CONTACT" id="administrative_area_level_1" value="<%=state%>"
readonly="readonly" /></td>
    </tr>
    <tr>
        <td height="59"><h3>ZIP CODE</h3></td>

```

```

        <td><h3>:</h3></td>

        <td><input name="PCODE" type="number" required="required"
class="CONTACT" id="postal_code" value="<%=pncd%>" /></td>

    </tr>

    <tr>

        <td height="59"><h3>COUNTRY</h3></td>

        <td><h3>:</h3></td>

        <td><input name="COUNTRY" type="text" required="required"
class="CONTACT" id="country" value="<%=cntry%>" readonly="readonly"
/></td>

    </tr>

    <tr>

        <td height="59"><h3>CONTACT NUMBER</h3></td>

        <td><h3>:</h3></td>

        <td><input name="CNUMBER" type="number" required="required"
class="CONTACT" value="<%=cntcno%>" id="CNUMBER" /></td>

    </tr>

    <tr>

        <td height="59"><h3>EMERGENCY CONTACT</h3></td>

        <td><h3>:</h3></td>

        <td><input name="ENUMBER" type="number" class="CONTACT"
value="<%=emgcntcno%>" id="ENUMBER" /></td>

    </tr>

    <tr>

        <td height="59"><h3>EMAIL ADDRESS</h3></td>

        <td><h3>:</h3></td>

        <td><input name="EADDRESS" type="email" required="required"
value="<%= eid %>" class="EMAIL" id="EADDRESS" autocomplete="off" /></td>

    </tr>

    <script>
var autocomplete;
var componentForm = {
    street_number: 'short_name',
    route: 'long_name',

```

```

locality: 'long_name',
administrative_area_level_1: 'long_name',
country: 'long_name',
postal_code: 'short_name'
};

```

```

function initAutocomplete() {
    autocomplete = new google.maps.places.Autocomplete(
(document.getElementById('autocomplete')),
    {types: ['geocode']}));
    autocomplete.addListener('place_changed', fillInAddress);
}

```

```

function fillInAddress() {
    var place = autocomplete.getPlace();

    for (var component in componentForm) {
        document.getElementById(component).value = "";
        document.getElementById(component).disabled = false;
    }

    for (var i = 0; i < place.address_components.length; i++) {
        var addressType = place.address_components[i].types[0];
        if (componentForm[addressType]) {
            var val = place.address_components[i][componentForm[addressType]];
            document.getElementById(addressType).value = val;
        }
    }
}

```

```

function geolocate() {
    if (navigator.geolocation) {
        navigator.geolocation.getCurrentPosition(function(position) {
            var geolocation = {
                lat: position.coords.latitude,
                lng: position.coords.longitude
            }

```



