

# **VIRTUAL TRANSACTION SYSTEM**

A Project Report

submitted in partial fulfillment of the requirements

of

.....Edunet Foundation Techsakhasm Program Certificate.....

by

**Aswin M, 822720104009**

**Mathiyazhagan A, 822720104020**

**Prasannakumar S, 822720104027**

**Abineshwar K S, 822720104002**

Under the Esteemed Guidance of

**Mrs. R. Umamaheshwari**

## ACKNOWLEDGEMENT

---

We would like to take this opportunity to express our deep sense of gratitude to all individuals who helped us directly or indirectly during this thesis work.

Firstly, we would like to thank my supervisor, Mrs. R. Umamaheshwari, for being a great mentor and the best adviser I could ever have. His advice, encouragement and critics are source of innovative ideas, inspiration and causes behind the successful completion of this dissertation. The confidence shown on me by him was the biggest source of inspiration for me. It has been a privilege working with him from last one year. He always helped me during my thesis and many other aspects related to academics. His talks and lessons not only help in thesis work and other activities of college but also make me a good and responsible professional.

.....

*This Acknowledgement should be written by students in your own language (Do not copy and Paste)*

.....

.....

....

.....

## *ABSTRACT*

---

The "Virtual Transaction System" project introduces a comprehensive solution for modernizing and enhancing traditional banking operations. This web-based application employs HTML as the frontend, Java Servlets for backend logic, and MySQL for secure database management. The primary goal of the project is to provide users with a secure, efficient, and user-friendly platform for conducting various banking transactions.

The system encompasses key functionalities, including user authentication, balance inquiry, cash withdrawal, fund transfer, and transaction history. With a focus on security, the project integrates encryption measures, secure communication protocols, and logging mechanisms to safeguard sensitive user data and ensure transaction integrity.

The user interface is designed with responsiveness in mind, allowing seamless interaction on various devices. The project utilizes Eclipse IDE for Java EE Developers for streamlined development and deployment.

In conclusion, the Virtual Transaction System project serves as a foundational framework for modernizing banking operations, focusing on user experience, security, and adaptability to future technological advancements. The implementation and future developments discussed herein lay the groundwork for a robust and scalable solution that aligns with the ever-changing needs of the financial industry.

## TABLE OF CONTENTS

Abstract .....	3
List of Figures .....	5
List of Tables .....	5
 <b>Chapter 1. Introduction</b> .....	 <b>6</b>
1.1 Problem Statement .....	7
1.2 Problem Definition .....	7
1.3 Expected Outcomes.....	8
1.4 Organization of the Report.....	8
 <b>Chapter 2. Literature Survey</b> .....	 <b>10</b>
2.1 Paper-1 .....	10
2.1.1 Brief Introduction of Paper .....	10
2.1.2 Techniques used in Paper .....	11
 <b>Chapter 3. Proposed Methodology</b> .....	 <b>13</b>
3.1 System Design .....	13
3.2 Modules Used ...	17
3.3 Data Flow Diagram .....	18
3.4 Advantages... ..	20
3.5 Requirements Specification... ..	21
 <b>Chapter 4. Implementation and Results</b> .....	 <b>25</b>
4.1 Home Page .....	26
4.2 Registration Page .....	26
4.3 Login Page .....	27
4.4 Transaction Page .....	27
4.5 Database Page .....	28
4.6 Overall View .....	28

<b>Chapter 5. Conclusion .....</b>	<b>31</b>
<b>GitHub Link .....</b>	<b>33</b>
<b>Video Link .....</b>	<b>33</b>
<b>References .....</b>	<b>34</b>

## CHAPTER 1

### INTRODUCTION

#### 1.1. Problem Statement:

Traditional banking systems often face inefficiencies and security concerns in their manual processes, hindering the seamless and secure execution of financial transactions. The absence of an automated and user-friendly solution results in challenges for users to access and manage their accounts efficiently. Consequently, there is a pressing need for a modernized and secure "Virtual Transaction System" that leverages web technologies to provide users with a reliable platform for conducting various banking transactions.

Existing systems lack the responsiveness required for diverse user interfaces, and the absence of robust security measures poses risks to sensitive user information. Additionally, the conventional infrastructure limits the incorporation of emerging technologies, hindering adaptability to evolving user expectations and industry standards.

#### 1.2. Problem Definition:

The existing banking systems face significant challenges in providing a seamless, secure, and user-friendly experience for account holders. Manual processes and outdated infrastructure hinder the efficient execution of financial transactions, leading to inconveniences for users. The absence of an automated solution results in a lack of responsiveness in handling diverse user interfaces and introduces security vulnerabilities.

### **1.3. Expected Outcomes:**

The implementation of the "Virtual Transaction System" project anticipates the delivery of a modernized banking platform with the following key outcomes:

- User-Friendly Interface
- Secure Transactions
- Efficient Backend Logic
- Reliable Database Management
- Transaction Features
- Adaptability and Scalability
- Future-Ready Framework

### **1.4. Organization of the Report**

The remaining report is organized as follows:

Chapter 2 - Literature Survey

Chapter 3 - Proposed Methodology

Chapter 4 - Implementation and Result

Chapter 5 - Conclusion

Chapter 6 - Appendix

## **CHAPTER 2**

### **LITERATURE SURVEY**



## CHAPTER 2

### LITERATURE SURVEY

#### 2.1. Paper-1

- Design and Development of a Secure Banking System using Advanced Java by Smith, J. Johnson, .A, Brown, .M
- "Enhancing User Experience in Banking Applications with Java Spring Boot and Angular" Authors: Williams, S., Davis, R., Thompson

##### 2.1.1. Brief Introduction of Paper

The literature survey for Paper 1 delves into an exploration of existing research, scholarly articles, and publications relevant to the selected topic. The objective is to establish a comprehensive understanding of the current state of knowledge, identify gaps or areas for further investigation, and contextualize the forthcoming research within the broader academic landscape.

This paper presents the design and implementation of a secure banking system utilizing Java Servlet and Java Database Connectivity. It discusses the architectural decisions, security considerations, and integration of key banking features. The paper highlights the benefits of the chosen technology stack for building a modern and robust banking system.

This paper explores how Advance Java with JSP can be leveraged to improve user experience in banking applications. It discusses techniques for building responsive and intuitive user interfaces, implementing real-time updates, and ensuring smooth navigation. The paper also presents case studies showcasing the positive impact of these frameworks on user satisfaction.

Establish the relevance of the reviewed literature to the specific focus of Paper 1. Discuss how insights gained from the literature survey will contribute to the research objectives and framework of the current study. In summary, the literature survey for Paper 1 synthesizes existing knowledge, highlights gaps, and sets the stage for the research to contribute meaningfully to the academic discourse in the chosen field.

### **2.1.2. Techniques used in Paper**

- Overview of Current Research:
- Notable Frameworks and Models:
- Comparative Analysis:
- Identified Gaps and Limitations:
- Methodological Approaches:
- Key Authors and Influential Works:
- Evolving Trends and Future Directions:
- Critique of Existing Literature:
- Relevance to Current Study

## **CHAPTER 3**

# **PROPOSED METHODOLOGY**

## CHAPTER 3

### PROPOSED METHODOLOGY

#### 3.1 System Design

The system design phase is a critical aspect of the proposed methodology, encompassing the architectural blueprint and technical specifications for the development of the Virtual Transaction System. This section outlines the key components, design principles, and methodologies that will guide the construction of the system.

##### 1. System Architecture:

Define the overall structure of the Virtual Transaction System. Discuss whether a monolithic or microservices architecture will be adopted, highlighting the rationale behind the chosen approach.

##### 2. Frontend Design:

Detail the design considerations for the HTML-based frontend. Discuss the user interface (UI) design principles, navigation flows, and responsiveness aspects to ensure a seamless user experience across devices.

##### 3. Backend Design:

Outline the design of the Java Servlets responsible for handling backend logic. Describe the modularization of functionalities, data processing workflows, and integration with external services.

##### 4. Database Schema:

Present the proposed MySQL database schema. Define tables, relationships, and constraints to efficiently store user account details, transaction records, and system logs.

5. Security Measures: Detail the security features integrated into the system design. This includes encryption algorithms for sensitive data, secure communication protocols, and mechanisms for preventing common security threats.

### 6. Integration of Technologies:

Discuss how technologies such as Java Servlets, HTML, and MySQL will be integrated to ensure seamless communication between the frontend and backend components.

### 7. User Authentication Mechanism:

Specify the design of the user authentication system. Detail the steps involved in verifying user credentials, enforcing password policies, and ensuring secure login sessions.

