NEXUS

BANKING APP

Institute: FAST-NU

Course: Fundamentals of software engineering

Instructor: Ms. Sania Urooj

Created by:

Mehreen Saghar 22k-4071

Wishma Shahab 22k-4126

Fatima Zafar 22k-4124

Insiya Fakhruddin 22k-4103

Table of Contents

**1. Introduction**

**1.1 Purpose**

**1.2 Scope**

**2. Overall Description**

2.1 Product Perspective

2.2 Product Functions

2.3 User Classes and Characteristics

2.4 Operating Environment

2.5 Design and Implementation Constraints

2.6 User Documentation

2.7 Assumptions and Dependencies

**3. External Interface Requirements**

3.1 User Interfaces

3.2 Hardware Interfaces

3.3 Software Interfaces

**4. System Features**

4.1 User Login

4.2User registeration

4.2.1 Send money.

4.2.2 Bills and top-ups

4.3 Transactions

4.3.1 Settings

4.3.2 Feedback

**5. Other Nonfunctional Requirements**

5.1 Performance Requirements

5.2 Safety Requirements

5.3 Security Requirements

5.4 Software Quality Attributes

**6. Use Case and Activity Diagram**

6.1 Use Case diagram

6.2 Activity diagram

6.3 Gantt chart

**7. Testing**

7.1 Introduction

7.2 Objectives

7.3 Testing Methods

7.4 Test Cases

**Introduction**

**1.1 Purpose**

The purpose of this Software Requirements Specification (SRS) document is to outline the requirements for the development of "Nexus," a banking application designed to provide users with secure and efficient online banking services. This document will define the features, functionalities, constraints, and performance requirements of the software system.

**1.2 Document Conventions**

This document adheres to standard conventions for writing SRS documents.

**1.3 Intended Audience and Reading Suggestions**

This document is intended for software developers, project managers, and stakeholders involved in the development of the "Nexus" banking application. Readers should have a basic understanding of software development and banking system architecture.

**1.4 Product Scope**

The scope of the "Nexus" banking application project is to create a comprehensive banking platform accessible through both web browsers and mobile applications. The software will offer various functionalities including account management, fund transfers, bill payments, loan applications, and other standard banking services.

The application will have one main user interfaces: User. The login page will allow users to select their role (admin, customer, or support staff) and access the relevant features.

**1.5 References**

No external references were consulted during the creation of this document.

**Overall Description**

**2.1 Product Perspective**

The "Nexus" banking application is positioned as a modern and versatile solution that revolutionizes the way users interact with banking services. It is designed to provide a comprehensive and user-friendly platform for managing financial transactions securely and efficiently.