```
<div class="footer">  
  <p>&nbsp;</p>  
  <!-- end .container --></div>  
</div>  
</body>  
</html>
```

8.2 OUTPUT SCREEN

8.2.1 WELCOME PAGE (Banner.html)

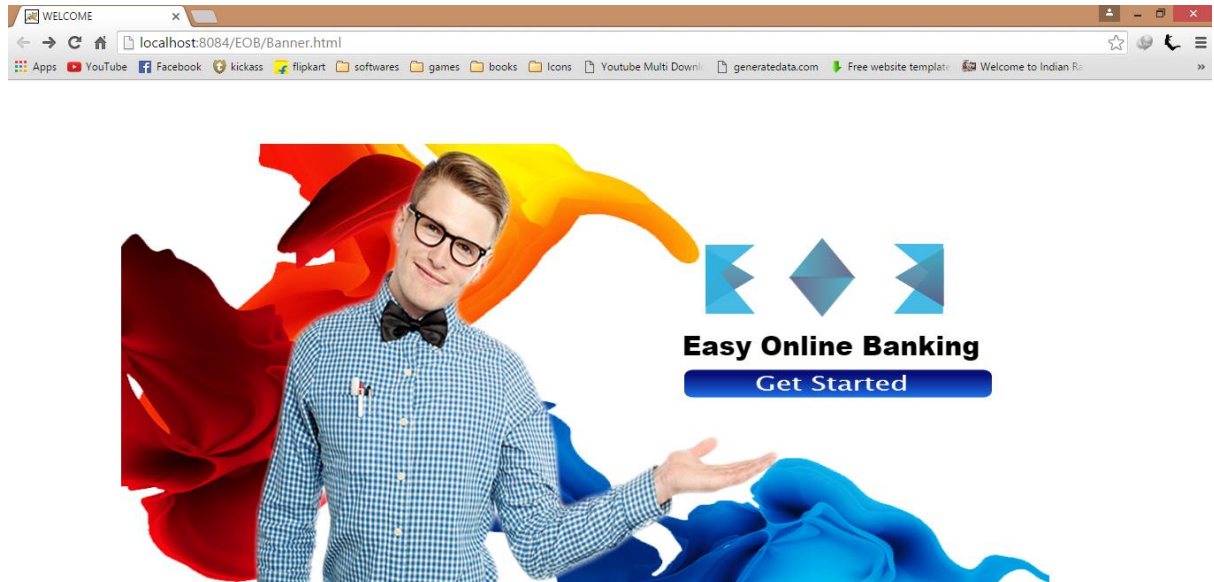


Fig: 8.2.1.1

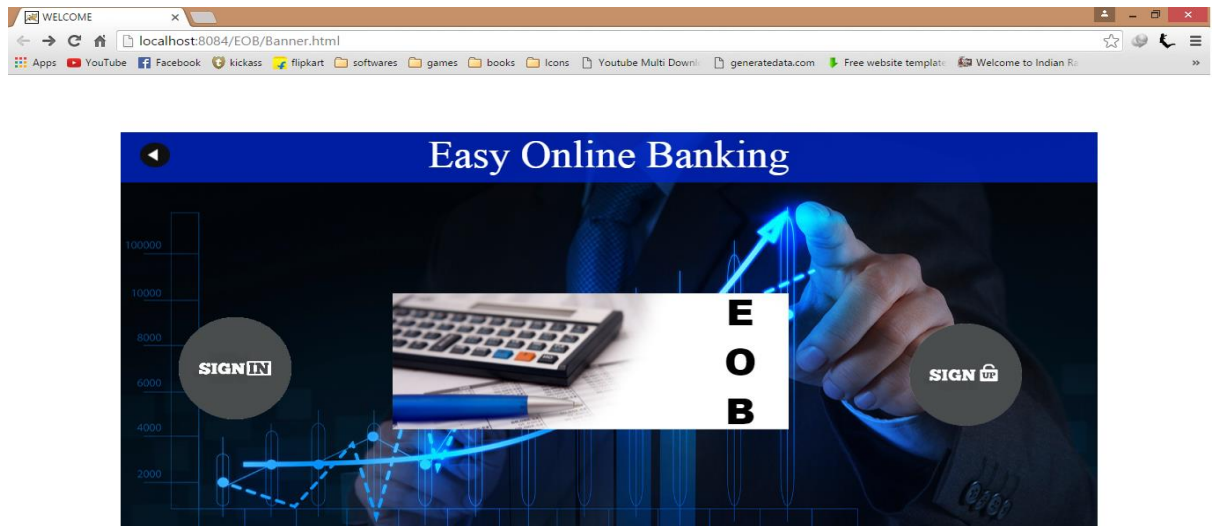
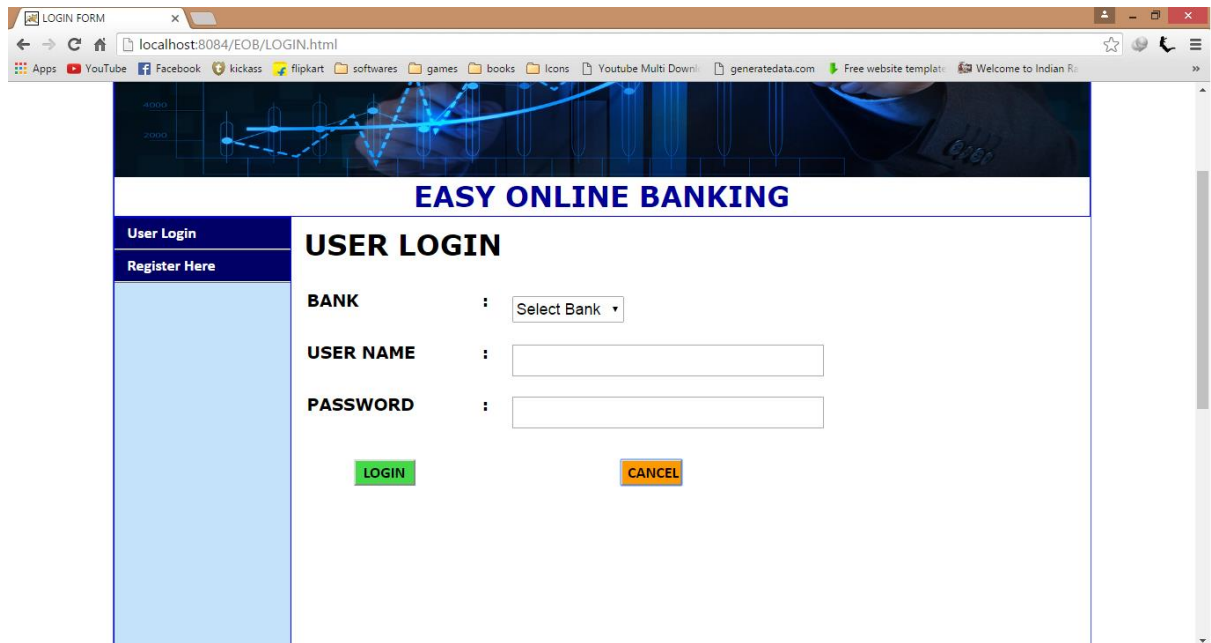


Fig: 8.2.1.2

This page is the Welcome page or start page of the project.

8.2.2 LOGIN PAGE (Login.html)

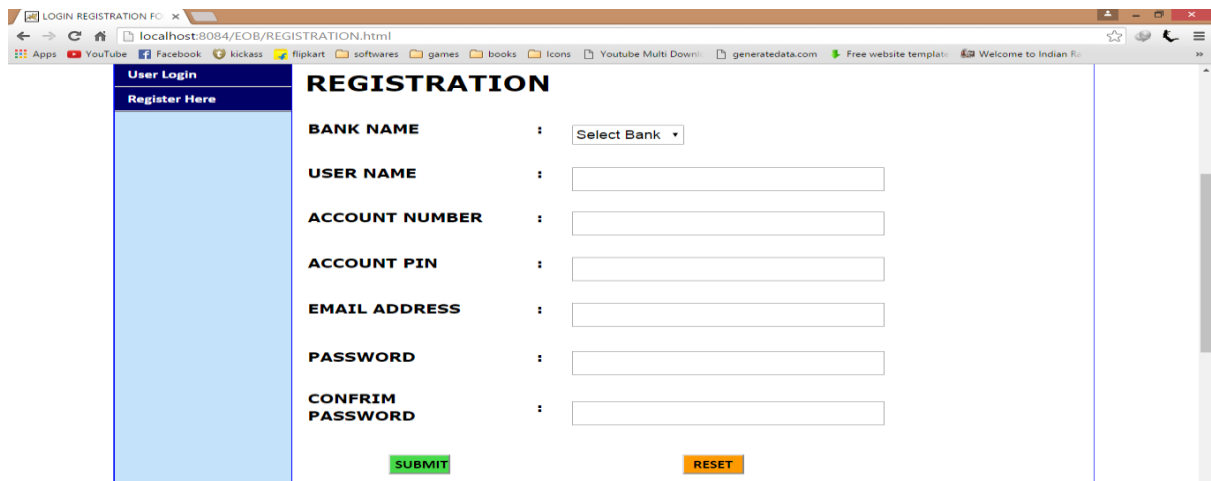


The screenshot shows a web browser window with the address bar displaying 'localhost:8084/EOB/LOGIN.html'. The page has a header with a blue background and a line graph. Below the header, there is a navigation bar with 'User Login' and 'Register Here' links. The main content area is titled 'EASY ONLINE BANKING' and 'USER LOGIN'. It contains a form with the following fields: 'BANK' (a dropdown menu with 'Select Bank' as the selected option), 'USER NAME' (a text input field), and 'PASSWORD' (a text input field). Below the form, there are two buttons: 'LOGIN' (green) and 'CANCEL' (orange).

Fig: 8.2.2.1

This page is used to login to both admin page and the user page.

8.2.3 REGISTRATION PAGE (Registration.html)



The screenshot shows a web browser window with the address bar displaying 'localhost:8084/EOB/REGISTRATION.html'. The page has a header with a blue background and a line graph. Below the header, there is a navigation bar with 'User Login' and 'Register Here' links. The main content area is titled 