### 8. Transaction Processing:

Define the workflow for transaction processing, covering operations like balance inquiry, cash withdrawal, fund transfer, and transaction history. Discuss error-handling mechanisms to ensure system robustness.

### 9. Responsive Design and Cross-Browser Compatibility:

Describe strategies for responsive design, ensuring that the HTML pages adapt to different screen sizes and orientations. Discuss measures taken to ensure cross-browser compatibility.

### 10. Logging and Auditing:

Outline the logging mechanisms implemented for recording important system events. Discuss how audit trails will be maintained to track user actions for security and accountability purposes.

### 11. Scalability and Future Expansion:

Address the system's scalability, highlighting design considerations that allow for future expansion and the incorporation of additional features or technologies.

### 12. Testing Methodology:

Provide an overview of the testing methodology employed during the system design phase. Discuss unit testing, integration testing, and any other testing strategies implemented to ensure the reliability of the proposed system.

### **3.1.1 Registration:**

The registration process is a crucial component of the system design, defining how users establish their identity within the Virtual Transaction System. This section outlines the steps and considerations involved in creating a seamless and secure registration experience.

#### **User Input and Validation:**

- Users initiate the registration by providing essential information, including personal details and contact information.
- Implement client-side validation using JavaScript to ensure that users enter accurate and valid information before submission.

#### **Server-Side Validation:**

- Employ server-side validation in Java Servlets to revalidate user input, checking for data consistency and preventing common security vulnerabilities.
- Ensure that usernames, email addresses, or other unique identifiers do not already exist in the system.

#### **Secure Password Handling:**

- Apply industry-standard hashing algorithms (e.g., encrypt) to securely store and manage user passwords in the database.
- Enforce password complexity requirements to enhance security.

#### **User Profile Creation:**

- Upon successful registration, automatically create a user profile with default settings and preferences.
- Allow users to customize their profiles after registration for a personalized experience.

#### **Account Confirmation and Welcome:**

- Provide users with a confirmation message upon successful registration.
- Welcome users to the system and guide them on the next steps, such as logging in for the first time.

### **Error Handling:**

- Implement comprehensive error handling mechanisms to gracefully manage situations such as network failures, server errors, or incomplete registrations.
- Display meaningful error messages to guide users in correcting their input.

The proposed methodology for the registration process focuses on creating a secure, user-friendly, and inclusive experience, adhering to industry standards for account creation within the Virtual Transaction System.

### **3.1.2 Recognition:**

Recognition within the system design pertains to the mechanisms by which the Virtual Transaction System identifies and authenticates users during subsequent interactions. The following outlines the key aspects and considerations in implementing a robust user recognition system.

#### **User Authentication:**

- Utilize secure authentication protocols, such as OAuth or Token-based authentication, to validate user identities during login.
- Employ industry-standard encryption for secure transmission of authentication credentials between the frontend and backend.

#### **Session Management:**

- Implement session management to track and maintain user sessions securely.
- Define session timeout policies to enhance security by automatically logging out inactive users.

#### **User Profile Modification:**

- Allow users to modify their profile information securely, incorporating validation checks and secure data transmission.

**Access Control Levels:**

- Implement access control levels to differentiate user roles and permissions within the system.
- Define roles such as standard user, administrator, or other applicable roles based on the system requirements.

**Secure Communication Protocols:**

- Enforce the use of HTTPS to ensure encrypted communication between the user's browser and the server, preventing unauthorized interception of sensitive information.

**Continuous Monitoring:**

- Implement continuous monitoring mechanisms to detect and respond to suspicious activities or unauthorized access promptly.

The proposed methodology for user recognition focuses on establishing secure and efficient mechanisms for identifying and authenticating users within the Virtual Transaction System. The design emphasizes not only current authentication needs but also prepares for future enhancements to bolster security and user recognition capabilities.

### **3.2 Modules Used**

- Account Creation and Validation
- Deposit
- Transfer Funds
- Checking Balance
- Front End: HTML, CSS, JavaScript
- Front End Framework: Bootstrap
- Back End: Java Servlets
- Database: MySQL
- Development Environment: Eclipse IDE for Java Developers



### **3.2.1 User Registration and Authentication:**

Handles user account creation, validation, and authentication processes

### **3.2.2 User Login and Validation:**

Handles user login validation after the registration.

### **3.2.3 Transaction Page:**

Handles Transactions like Deposit the Money, Transfer Funds to other users, Checking Balance Enquiry.

### **3.2.4 Deposit:**

Deposits the amount using the Input Field and added to the user's bank account.

### **3.2.5 Check Balance:**

Checking the current balance of the user before and after the transactions.

### **3.2.6 Transfer Money:**

Used to transfer money/funds from one user to destination user with the user id of the destination user.

### **3.2.7 Backend Processing:**

Utilizes Java Servlets to handle backend logic, including user actions, transaction processing, and communication with the database.

## **3.3 Data Flow Diagram**

A Data Flow Diagram (DFD) is a graphical representation of the "flow" of data through an information system, modeling its process aspects. A DFD is often used as a preliminary step to create an overview of the system, which can later be elaborated. DFDs can also be used for the visualization of data processing (structured design).

### 3.3.1. DFD Level 0



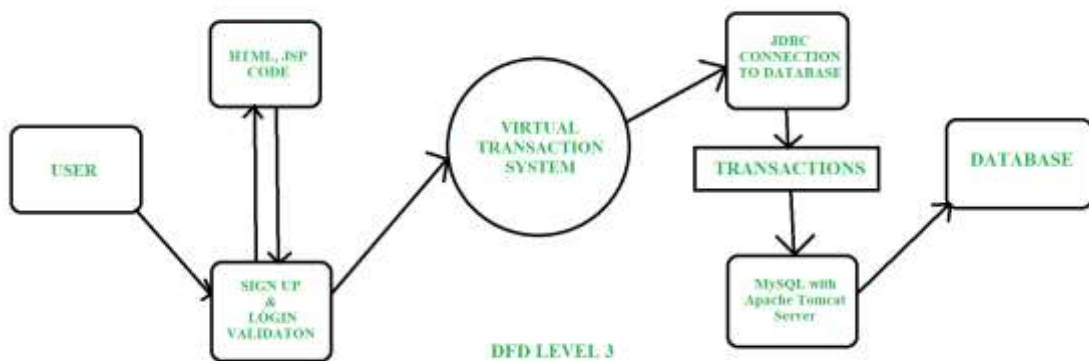
LEVEL 0 - DFD

### 3.3.2. DFD Level 1



DFD LEVEL 1

### 3.3.3. DFD Level 2



DFD LEVEL 3

### **3.4 Advantages:**

The proposed methodology for the Virtual Transaction System offers several advantages, contributing to the success and effectiveness of the project:

#### **Security Enhancement:**

Incorporates robust security measures, including encryption, secure communication protocols, and secure password handling, ensuring the confidentiality and integrity of user data.

#### **Scalability and Adaptability:**

Designs the system with scalability in mind, allowing for future expansion and the integration of emerging technologies, ensuring the system remains adaptable to evolving user expectations.

#### **User-Friendly Interface:**

Utilizes HTML-based frontend and responsive design principles, providing users with a seamless and intuitive interface across various devices.

#### **Efficient Transaction Processing:**

Implements Java Servlets for backend logic, enabling swift and accurate transaction processing for operations such as balance inquiries, cash withdrawals, and fund transfers.

#### **Comprehensive Logging and Auditing:**

Incorporates logging mechanisms and audit trails, providing a comprehensive record of system events for security and accountability purposes.

#### **Cross-Browser Compatibility:**

Ensures that the system's frontend is compatible with various web browsers, offering users a consistent and reliable experience.

### **3.5 Requirement Specification**

#### **3.5.1. Hardware Requirements:**

**Server:**

Processor: Multi-core processor (e.g., Intel Core i5 or equivalent)

RAM: 8 GB or higher

Storage: SSD storage for improved performance

**Database Server:**

MySQL database server

Adequate storage based on the expected volume of data

**Network Components:**

High-speed internet connection

Network switches and routers for local network connectivity

**Client Devices:**

Desktops, laptops, tablets, or smartphones

Compatible web browsers (Chrome, Firefox, Safari, Edge)

#### **3.5.2. Software Requirements:**

**Operating System:**

Server: Linux (e.g., Ubuntu Server) or Windows Server

Client Devices: Windows, macOS, Linux, or mobile OS (iOS, Android)

**Web Server:**

Apache Tomcat or any Java EE-compatible server for deploying Java Servlets.

