**NED UNIVERSITY OF ENGINEERING AND TECHNOLOGY**

**B.E / BS PROGRAMME OF USMAN INSTITUTE OF TECHNOLOGY**

**SOFTWARE ENGINEERING DEPARTMENT**

**Spring 2024**

**SE-312 Software Construction and Development**

**Project Summary Report**

Semester: 6th

* **Project Title:** Online Banking Service System
* **Team Members:**

|  |  |
| --- | --- |
| **Name** | **Responsibility/Task Assigned** |
| Hafsa Shoaib | 21B-054-SE |
| Mahreen Khan | 21B-010-SE |
| Aliza Khan | 21B-205-SE |
|  |  |
|  |  |

**Project Description:**

In this project, we have developed an Online Banking System as a desktop application using Java Swing, AWT, and MySQL with XAMPP for the database. The primary focus of this system is to provide customers with mini statements that detail all their transactions, including deposits and withdrawals. Given our current constraints of not having direct connections with an actual bank, we have implemented methods to simulate deposit and withdrawal functionalities where the customer can manually enter the amounts. This allows us to generate and display mini statements to show transaction history to the customers.

**Problem Statement:**  
Customers need an easy and efficient way to track their financial transactions. Traditional banking methods can be unmanageable and slow in providing real-time transaction updates. This project aims to bridge this gap by offering a streamlined, user-friendly interface for viewing transaction history.

**Objectives:**

* To create a user-friendly desktop application for online banking service.
* To provide real-time mini statements showing transaction history.
* To simulate deposit and withdrawal transactions for demonstration purposes.

**Scope:**  
The project scope includes:

* Developing a desktop application for banking services.
* Implementing functionalities for deposit and withdrawal transactions.
* Storing transaction data in a MySQL database.
* Displaying transaction history through mini statements.

**Constraints and Limitations:**

* The project is limited to a desktop application and does not include mobile or web versions.
* Currently, there is no real-time connection with a bank; transactions are simulated.
* The application is designed for demonstration purposes and may require further enhancements for real-world implementation.

**Frontend Languages/Framework/Libraries:**

**Language:** Java

**Framework:** Java's UI frameworks include Java AWT and Java Swing. This plays a very important role in creating the user experience of Java applications. These frameworks provide a range of tools and components for creating graphical user interfaces (GUIs) that are not only functional but also visually appealing.

### Libraries:

* **Java Swing**: This library is used for creating the graphical user interface (GUI). It includes components such as JFrame, JLabel, JButton, and ImageIcon. Swing is responsible for the overall look and feel of the application, handling user interactions, and rendering the GUI elements.
* **AWT (Abstract Window Toolkit)**: AWT is used in conjunction with Swing for handling the layout and event handling (e.g., ActionEvent and ActionListener). It integrates well with Swing to enhance the GUI capabilities.
* **JDateChooser (com.toedter.calendar.JDateChooser)**: Adds a convenient date picker to the user interface, making date input easier for users.

**Rationale:** Java Swing and AWT were selected due to their robustness and flexibility in creating interactive and responsive user interfaces. These technologies enable us to build a comprehensive user interface that enhances user experience and facilitates easy navigation through the application.

**Functionalities:** The project includes several key functionalities, each contributing to the overall purpose of the application:

1. Login Functionality
2. Sign Up Functionality
3. Deposit Functionality
4. Balance Enquiry Functionality
5. Fast Cash Functionality
6. Withdraw Functionality
7. Mini statement
8. Pin change

**Backend Languages/Framework/Libraries:**

**Backend Languages:** Java and SQL.

**Libraries:**

 **Java Utility (java.util.\*)**: It provides utility classes and functionalities that are commonly used in the backend, it focuses on providing common functionalities such as date and time manipulation, input/output operations etc.

 **JDBC (java.sql.\*)**: Allows the application to communicate with the database, essential for storing and retrieving information.

#### **JDBC** (Java Database Connectivity)

* JDBC is used to establish a connection between the Java application and the MySQL database. It enables the execution of SQL queries and updates, facilitating communication between the frontend components and the backend database.

#### **XAMPP**

* In this project, XAMPP is used primarily to manage the MySQL database, providing an easy-to-configure local server environment.

**Data Processing:**

* Java is the primary language for implementing the business logic of the application. It processes user inputs, such as deposit and withdrawal amounts, and performs the necessary calculations.
* For instance, when a user deposits money, Java methods handle the input validation, business logic, and updating the user's balance.

**Communication:**

* Java communicates with the MySQL database through JDBC (Java Database Connectivity). JDBC allows Java applications to execute SQL queries and updates, enabling data exchange between the application and the database.
* Java also handles the communication between the frontend (user interface) and backend components, ensuring that user actions (like button clicks) trigger the appropriate backend processes.

**SQL**

**Data Storage:**

* SQL is used to define and manage the structure of the MySQL database. It creates tables to store user information, transaction records, and account details.
* When a transaction occurs (e.g., a deposit or withdrawal), SQL queries are executed to insert the transaction details into the appropriate tables.

**Data Retrieval:**

* SQL queries retrieve data from the MySQL database when needed. For example, to display a user's mini statement, SQL queries fetch all transactions associated with the user's account.

**The Best/Unique Feature of your Project:**

**Standout Feature:** The most unique feature of our project is the mini statement functionality that provides customers with real-time transaction history. This feature stands out because it offers a clear and concise view of all transactions, enabling customers to track their deposits and withdrawals easily.

**Significance:** This feature is significant as it addresses the common customer need for quick access to transaction history without the delay typically associated with traditional banking methods. It adds value by enhancing user satisfaction, making financial management more accessible.

**Project and Lab Topics Synchronization:**

* Basics of Java programming, syntax, and structure.
* The project uses Java standard libraries such as java.awt for GUI components and java.sql for database operations, showcasing practical applications of these libraries.
* The project heavily relies on Swing for creating user interfaces (JFrame, JButton, JLabel, etc.), and AWT for basic window operations and image handling.
* The project includes event handling for user interactions, such as clicking buttons (ActionListener), which is a direct implementation of the event handling concepts learned in the lab.
* The project uses JDBC for database connectivity to handle user data, perform transactions, and query account balances.
* The project includes exception handling to manage potential errors during database operations and user interactions, reinforcing the importance of robust error handling in applications.
* Mysterious Name

### Inheritance

1. **Extending Classes**: Main classes like Login, Signup, and Signup2 extend JFrame, inheriting its properties and methods to function as GUI windows.
2. **Implementing Interfaces**: These classes implement the ActionListener interface, requiring the definition of the actionPerformed method for event handling.

### Polymorphism

1. **Method Overriding**: The actionPerformed method is overridden in each class that implements ActionListener, providing specific implementations.