'EASY ONLINE BANKING' and 'REGISTRATION'. It contains a form with the following fields: 'BANK NAME' (a dropdown menu with 'Select Bank' as the selected option), 'USER NAME' (a text input field), 'ACCOUNT NUMBER' (a text input field), 'ACCOUNT PIN' (a text input field), 'EMAIL ADDRESS' (a text input field), 'PASSWORD' (a text input field), and 'CONFIRM PASSWORD' (a text input field). Below the form, there are two buttons: 'SUBMIT' (green) and 'RESET' (orange).

Fig: 8.2.3.1

This page is used to register for the Easy Online Banking.

8.2.4 ADMIN PAGE (ACCOUNT_DETAILS.jsp)

The screenshot shows a web browser window with the title "ACCOUNT DETAILS". The address bar shows "localhost:8084/EOB/ACCOUNT%20DETAILS.jsp". The page has a sidebar with a list of applications (Apps, YouTube, Facebook, kickass, flipkart, softwares, games, books, Icons, Youtube Multi Downl, generatedata.com, Free website templat, Welcome to Indian R). The main content area is divided into two sections: "ACCOUNT INFORMATION" and "CONTACT INFORMATION".

ACCOUNT INFORMATION

ACCOUNT NUMBER :

ACCOUNT PIN :

BALANCE AMOUNT :

CONTACT INFORMATION

HOME ADDRESS :

CITY :

STATE :

PIN CODE :

Fig: 8.2.4.1

The screenshot shows a web browser window with the title "ACCOUNT DETAILS". The address bar shows "localhost:8084/EOB/ACCOUNT%20DETAILS.jsp". The page has a sidebar with a list of applications (Apps, YouTube, Facebook, kickass, flipkart, softwares, games, books, Icons, Youtube Multi Downl, generatedata.com, Free website templat, Welcome to Indian R). The main content area is divided into three sections: "ACCOUNT DETAILS", "PERSONAL INFORMATION", and "ACCOUNT INFORMATION".