**Database Management System:**

MySQL Database Server (version 5.7 or higher)

**Development Environment:**

Eclipse IDE for Java EE Developers (or any other Java development environment)

**Programming Languages:**

- Java for backend development
- HTML, CSS, JavaScript for frontend development

**Frameworks and Libraries:**

- Java Servlets for backend logic
- Bootstrap or any responsive design framework for frontend development
- jQuery or other JavaScript libraries for enhanced interactivity

**Version Control:**

Git for version control and collaboration

**Testing Frameworks:**

- JUnit for unit testing Java code
- Selenium or similar for automated testing of the web interface

**Documentation Tools:**

Markdown or any documentation tool for creating project documentation

**Deployment Tools:**

Tools like GitHub for continuous integration and deployment.

**Collaboration Tools:**

Communication tools (e.g., GitHub, Google Meet, Microsoft Teams) for team collaboration.

**Backup and Recovery Tools:**

- Regular backup mechanisms for database and server data
- Recovery tools in case of system failures



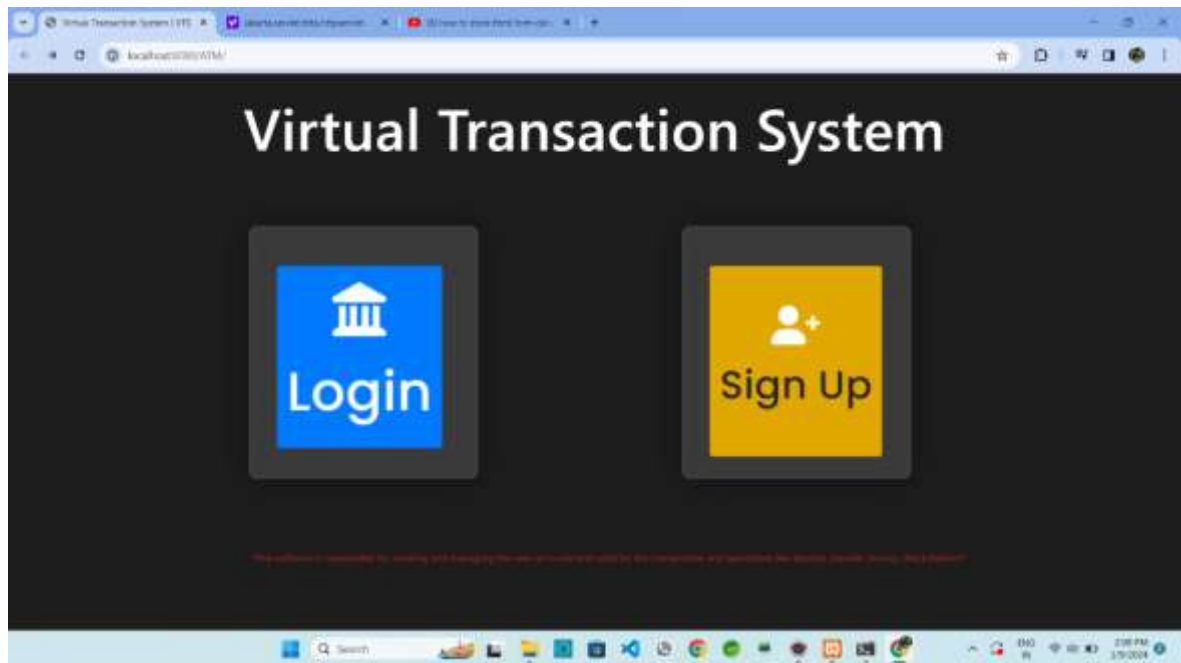
## **CHAPTER 4**

### **Implementation and Result**

## CHAPTER 4

### IMPLEMENTATION and RESULT

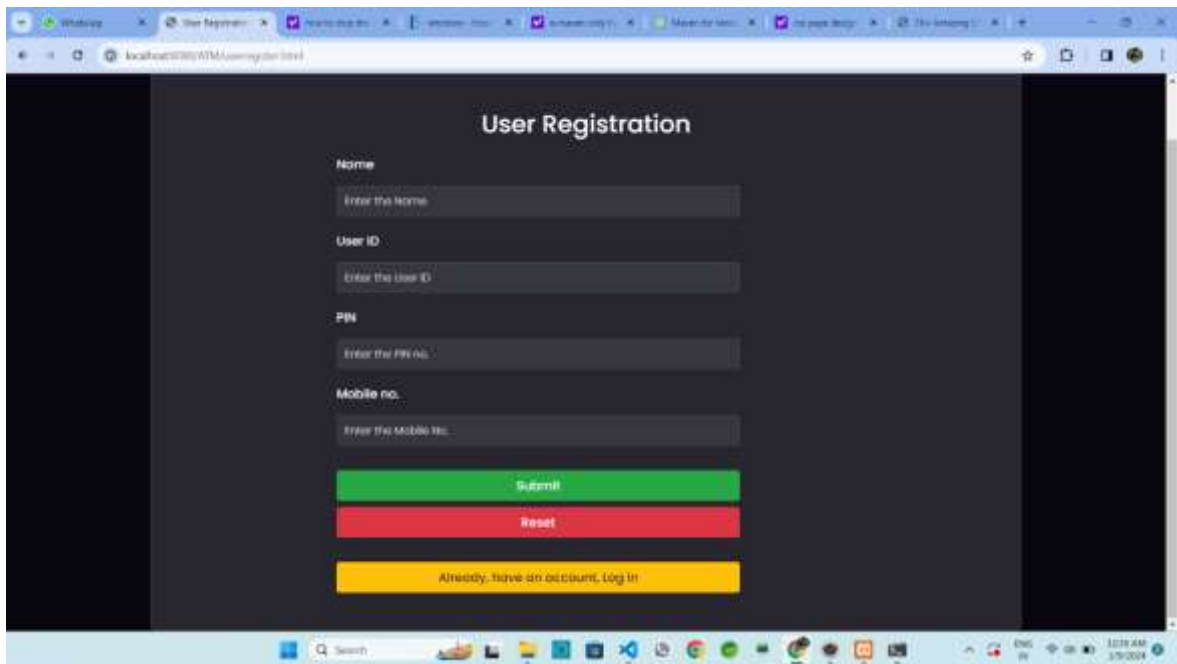
#### 4.1. Home Page



This is the initial page of our project which shows the Project Title “Virtual Transaction System” and Login, Signup Portals, also shows the instruction line mentioned below.

#### 4.2. Registration (Sign up) Page



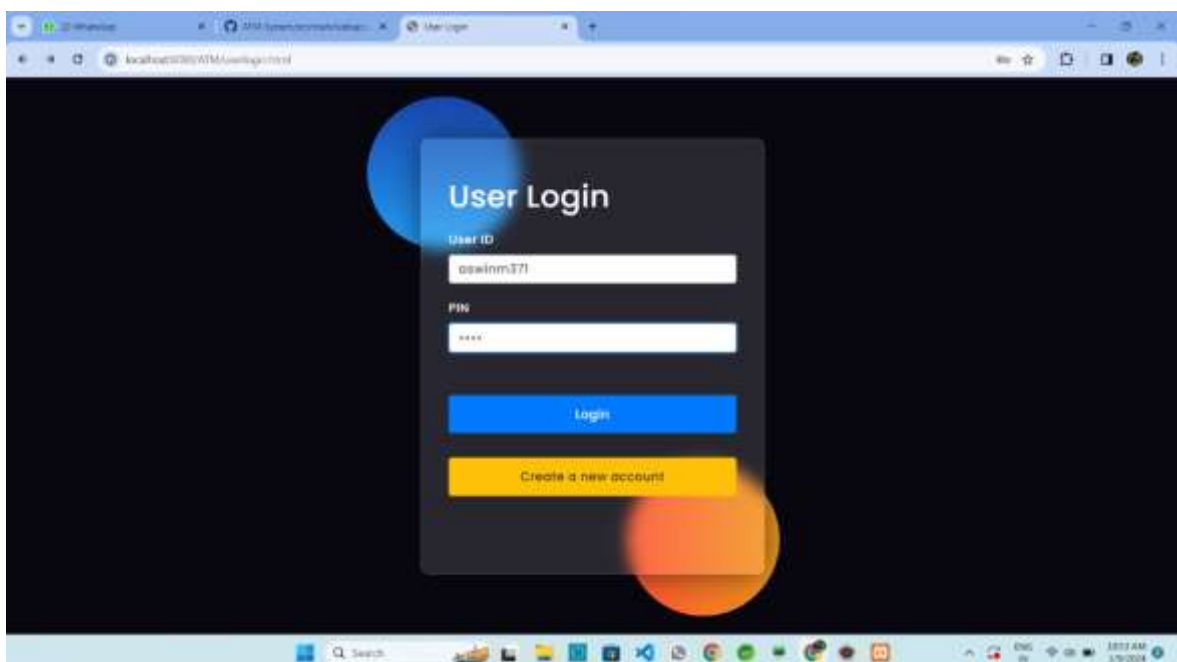


The screenshot shows a web browser window with the URL `localhost:5778/ATMUserRegister.html`. The page has a dark background and is titled "User Registration". It contains the following fields and buttons:

- Name:** A text input field with the placeholder "Enter the Name".
- User ID:** A text input field with the placeholder "Enter the User ID".
- PIN:** A text input field with the placeholder "Enter the PIN no.". Below this field is a small "show" link.
- Mobile no.:** A text input field with the placeholder "Enter the Mobile No.". Below this field is a small "show" link.
- Buttons:** A green "Submit" button, a red "Reset" button, and a yellow button at the bottom that says "Already, have an account, Log in".