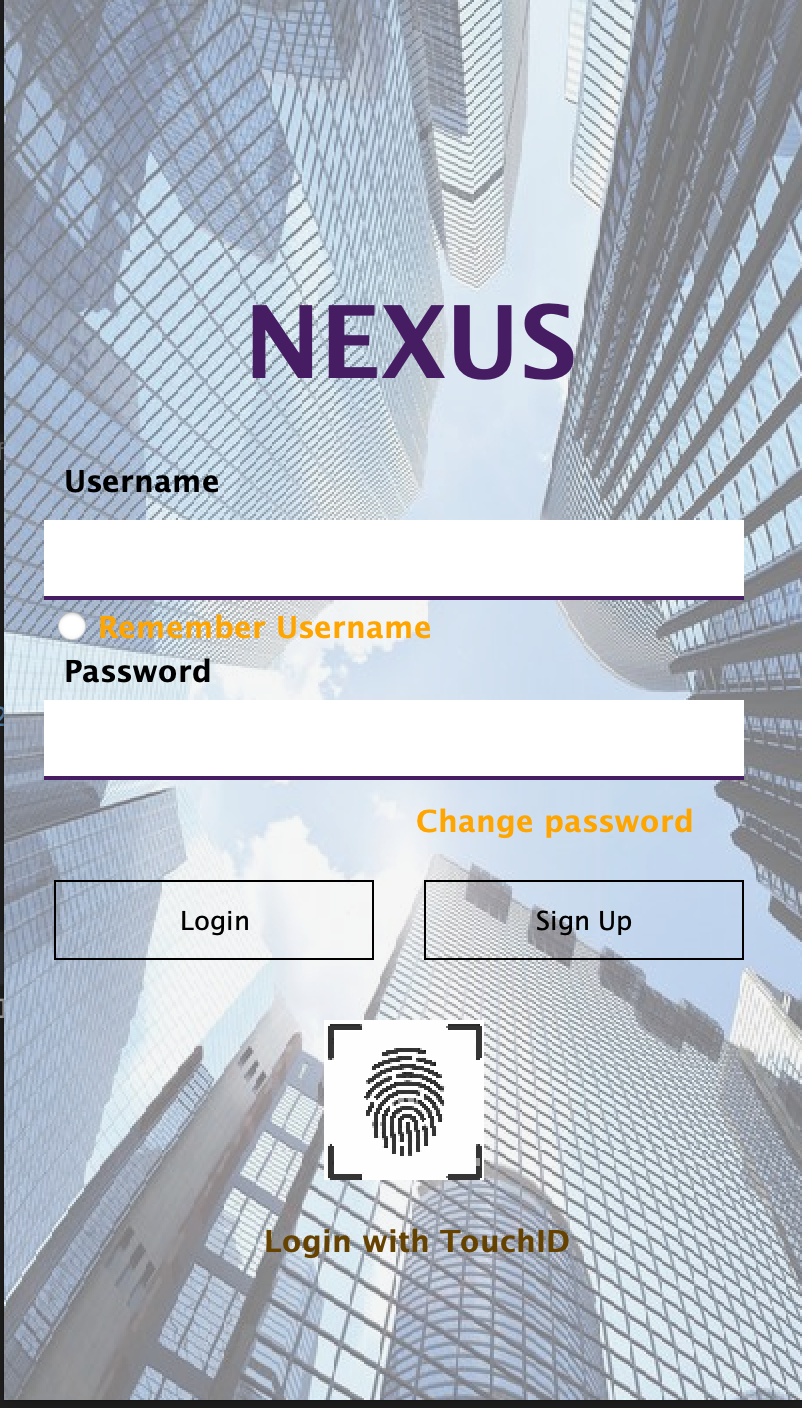
**2.2 Product Functions**

The following are the functional requirements of the "Nexus" software:

**2.2.1 Login Interface**

The login interface should allow the user to enter username and a 4-digit pin, upon verification it directs user to the main page of the app.

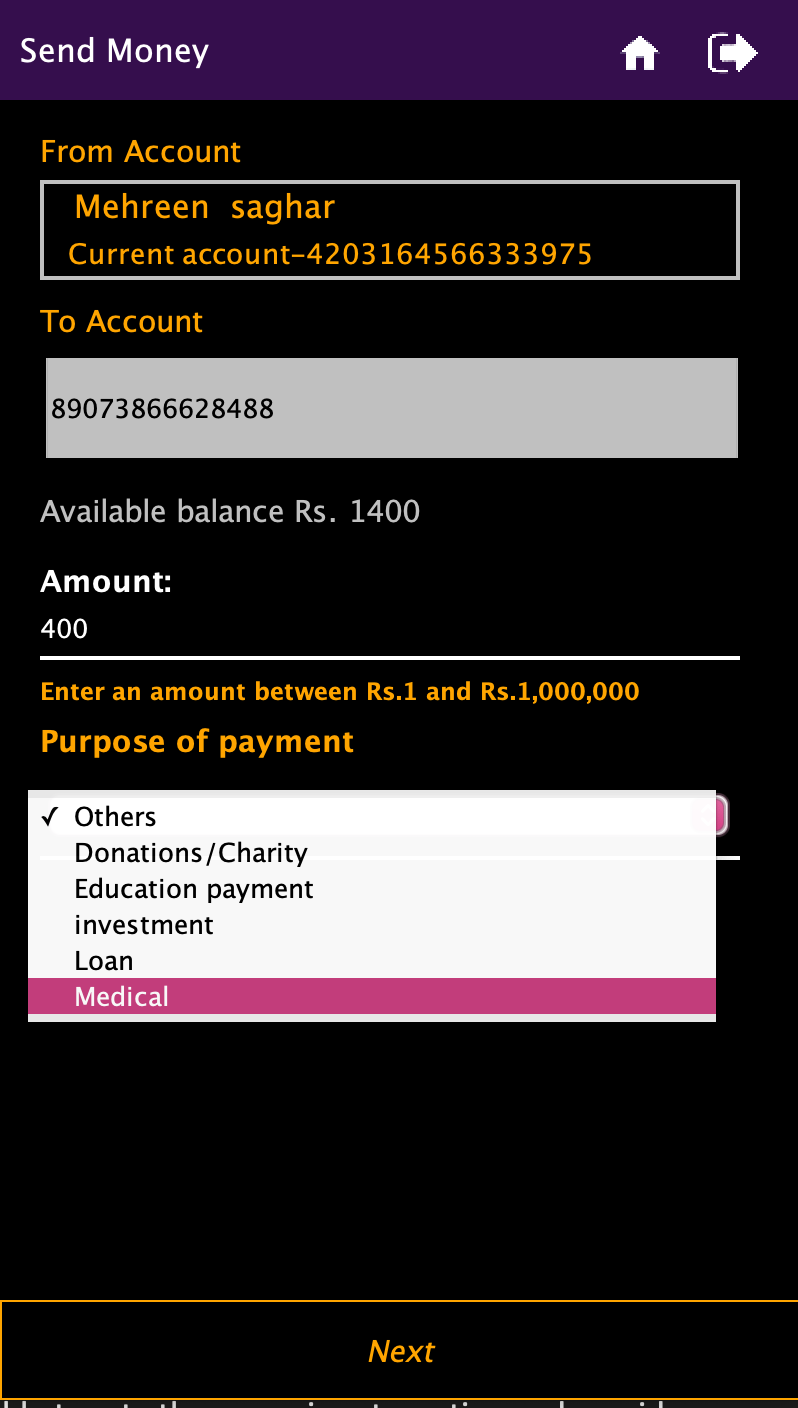
If the user is a new user, they can sign up and an create account.