ACCOUNT DETAILS

BANK NAME :

PERSONAL INFORMATION

USER NAME :

DATE OF BIRTH :

AGE :

GENDER : ☐ MALE ☐ FEMALE

ACCOUNT INFORMATION

ACCOUNT NUMBER :

ACCOUNT PIN :

Fig: 8.2.4.2

ACCOUNT DETAILS

localhost:8084/EOB/ACCOUNT%20DETAILS.jsp

CITY :

STATE :

PIN CODE :

COUNTRY :

CONTACT NUMBER :

EMERGENCY CONTACT :

EMAIL ADDRESS :

INSERT SEARCH UPDATE CANCEL DELETE

Fig: 8.2.4.3

This page is used to do crud operation by the administrator.

8.2.5 CLIENT PAGE (DETAILS.jsp)

ACCOUNT DETAILS

localhost:8084/EOB/DETAILS.jsp

Account Details Transaction

ACCOUNT DETAILS SIGN OUT

BANK NAME :

PERSONAL INFORMATION

USER NAME :

DATE OF BIRTH :

AGE :

GENDER : ☒ MALE ☐ FEMALE

ACCOUNT INFORMATION

ACCOUNT NUMBER :

Fig: 8.2.5.1

ACCOUNT DETAILS

localhost:8084/EOB/DETAILS.jsp

Apps YouTube Facebook kickass flipkart softwares games books Icons Youtube Multi Downl generatedata.com Free website templat: Welcome to Indian R

ACCOUNT PIN : 2222

BALANCE AMOUNT : 19000

CONTACT INFORMATION

HOME ADDRESS : A11, Vigenesh empire appartments, Woraiyur

CITY : Trichy

STATE : Tamilnadu

PIN CODE : 620003

COUNTRY : India

CONTACT NUMBER : 9942743109

Fig: 8.2.5.2

ACCOUNT DETAILS

localhost:8084/EOB/DETAILS.jsp