This page used to create accounts with the initial balance amount for the every user.

### 4.3. Login Page

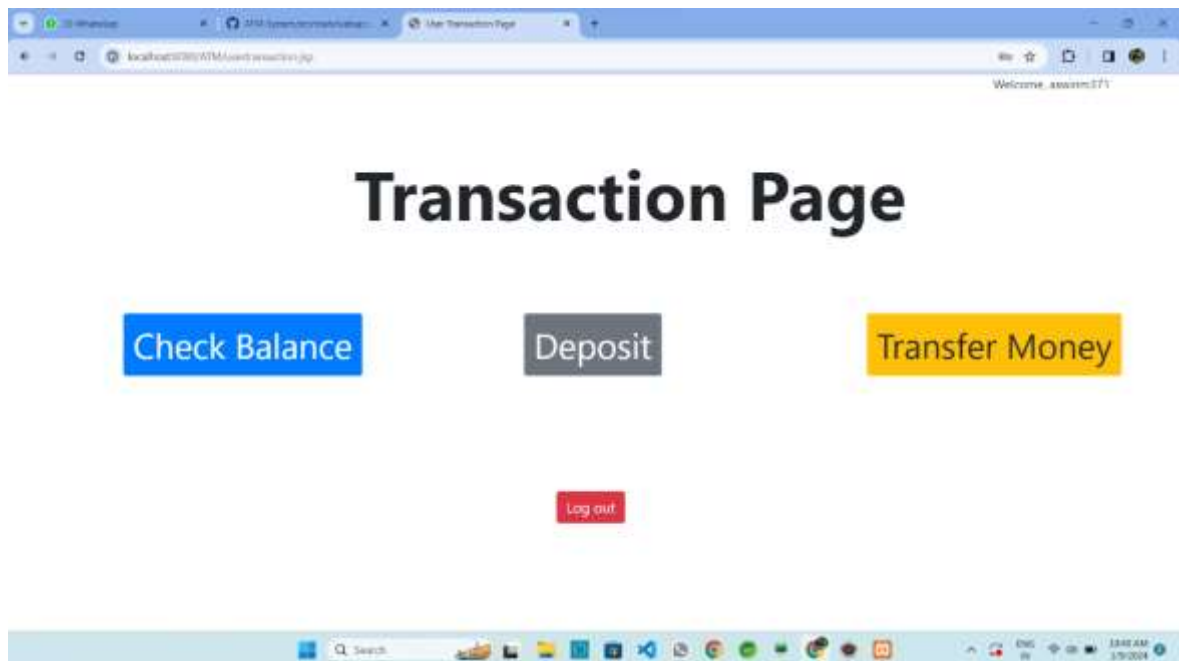


The screenshot shows a web browser window with the URL `localhost:5778/ATMUserLogin.html`. The page has a dark background and is titled "User Login". It contains the following fields and buttons:

- User ID:** A text input field containing the value "pswinm371".
- PIN:** A text input field containing the value "1234".
- Buttons:** A blue "Login" button and a yellow button at the bottom that says "Create a new account".

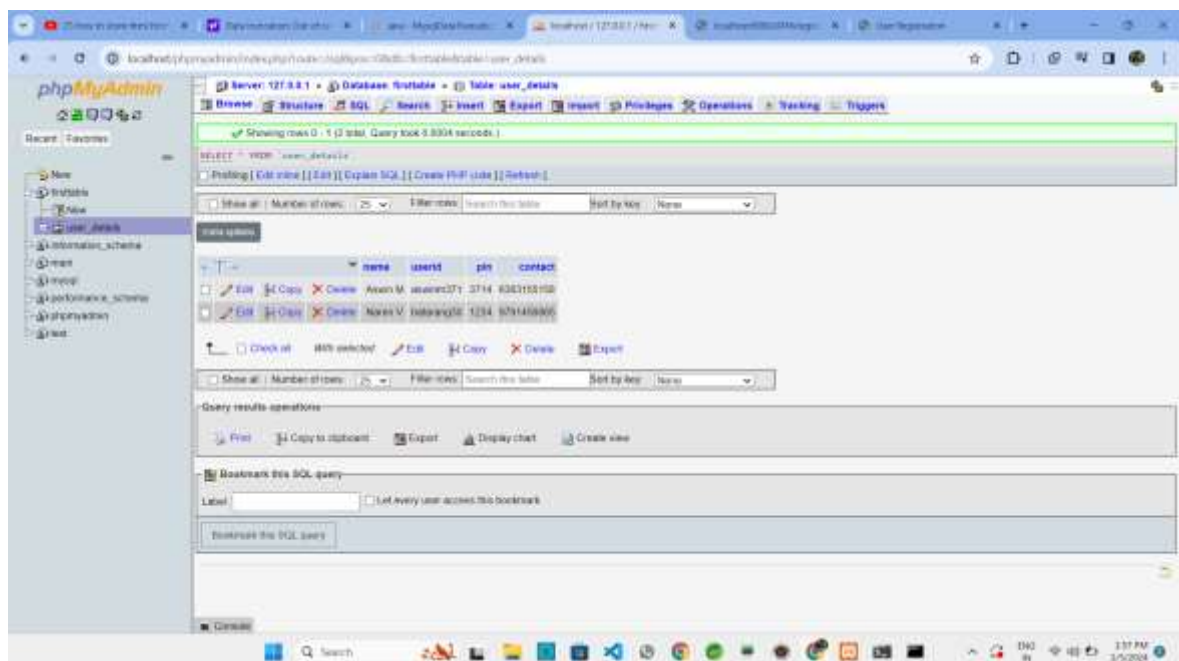
This the login page, it will validate and verify the details given from the registration page and then to move on to the Transaction page.