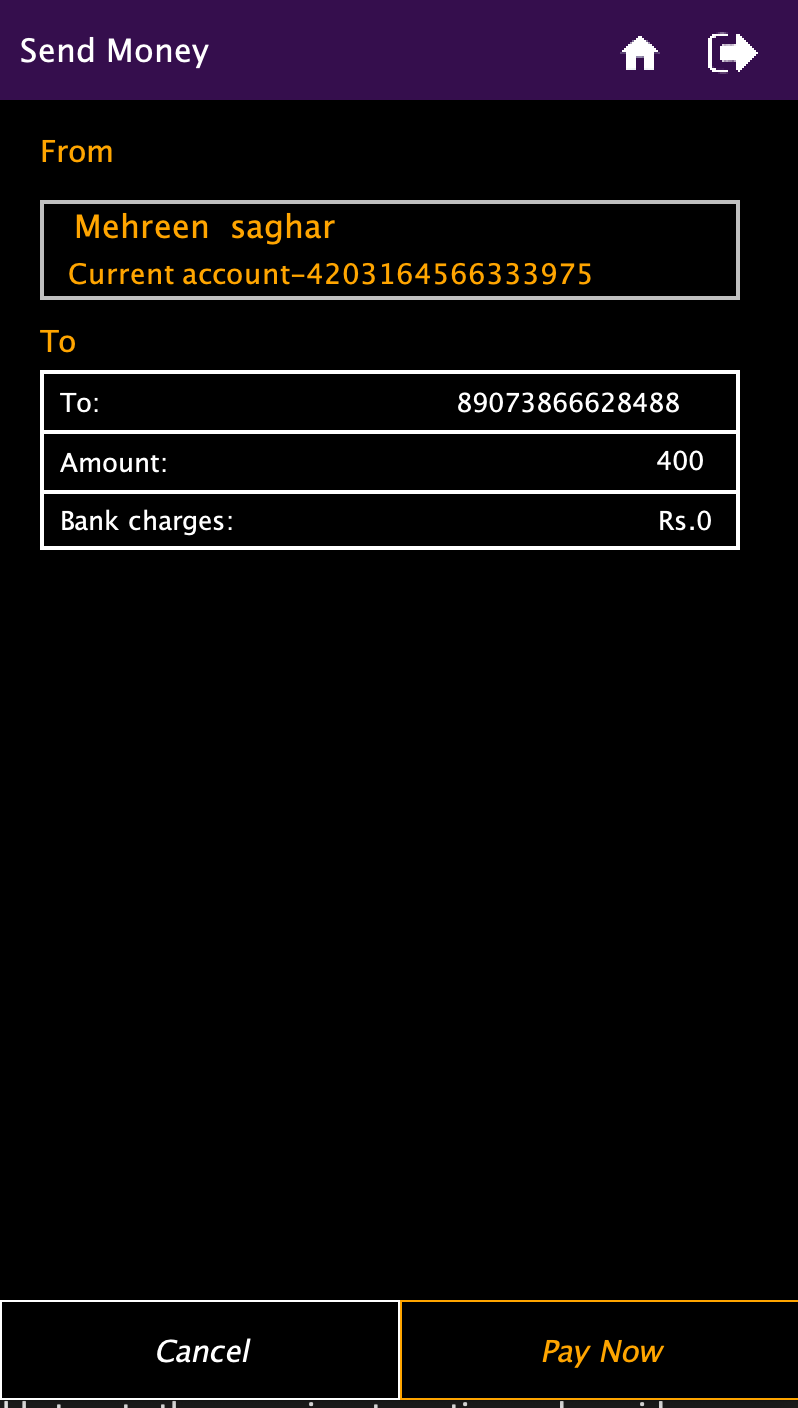
**2.2.2 Customer Interface**

The customer interface should provide the following functionalities:

* Send Money: The customer should be able to enter amount they want to send to an account number and choose purpose of payment.



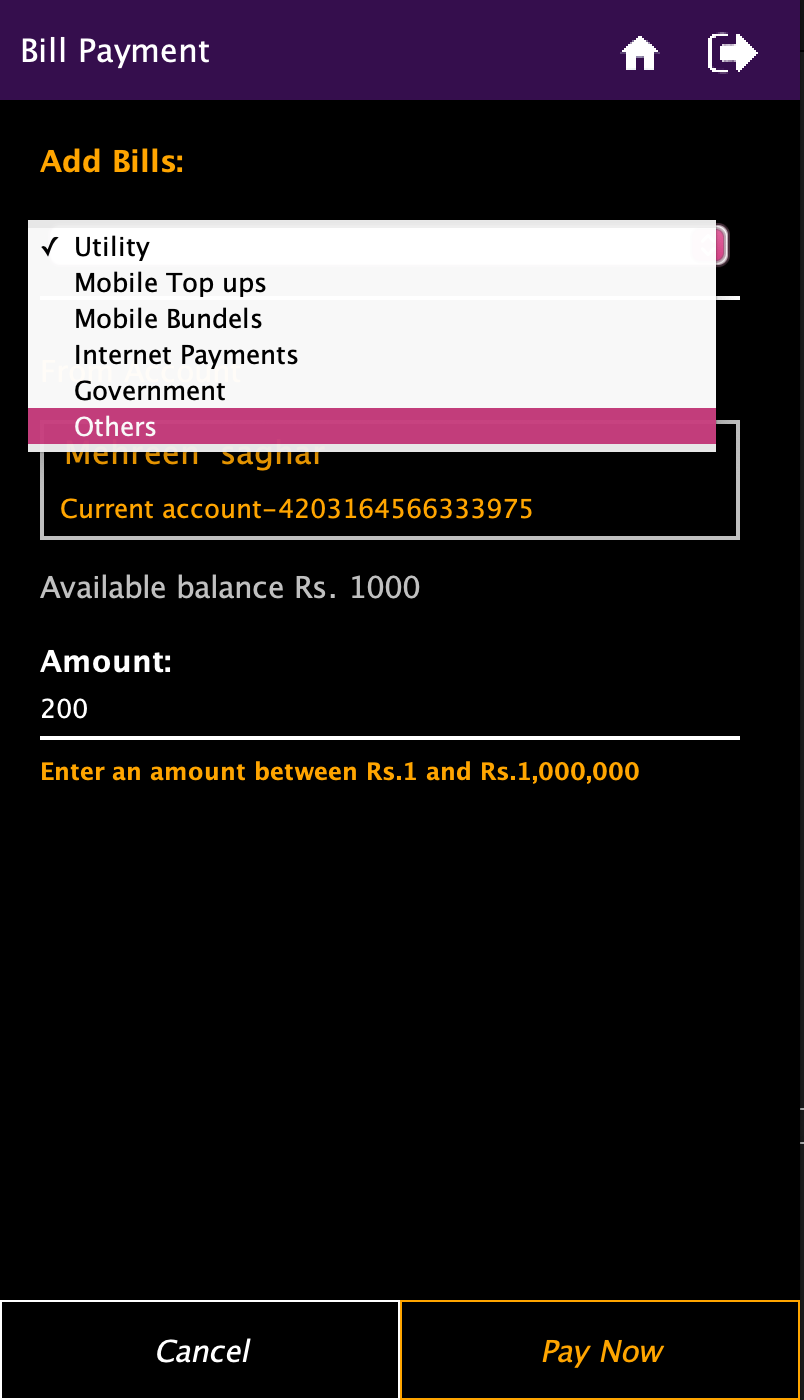
After user selects the details and selects ‘next’ they are directed to a confirmation page:



After confirmation the program send an OTP to the users email account to confirm the transaction



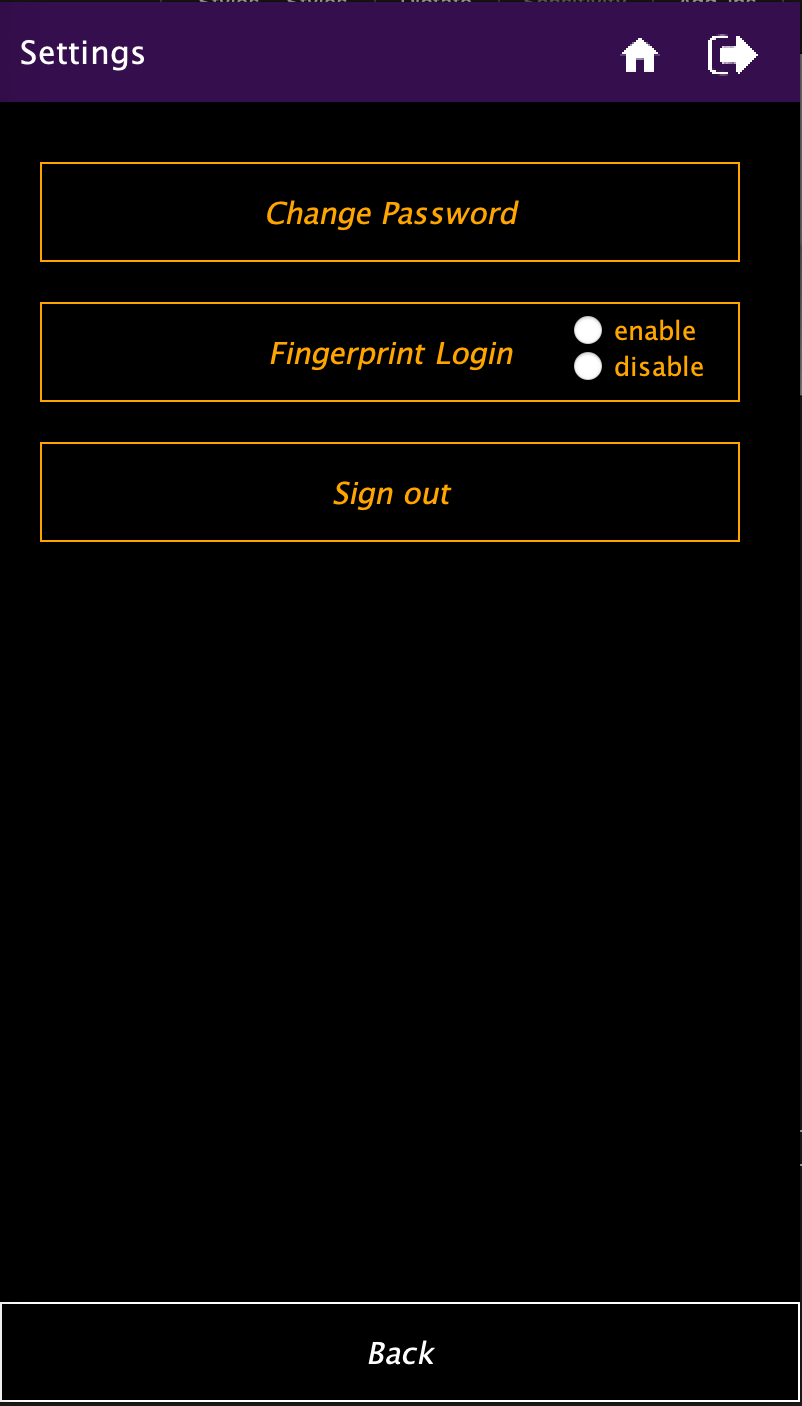
* Bill payment: The customer should be able to enter amount they want to send to an account number and choose the details of the bill.