Apps YouTube Facebook kickass flipkart softwares games books Icons Youtube Multi Downl generatedata.com Free website templat: Welcome to Indian R

CITY : Trichy

STATE : Tamilnadu

PIN CODE : 620003

COUNTRY : India

CONTACT NUMBER : 9942743109

EMERGENCY CONTACT :

EMAIL ADDRESS : Mycount.siva@gmail.com

UPDATE **CANCEL**

Fig: 8.2.5.3

This page is used to view and update the personal details by the client user.

8.2.6 TRANSACTION PAGE (TRANSACTION.jsp)

The screenshot shows a web browser window with the URL `localhost:8084/EOB/TRANSACTION.jsp`. The page has a sidebar with 'Account Details' and 'Transaction' (selected). The main content area is titled 'TRANSACTION' and includes a 'SIGN OUT' button. A 'VIEW ALL' button is present. The 'FROM ACCOUNT' section contains the following fields:

BANK NAME	:	<input type="text" value="AXIS"/>
ACCOUNT USER	:	<input type="text" value="Siva"/>
ACCOUNT NUMBER	:	<input type="text" value="1408"/>
ACCOUNT PIN	:	<input type="text"/>
SAVINGS AMOUNT	:	<input type="text" value="19000"/>
TRANSACT AMOUNT	:	<input type="text"/>
BALANCE AMOUNT	:	<input type="text"/>

Fig: 8.2.6.1

The screenshot shows the same web browser window, but the 'TO ACCOUNT' section is visible. It contains the following fields:

ACCOUNT NUMBER	:	<input type="text" value="1408"/>
ACCOUNT PIN	:	<input type="text"/>
SAVINGS AMOUNT	:	<input type="text" value="19000"/>
TRANSACT AMOUNT	:	<input type="text"/>
BALANCE AMOUNT	:	<input type="text"/>

Below these fields is a pink header for the 'TO ACCOUNT' section, followed by:

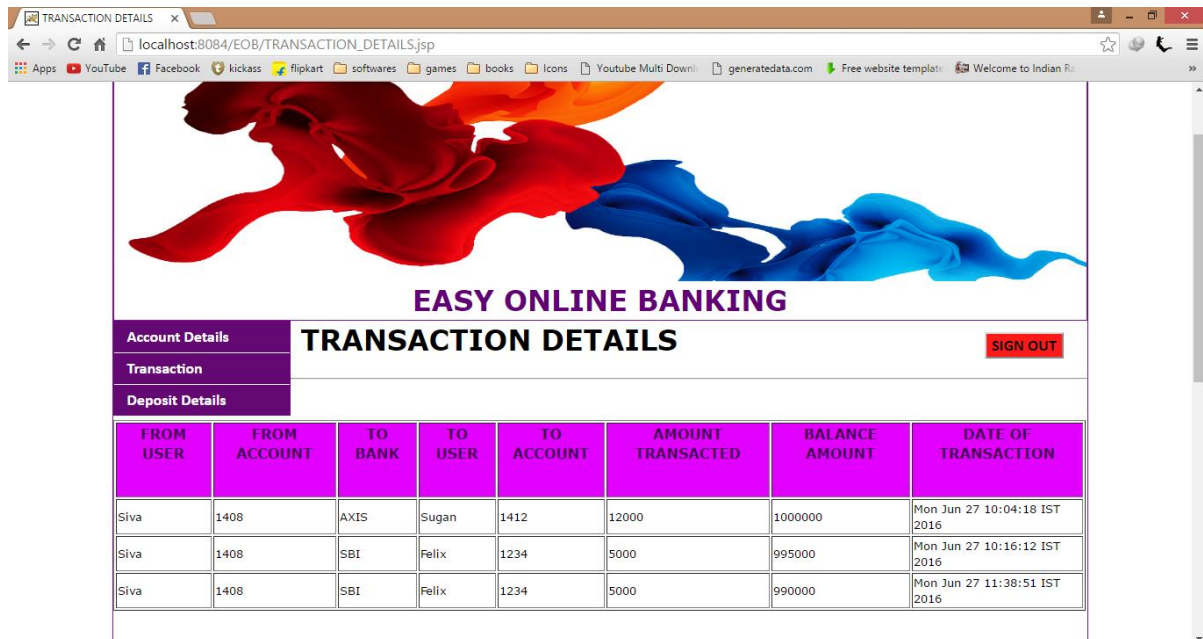
BANK NAME	:	<input type="text" value="Select Bank"/>
ACCOUNT NUMBER	:	<input type="text"/>

At the bottom, there are two buttons: 'TRANSACTION' (green) and 'CANCEL' (orange).

Fig: 8.2.6.2

This page is used to Transact money form one account to other account.

8.2.7 TRANSACTION DETAILS PAGE (TRANSACTION_DETAILS.jsp)

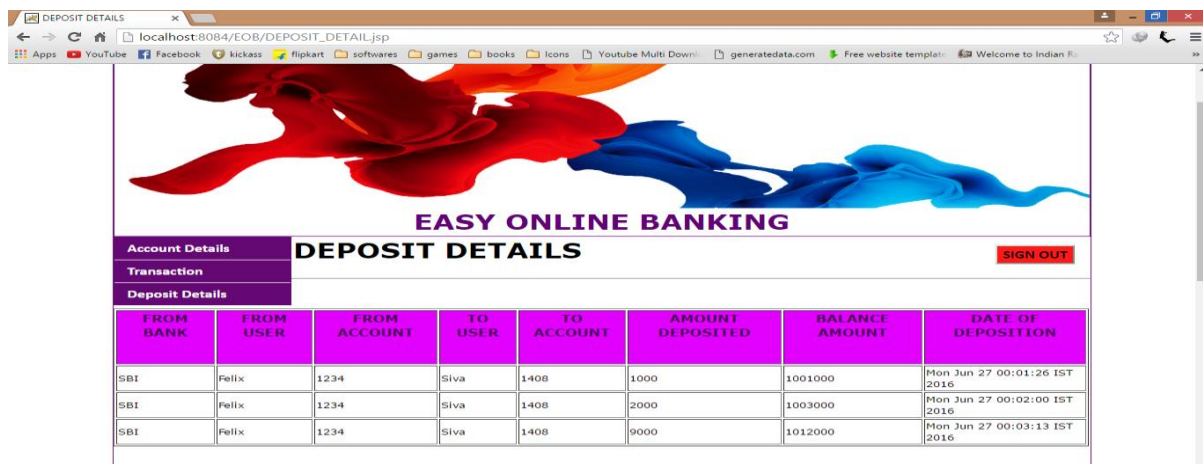


FROM USER	FROM ACCOUNT	TO BANK	TO USER	TO ACCOUNT	AMOUNT TRANSACTED	BALANCE AMOUNT	DATE OF TRANSACTION
Siva	1408	AXIS	Sugan	1412	12000	1000000	Mon Jun 27 10:04:18 IST 2016
Siva	1408	SBI	Felix	1234	5000	995000	Mon Jun 27 10:16:12 IST 2016
Siva	1408	SBI	Felix	1234	5000	990000	Mon Jun 27 11:38:51 IST 2016

Fig: 8.2.7.1

This page is used to view the transaction details of a particular client user.

8.2.8 DEPOSIT DETAILS PAGE (DEOSIT_DETAIL.jsp)



FROM BANK	FROM USER	FROM ACCOUNT	TO USER	TO ACCOUNT	AMOUNT DEPOSITED	BALANCE AMOUNT	DATE OF DEPOSITION
SBI	Felix	1234	Siva	1408	1000	1001000	Mon Jun 27 00:01:26 IST 2016
SBI	Felix	1234	Siva	1408	2000	1003000	Mon Jun 27 00:02:00 IST 2016
SBI	Felix	1234	Siva	1408	9000	1012000	Mon Jun 27 00:03:13 IST 2016

Fig: 8.2.8.1

This page is used to view the deposit details of a particular client user.

8.2.9 ALL TRANSACTION DETAILS PAGE

(TRANSACTION_DETAILS_ADMIN.jsp)

FROM USER	FROM ACCOUNT	TO BANK	TO USER	TO ACCOUNT	AMOUNT TRANSACTED	BALANCE AMOUNT	DATE OF TRANSACTION
Felix	1234	AXIS	Siva	1408	1000	999000	Mon Jun 27 00:01:26 IST 2016