## 4.4. Transaction Page



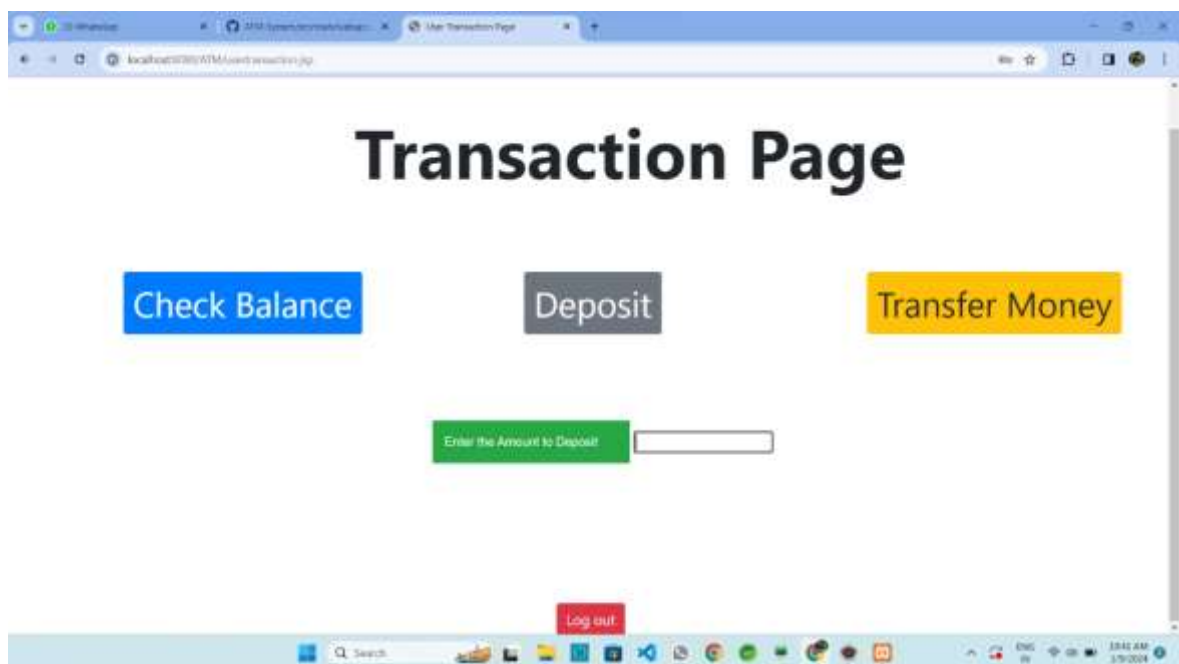
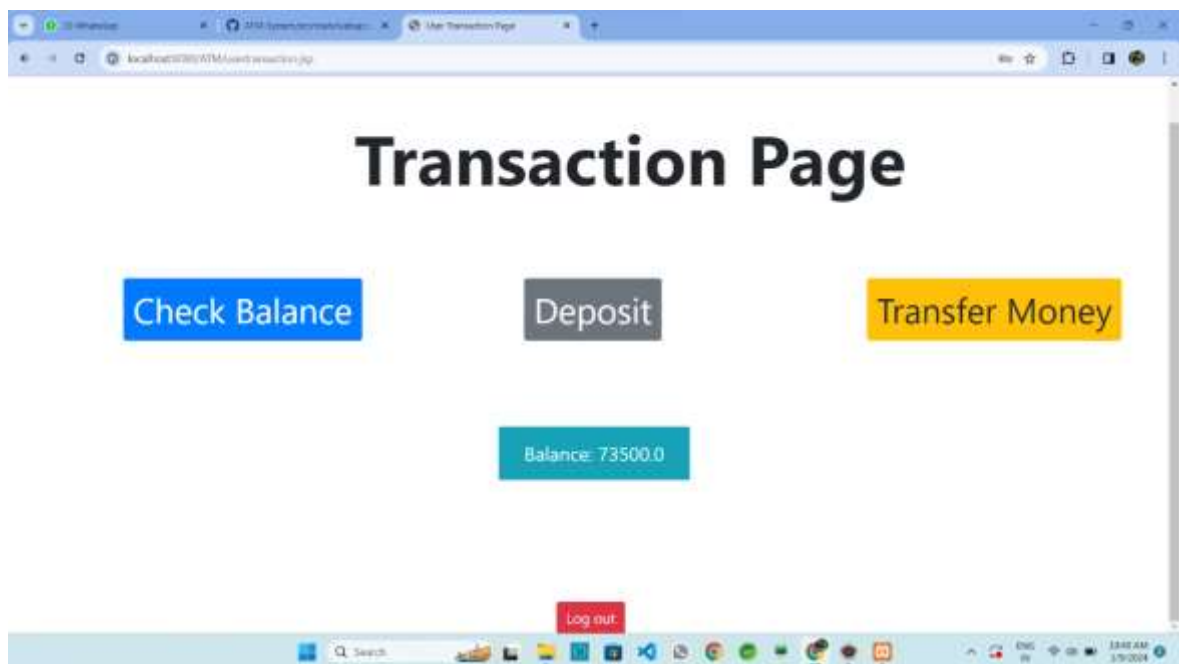
This the Transaction Page which controls overall process of the Project.

## 4.5. Database Page



This is the page which controls the database and its records of users who are using this Transaction System.

## 4.6. Overall View





## **CHAPTER 5**

## **CONCLUSION**

## CHAPTER 5

### CONCLUSION

#### ADVANTAGES:

1. Enhanced User Experience:

The HTML-based frontend ensures a responsive and intuitive interface, providing users with a seamless and user-friendly banking experience.

2. Security Measures:

Robust security features, including encryption and secure communication protocols, safeguard sensitive user data, ensuring the confidentiality and integrity of transactions.

3. Efficient Transaction Processing:

Java Servlets handle backend logic efficiently, enabling swift and accurate transaction processing for operations such as balance inquiries, cash withdrawals, and fund transfers.

4. Scalability and Adaptability:

The system is designed with scalability in mind, allowing for future expansion and the integration of emerging technologies to meet evolving user expectations.

## **SCOPE:**

The scope of the "Virtual Transaction System" project encompasses the design, development, and implementation of a contemporary and secure online banking platform. Key elements within the project scope include:

1. Frontend Development:
  - Design and implement a responsive HTML-based frontend for a user-friendly interface across various devices.
2. Backend Development:
  - Develop Java Servlets to handle backend logic, ensuring efficient transaction processing, user authentication, and data management.
3. Database Management:
  - Implement and manage a MySQL database for storing user account details, transaction records, and system logs securely.
4. Security Measures:
  - Incorporate robust security features, including encryption and secure communication protocols, to protect sensitive user information.
5. Transaction Operations:
  - Enable standard banking operations such as balance inquiry, cash withdrawal, fund transfer, and transaction history.
6. Scalability and Adaptability:
  - Design the system with scalability in mind, allowing for future expansion and integration with emerging technologies.
7. Logging and Auditing:
  - Implement logging mechanisms and audit trails to track user actions for security and accountability purposes.
8. Cross-Browser Compatibility:
  - Ensure that the system's frontend is compatible with various web browsers, providing a consistent user experience.

9. Future Enhancements:

- Lay the groundwork for potential future enhancements, including multi-factor authentication, mobile banking integration, and advanced analytics.

10. Testing and Quality Assurance:

- Implement a thorough testing methodology, including unit testing and integration testing, to ensure the reliability and quality of the system.

11. Documentation:

- Provide comprehensive documentation detailing the system architecture, design choices, and implementation details for future reference and maintenance.

12. Deployment:

- Deploy the Virtual Transaction System on a web server, making it accessible to users for real-world testing and usage.

The overarching goal of the project is to create a secure, efficient, and adaptable banking platform that enhances user experiences and addresses the challenges faced by traditional banking systems.

## GITHUB LINK

**Project Link:** <https://github.com/aswinm371/ATM-System>

**Deployment Link:** <https://aswinm371.github.io/ATM-System/src/main/webapp/index.html>

## VIDEO LINK:

<https://drive.google.com/file/d/14RZNVIAx2EEj9zRwpidEBNgZR4lb3uqQ/view>



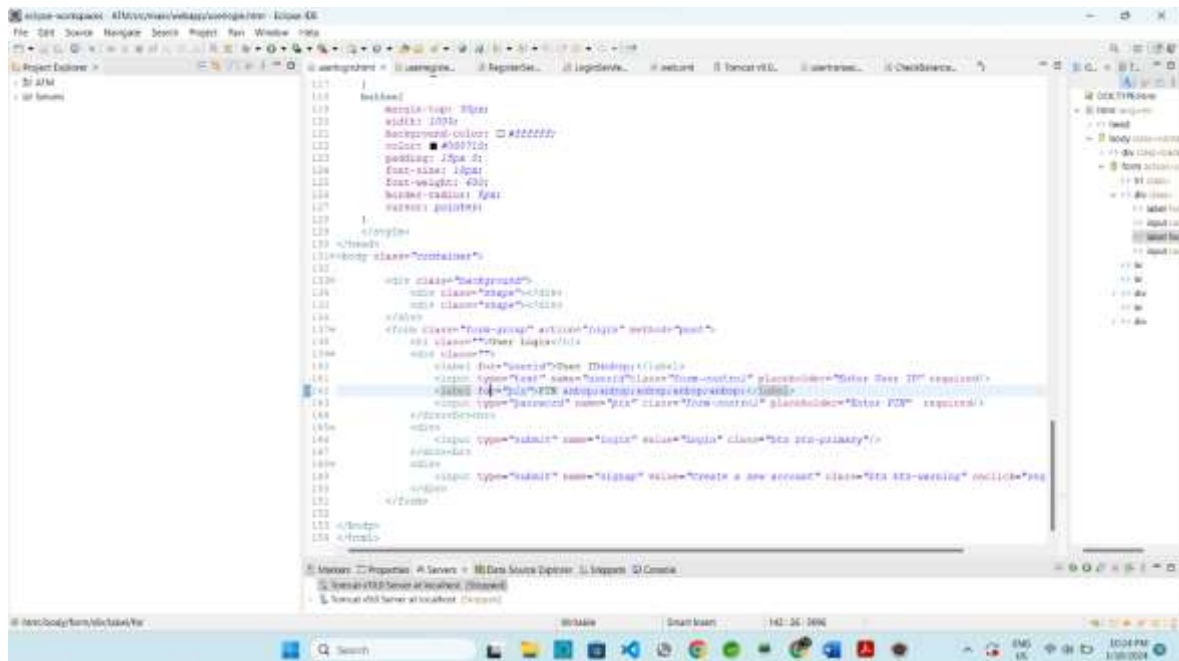
## REFERENCES

- [1]. Prof. Bharati Pursharthi a, Kanchan Ramesh Deshmukh, "Building a Modern Banking System: Implementation of a Advanced Java-Based Solution", ISSN 2582-7421.
- [2]. Search engines by W. Bruce Croft, Donald Metzler, Trevor Strohman
- [3]. Web search engine and International journal and magazine of engineering, technology, management and research.
- [4]. <https://www.w3schools.com/>
- [5]. <https://stackoverflow.com/>
- [6]. <https://www.mysql.com/>