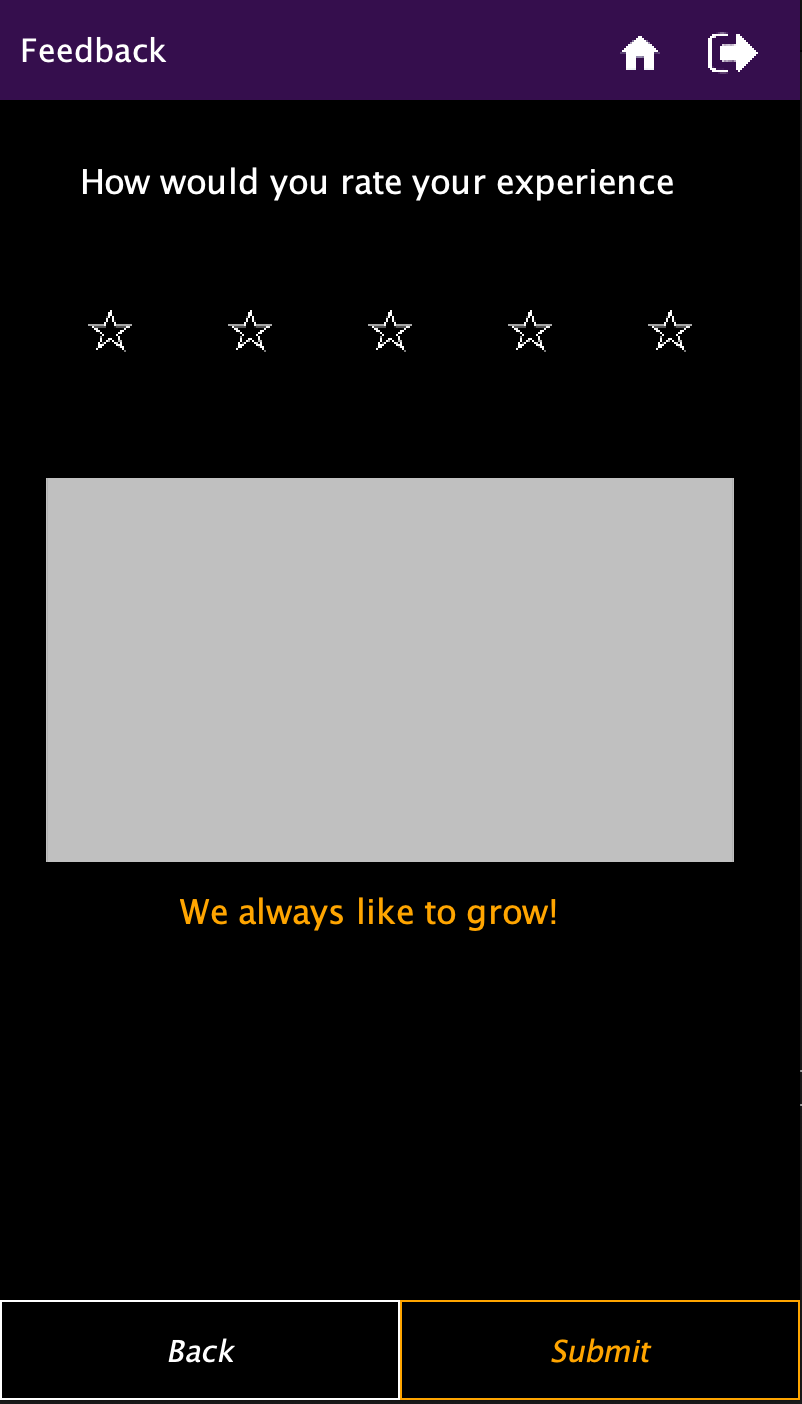
* Transaction: The customer should be able to perform all kinds of ATM transactions through this section including deposit, withdraw, fast cash and check mini statement.



* Settings: The customer should be able to change pin, select whether they want to enable fingerprint sensor and sign out of their account



* Feedback: The customer should be able to rate the app using star rating and provide detailed feedback on their experience.



**2.3 User Classes and Characteristics**

The "Nexus" bank will have five Main user classes:

* Send Money
* Bill payment and top ups
* ATM transactions

1. Deposit
2. Withdraw
3. Fast cash
4. Mini statement
5. Balance inquiry

* Settings
* Feedback

These are the main requirement for the idea of a banking app, Other than these the other classes are:

* Login
* Sign up.
* OTP
* Change password.

**2.4 Operating Environment**

The “Nexus” banking application will be a web-based platform accessible through browsers and mobile applications.

Frontend: Java Swing and JFrames

Backend: Java programming Language

**2.5 Design and Implementation Constraints**

The "Nexus" banking application is subject to various design and implementation constraints that shape its development and operational framework. These constraints are essential considerations to ensure the system's reliability, security, and scalability:

* Security Measures:

The "Nexus" banking application must prioritize the highest standards of security to protect user data and transactions. Robust encryption protocols, secure authentication mechanisms, and comprehensive access controls are imperative to safeguard sensitive information from unauthorized access or malicious activities.

* Scalability Requirements:

To accommodate future growth and an increasing user base, the software architecture must be designed with scalability in mind. Scalable database solutions, distributed computing technologies, and load balancing mechanisms are essential to support expanding transaction volumes and user interactions without compromising performance.

* Concurrency Management:

The system must effectively handle multiple concurrent user requests without experiencing performance degradation or system instability. Efficient concurrency management techniques, such as thread pooling, asynchronous processing, and resource allocation optimization, are necessary to ensure smooth operation under

**2.6 User Documentation**

The Nexus Bank software system must have comprehensive user documentation that is easily accessible and understandable to all user classes. The user documentation should provide step-by-step instructions on how to use the software system, including how to log in, sign up, perform transactions, bill payment, and use settings. The user documentation must also provide troubleshooting tips and frequently asked questions to help users resolve common issues.