Felix	1234	AXIS	Siva	1408	2000	997000	Mon Jun 27 00:02:00 IST 2016
Felix	1234	AXIS	Sugan	1412	7000	990000	Mon Jun 27 00:02:49 IST 2016
Felix	1234	AXIS	Siva	1408	9000	981000	Mon Jun 27 00:03:13 IST 2016

Fig: 8.2.9.1

This page is used to view all the transaction details of a particular bank by the Administrator.

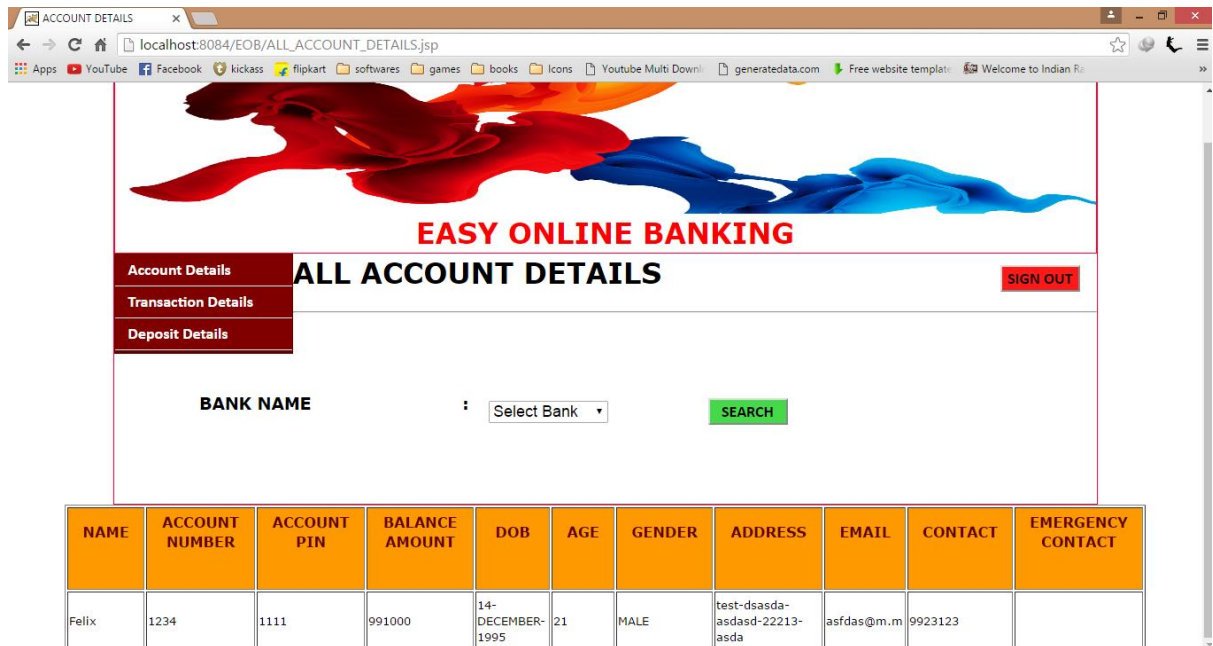
8.2.10 ALL DEPOSIT DETAILS PAGE (DEPOSIT_DETAILS_ADMIN.jsp)

FROM BANK	FROM USER	FROM ACCOUNT	TO USER	TO ACCOUNT	AMOUNT DEPOSITED	BALANCE AMOUNT	DATE OF DEPOSITION
SBI	Felix	1234	Siva	1408	1000	1001000	Mon Jun 27 00:01:26 IST 2016
SBI	Felix	1234	Siva	1408	2000	1003000	Mon Jun 27 00:02:00 IST 2016
SBI	Felix	1234	Sugan	1412	7000	1007000	Mon Jun 27 00:02:49 IST 2016
SBI	Felix	1234	Siva	1408	9000	1012000	Mon Jun 27 00:03:13 IST 2016
AXIS	Siva	1408	Sugan	1412	12000	1019000	Mon Jun 27 10:04:18 IST 2016

Fig: 8.2.10.1

This page is used to view all the deposit details of a particular bank by the Administrator.

8.2.11 ALL ACCOUNT DETAILS PAGE (ALL_ACCOUNT_DETAILS.jsp)



EASY ONLINE BANKING

Account Details
Transaction Details
Deposit Details

ALL ACCOUNT DETAILS **SIGN OUT**

BANK NAME : **SEARCH**

NAME	ACCOUNT NUMBER	ACCOUNT PIN	BALANCE AMOUNT	DOB	AGE	GENDER	ADDRESS	EMAIL	CONTACT	EMERGENCY CONTACT
Felix	1234	1111	991000	14-DECEMBER-1995	21	MALE	test-dsasda-asdasd-22213-asda	asfdas@m.m	9923123	

Fig: 8.2.11.1

This page is used to view all the Account details of a particular bank by the Administrator.