## APPENDIX

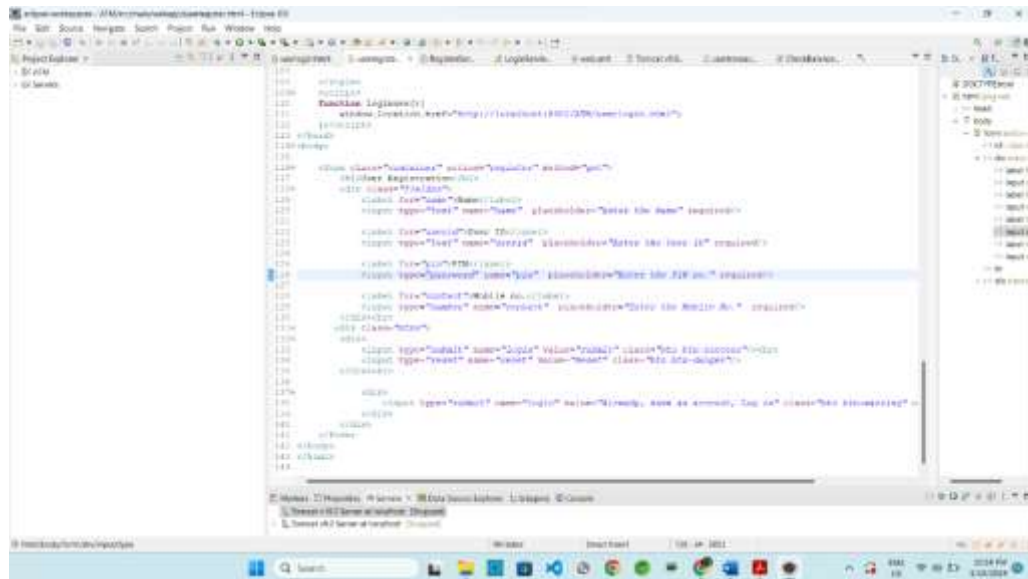
## SAMPLE SOURCE CODE

## Login.html



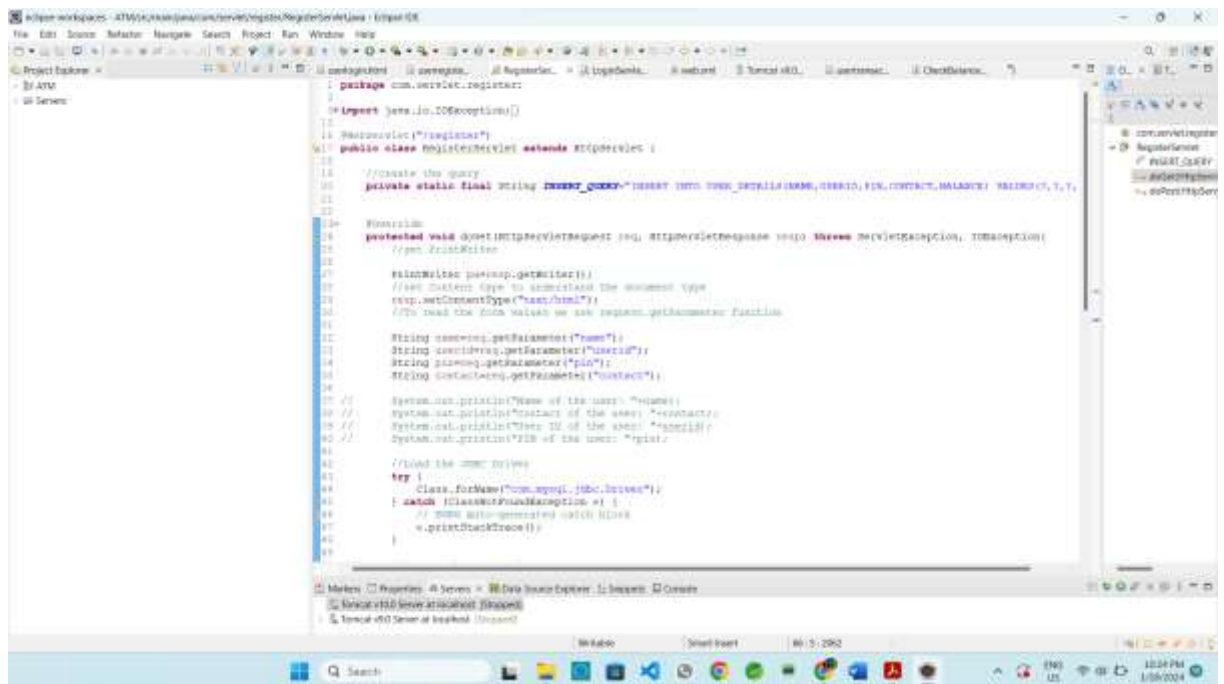
# VIRTUAL TRANSACTION SYSTEM

## Userregister.html



```
101 <html>
102 <head>
103 <title>
104     Userregister
105 </title>
106 </head>
107 <body>
108     <div class="container">
109         <div class="row">
110             <div class="col-md-6">
111                 <div class="card">
112                     <div class="card-header">
113                         Userregister
114                     </div>
115                     <div class="card-body">
116                         <div class="form-group">
117                             <input type="text" class="form-control" value="Name" />
118                         </div>
119                         <div class="form-group">
120                             <input type="text" class="form-control" value="Email" />
121                         </div>
122                         <div class="form-group">
123                             <input type="password" class="form-control" value="Password" />
124                         </div>
125                         <div class="form-group">
126                             <input type="text" class="form-control" value="Phone" />
127                         </div>
128                         <div class="form-group">
129                             <input type="button" value="Submit" />
130                         </div>
131                     </div>
132                 </div>
133             </div>
134         </div>
135     </div>
136 </body>
137 </html>
```

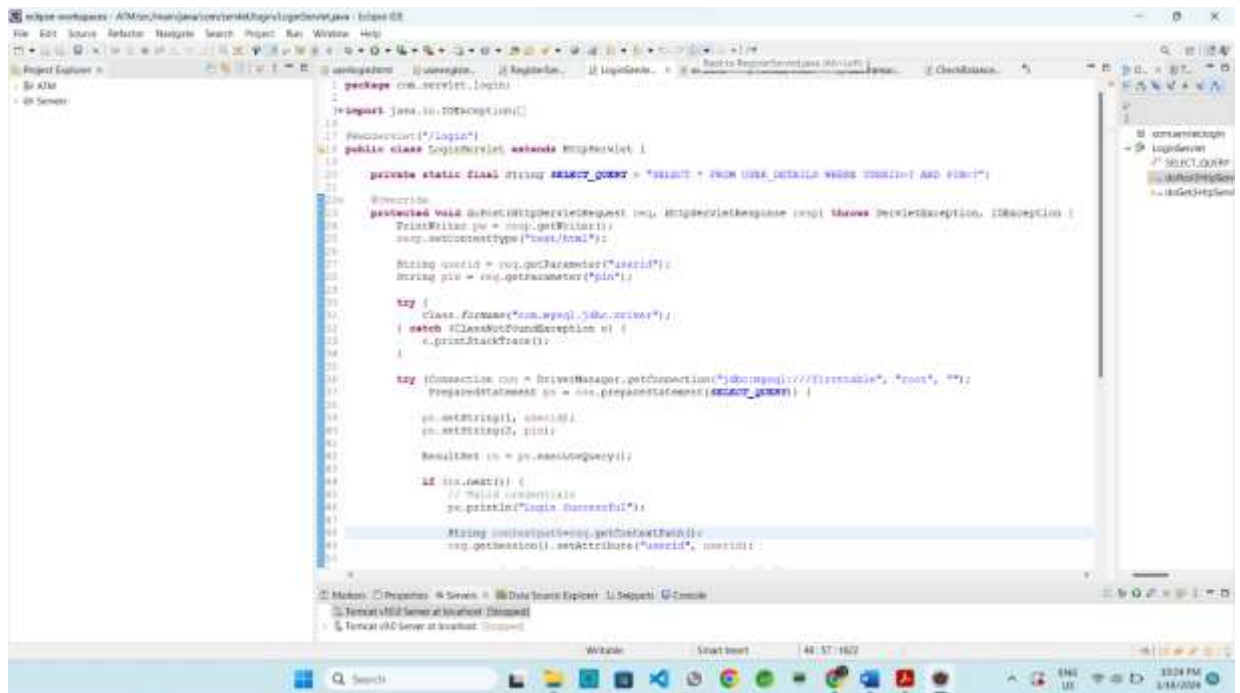
## RegisterServlet.java



```
1 package com.servlet.register;
2
3 import java.io.IOException;
4
5 import javax.servlet.ServletException;
6 import javax.servlet.http.HttpServlet;
7 import javax.servlet.http.HttpServletRequest;
8 import javax.servlet.http.HttpServletResponse;
9
10 public class RegisterServlet extends HttpServlet {
11
12     //create the query
13     private static final String QUERY = "INSERT INTO USER (USERNAME, EMAIL, PIN, CONTACT, BALANCE) VALUES (?, ?, ?, ?, ?)";
14
15     //override the doGet method
16     protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {
17         //get the request
18         PrintWriter writer = resp.getWriter();
19         //set the content type to text/html
20         resp.setContentType("text/html");
21         //To read the form values we use request.getParameter() function
22
23         String username = req.getParameter("name");
24         String email = req.getParameter("email");
25         String password = req.getParameter("pin");
26         String contact = req.getParameter("contact");
27
28         //Print the details of the user
29         System.out.println("Name of the user: " + username);
30         System.out.println("Email of the user: " + email);
31         System.out.println("PIN of the user: " + password);
32         System.out.println("Contact of the user: " + contact);
33
34         //Load the JDBC driver
35         try {
36             Class.forName("com.mysql.jdbc.Driver");
37         } catch (ClassNotFoundException e) {
38             //Print the stack trace
39             e.printStackTrace();
40         }
41     }
42 }
```