**2.7 Assumptions and Dependencies**

The following assumptions and dependencies are made during the development of the Nexus bank software system:

* The system assumes that users have access to a reliable internet connection and compatible devices.
* The system depends on the availability and reliability of third-party payment gateways and shipping options.
* The system assumes that users have basic computer literacy skills to navigate the software system and follow instructions.
* The system depends on the availability and compatibility of modern programming languages and frameworks.

**External Interface Requirements**

**3.1 User Interfaces**

The Nexus bank software system has one main user interfaces: user. The login page serves as the gateway to this interface, allowing users to either login in an existing account or sign up.

The user interfaces must be designed to be user-friendly and intuitive, allowing users to easily navigate the system and perform tasks without encountering issues.

**3.2 Hardware Interfaces**

The Nexus software system does not have any specific hardware requirements. The system should have at least two cores, 1.6 Ghz clock frequency, and 2GB of ram. It is compatible with most modern devices, including desktops, laptops, tablets, and smartphones, as long as they have a reliable internet connection.

**3.3 Software Interfaces**

The software interfaces of the Ez Mart system are as follows:

**3.3.1 Application Programming Interface (API)**

The Nexus bank system provides an API for third-party applications to access and interact with the system's functionality. The API will be based on REST (Representational State Transfer) principles and will provide secure access to data and functionality.

**3.3.2 Payment Gateway**

The Nexus bank system will integrate with one or more payment gateway providers to enable customers to make online payments for their purchases. The payment gateway will be responsible for securely processing payments and communicating payment status back to the system.

**3.3.3 Database Management System (DBMS)**

The Nexus Bank system will use a DBMS to store and manage data. The DBMS will be responsible for ensuring data integrity, reliability, and consistency. The system will support MySQL.

**3.3.24 Email**

The system will use email to send notifications to users about their orders, products, and other relevant information.

**System Features**

The following section describes the features and functionality of the Nexus bank system:

**4.1 User Management**

The following features will be supported:

**4.1.1 User Login**

The system will provide a login page from where users can login to an existing account by entering their credential.

**4.1.2 User Registration**

The system will allow user to register for an account by providing their basic information such as name, email address, phone number.

**4.1.3 User Profile Management**

The system will allow users to manage their profile information such as name, email address, phone number, and password.

**4.2 User Interface**

**4.4.1 Send Money**

Description: The customer can add the account number and the amount, which is visible to them, then select the purpose of payment and choose next option.

This will direct them to confirmation page that shows all transaction details, from account, to account, amount, and bank charges. If the user confirms by choosing “pay now “option they are asked to enter the OTP that has been sent to their email address.

Upon verification the transaction is conducted, and the user is informed.

**4.4.2 Bills and top-ups**

Description: The customer can add the account number and the amount, which is visible to them, then select the type of bill.

If the user confirms by choosing “pay now “option they are asked to enter the OTP that has been sent to their email address.

Upon verification the transaction is conducted, and the user is informed.

**4.4.3 Transactions**

Description: The customer should be able to perform all kinds of ATM transactions through this section including deposit, withdraw, fast cash and check mini statement.

* + 1. **Settings**

Description**: The** customer should be able to rate the app using star rating and provide detailed feedback on their experience.

**4.4.4 Feedback**

Description**:** The customer should be able to rate the app using star rating and provide detailed feedback on their experience.

5.**Other Nonfunctional Requirements**

**5.1 Performance Requirements**

* **Response Time**: The system should respond to user requests within 2 seconds.
* **Concurrent Users**: The system should support up to 10,000 concurrent users.
* **Data Handling**: The system should efficiently manage large volumes of data without performance degradation.

**5.2 Safety and Security Requirements**

* **User Authentication**: Implement robust user authentication and authorization mechanisms to restrict access to authorized users only.
* **Data Encryption**: Encrypt and securely store user data, including personal and financial information.
* **Prevention of Unauthorized Access**: Design the system to prevent unauthorized access and data breaches.
* **Regular Security Testing**: Conduct regular security testing to identify and address vulnerabilities promptly.