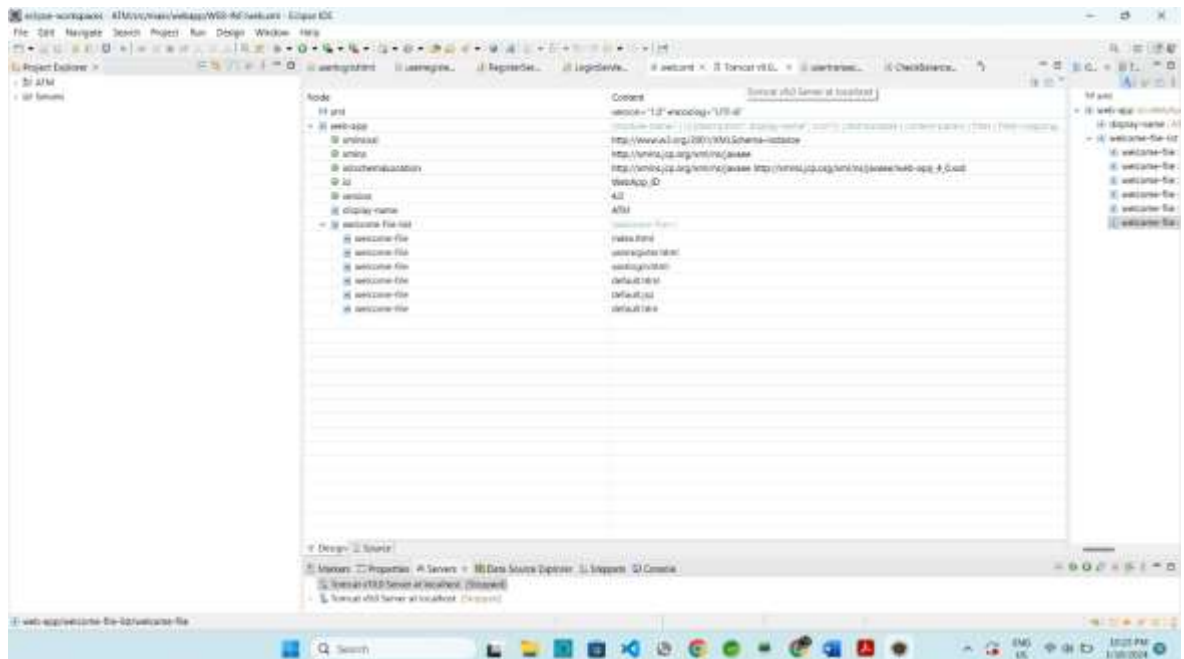
## LoginServlet.java

# VIRTUAL TRANSACTION SYSTEM



```
1 package com.servlet.login;  
2  
3 import java.io.IOException;  
4  
5 import javax.servlet.*;  
6  
7 public class LoginServlet extends HttpServlet {  
8  
9     private static final String SELECT_QUERY = "SELECT * FROM USER_DETAILS WHERE USERID=? AND PIN=?";  
10  
11     @Override  
12     protected void doGet(HttpServletRequest req, HttpServletResponse resp) throws ServletException, IOException {  
13         PrintWriter pw = resp.getWriter();  
14         resp.setContentType("text/html");  
15  
16         String userid = req.getParameter("userid");  
17         String pin = req.getParameter("pin");  
18  
19         try {  
20             Class.forName("com.mysql.jdbc.Driver");  
21             Connection con = DriverManager.getConnection("jdbc:mysql://localhost:3306/vts", "root", "");  
22             PreparedStatement ps = con.prepareStatement(SELECT_QUERY);  
23  
24             ps.setString(1, userid);  
25             ps.setString(2, pin);  
26  
27             ResultSet rs = ps.executeQuery();  
28  
29             if (rs.next()) {  
30                 // Valid credentials  
31                 pw.println("Login Successful");  
32  
33                 String sessionId = req.getSession().getId();  
34                 resp.getSession().setAttribute("userid", userid);  
35  
36             }  
37  
38         } catch (Exception e) {  
39             pw.println("Login Failed");  
40         }  
41     }  
42 }  
43  
44 }
```