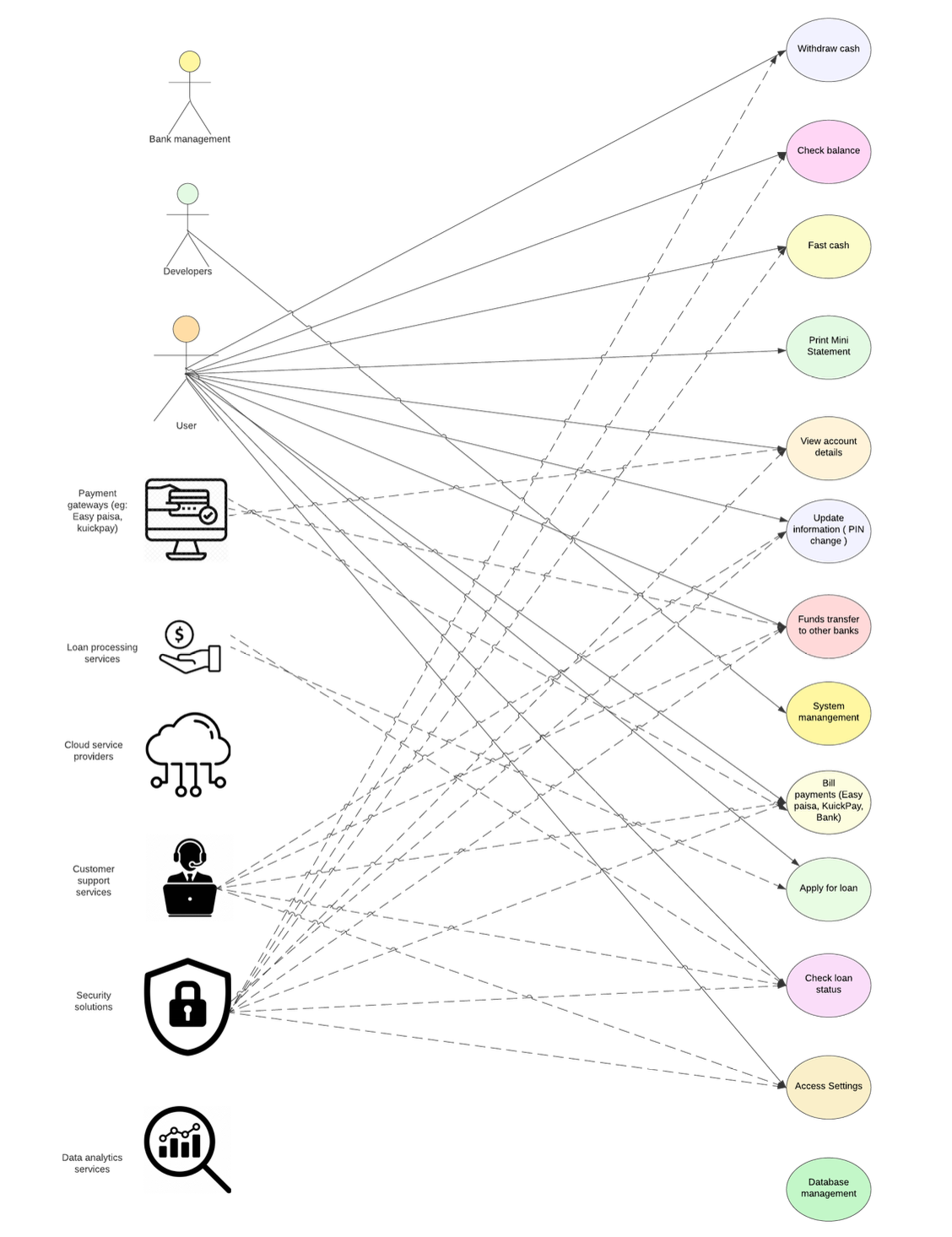
**5.3 Additional Security Requirements**

* **Prevention of Common Attacks**: Design the system to prevent common attacks such as cross-site scripting (XSS), SQL injection, and buffer overflow.
* **Denial of Service (DoS) Prevention**: Implement measures to detect and prevent denial of service attacks.
* **Encryption for Communication**: Use encryption for all communication between the server and client to ensure data confidentiality.

**5.4 Software Quality Attributes**

* **Reliability**: Ensure the system is available 24/7 with minimal downtime.
* **Maintainability**: Design the software with modular and reusable components to facilitate easier maintenance and updates.
* **Scalability**: Ensure the system can handle increasing user loads and data volumes without significant performance degradation.
* **Usability**: Design an intuitive and user-friendly interface for seamless user interaction.

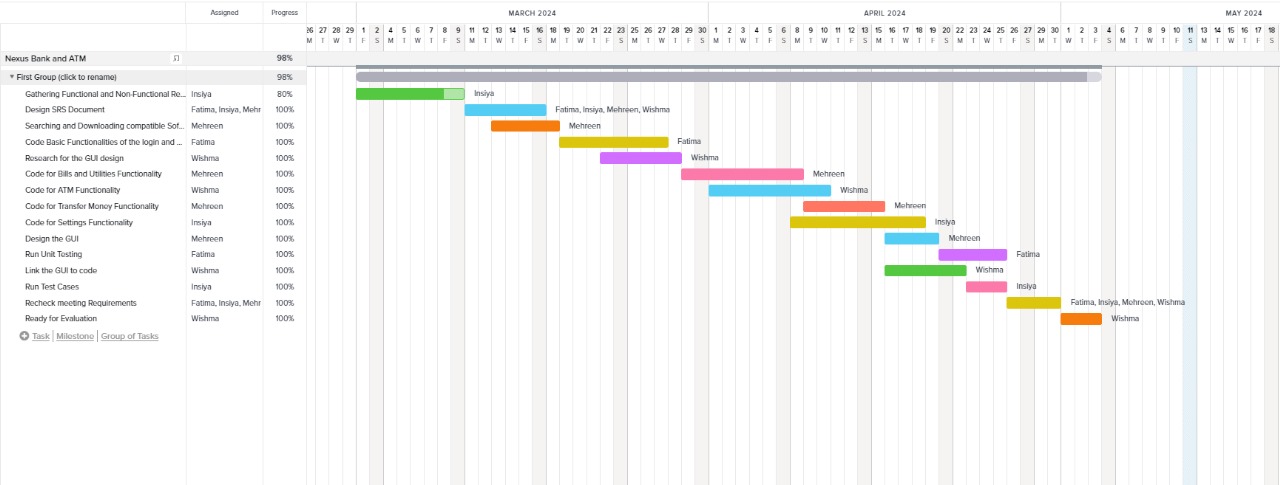
**Use Case and Activity Diagram**

**6.1 Use Case diagram6.2 Activity diagram**

**A diagram of a company

Description automatically generated**

**Gantt Chart**



**Testing**

**7.1 Introduction**

The testing section of this SRS report outlines the testing approach and results for key features of the "Nexus" banking application. This section includes test cases, expected results, and actual results obtained during testing.