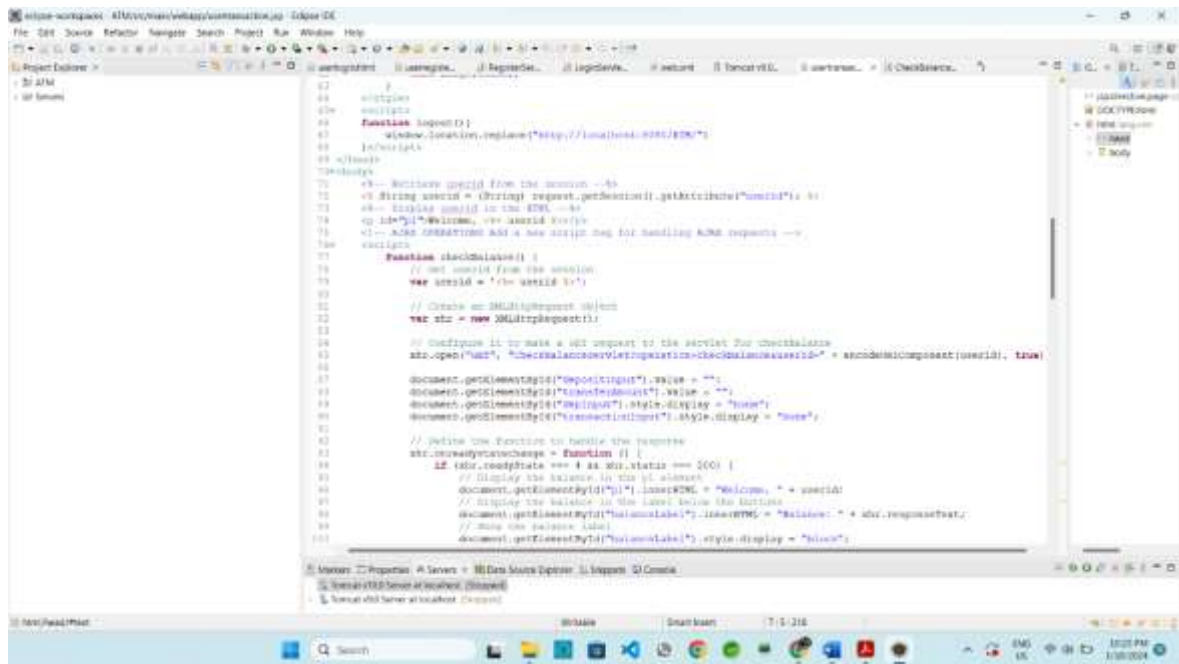
## Web.xml



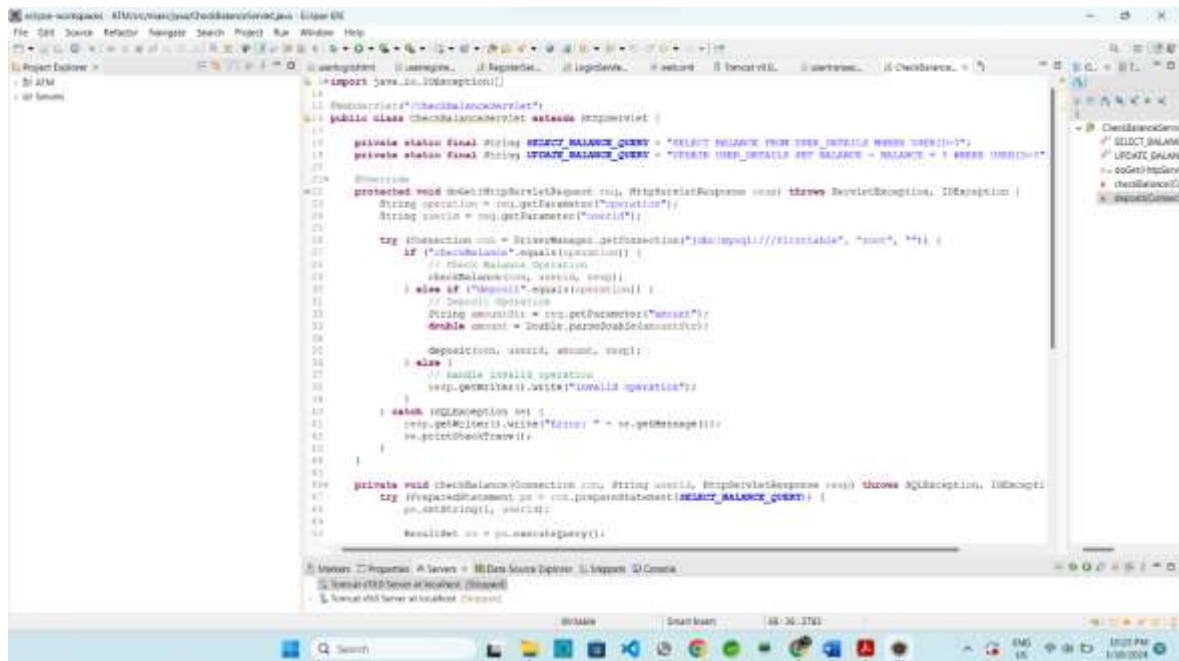
```
1 <?xml version="1.0" encoding="UTF-8"?>  
2 <web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd" >  
3     <servlet>  
4         <servlet-name>LoginServlet</servlet-name>  
5         <servlet-class>com.servlet.login.LoginServlet</servlet-class>  
6     </servlet>  
7     <servlet-mapping>  
8         <servlet-name>LoginServlet</servlet-name>  
9         <servlet-url-pattern>/login</servlet-url-pattern>  
10    </servlet-mapping>  
11 </web-app>
```

## Usertransaction.jsp

# VIRTUAL TRANSACTION SYSTEM

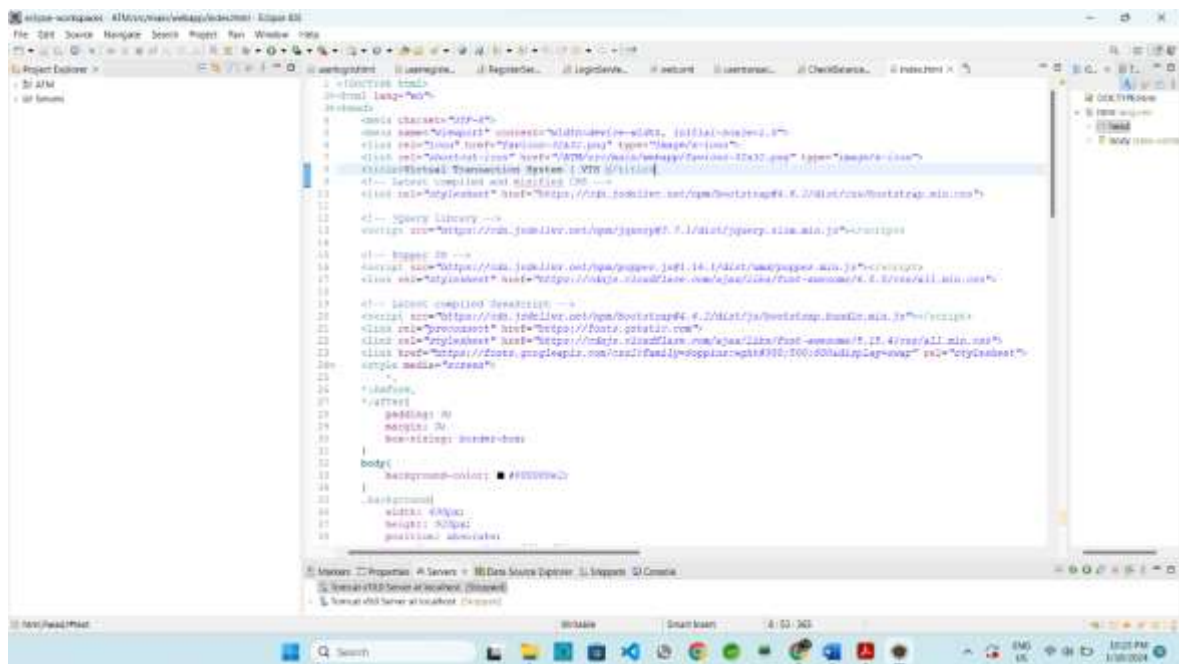


## CheckBalanceServlet.java

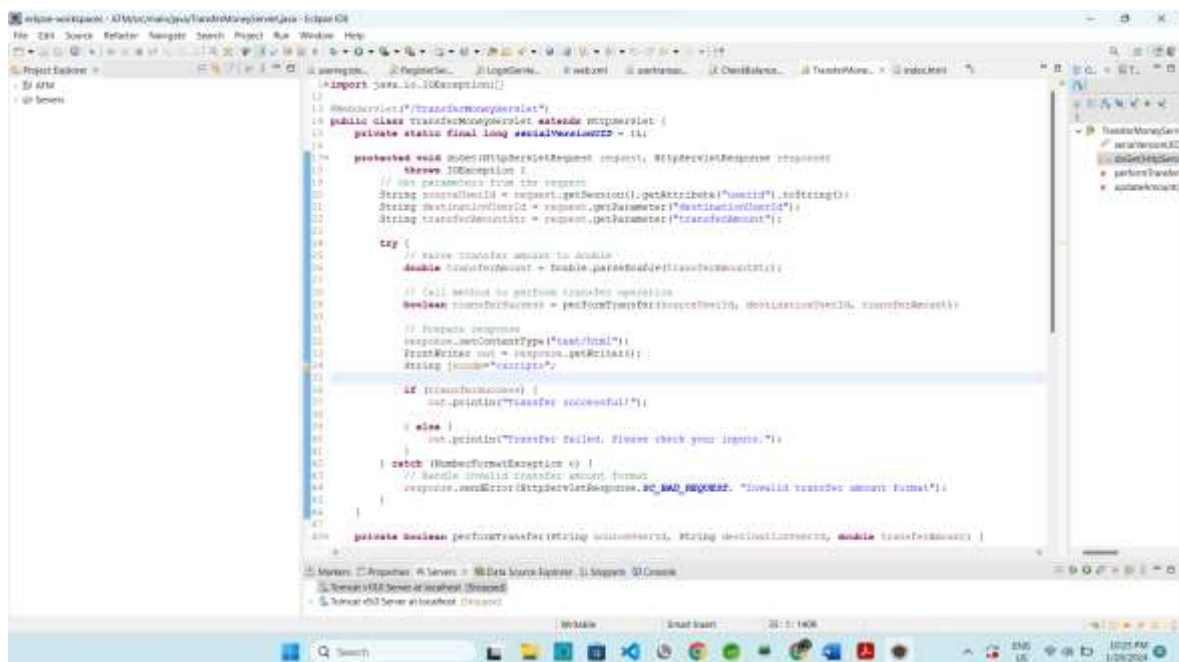


## Index.html

# VIRTUAL TRANSACTION SYSTEM



## TransferMoneyServlet.java



## Code Source File and Path