**7.2 Objectives**

The objective of this testing section is to ensure that critical features of the "Nexus" banking application, including account management, fund transfers, and security measures, are functioning as intended and meeting the requirements outlined in the SRS report. Rigorous testing aims to identify any defects or issues that may impact the functionality of the system and provide recommendations for improvement.

**7.3 Testing Methods**

The testing phase employs various methods to validate the functionality and performance of the "Nexus" banking application:

**7.3.1 Unit Testing**

Unit testing focuses on individual components of the software to ensure they work as intended. Each feature of the "Nexus" banking application, such as account management and fund transfers, undergoes thorough unit testing.

**7.3.2 Integration Testing**

Integration testing evaluates how different components of the software work together as a cohesive system. The "Nexus" application undergoes integration testing to verify seamless interaction between features like account management and transaction processing.

**7.3.3 System Testing**

System testing assesses the entire application to ensure it meets desired requirements. "Nexus" undergoes comprehensive system testing to validate its functionality, security measures, and user interface across different platforms.

**7.3.4 User Acceptance Testing (UAT)**

User Acceptance Testing (UAT) involves end-users testing the application to ensure it meets their needs and expectations. "Nexus" undergoes UAT to verify its user-friendliness, effectiveness in performing banking operations, and compliance with user requirements.

Testing methods are executed using appropriate tools and techniques to ensure accuracy and reliability. The following sections present test cases and results for key features of the "Nexus" banking application.

**7.4 Test Cases**

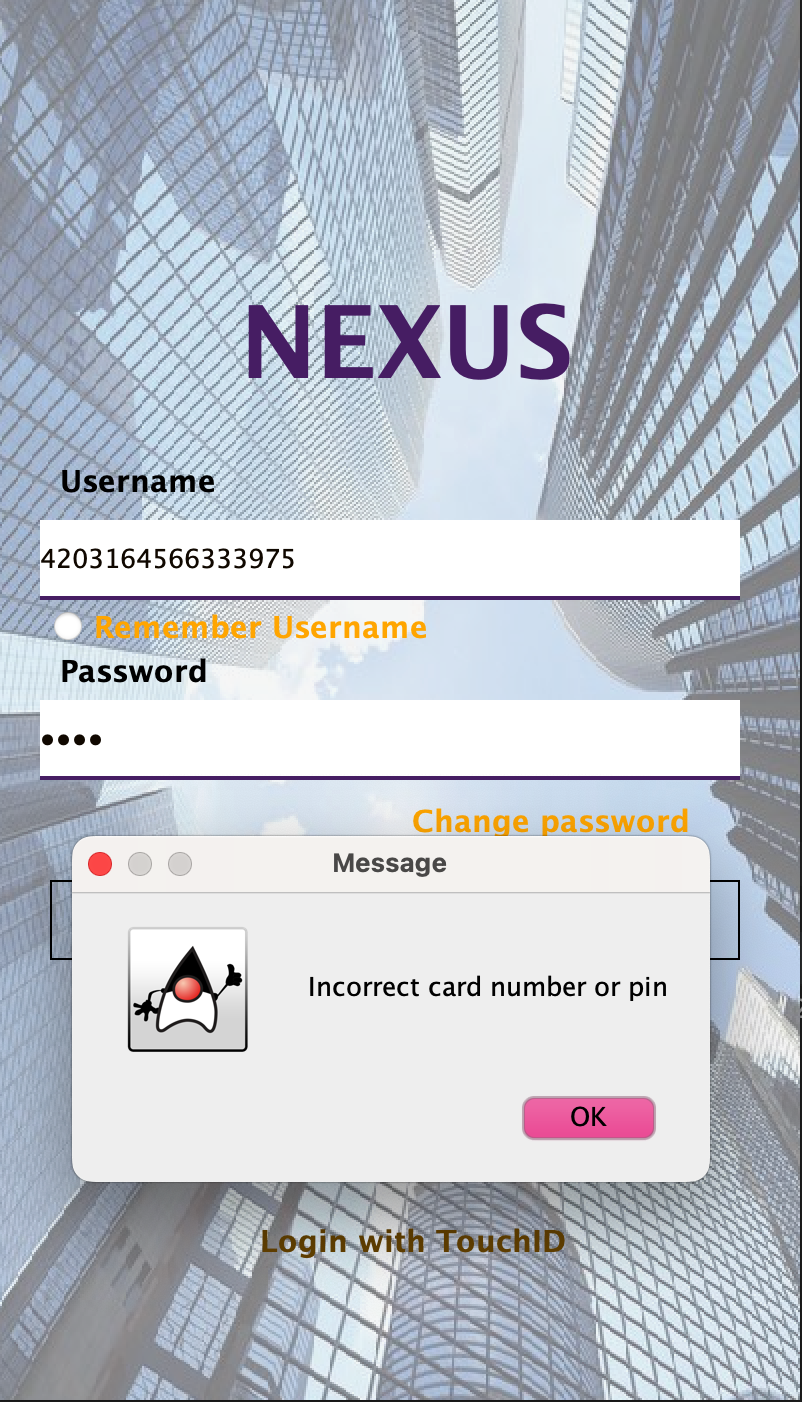
This section provides a comprehensive overview of the various test cases conducted on the Nexus bank project. This section outlines the different testing scenarios performed on the system's major features, including login, send money, Transaction, bill. It includes the methodology used to conduct the tests, the criteria used to determine success or failure, and the results obtained from each test case. This section serves as a valuable resource for project stakeholders and development teams to evaluate the system's performance and ensure that it meets the project's objectives.

**7.4.1 Login Feature**

Test Case 1: Verify that the user is able to log in with valid credentials.

Test Result: Pass opens the Main page.

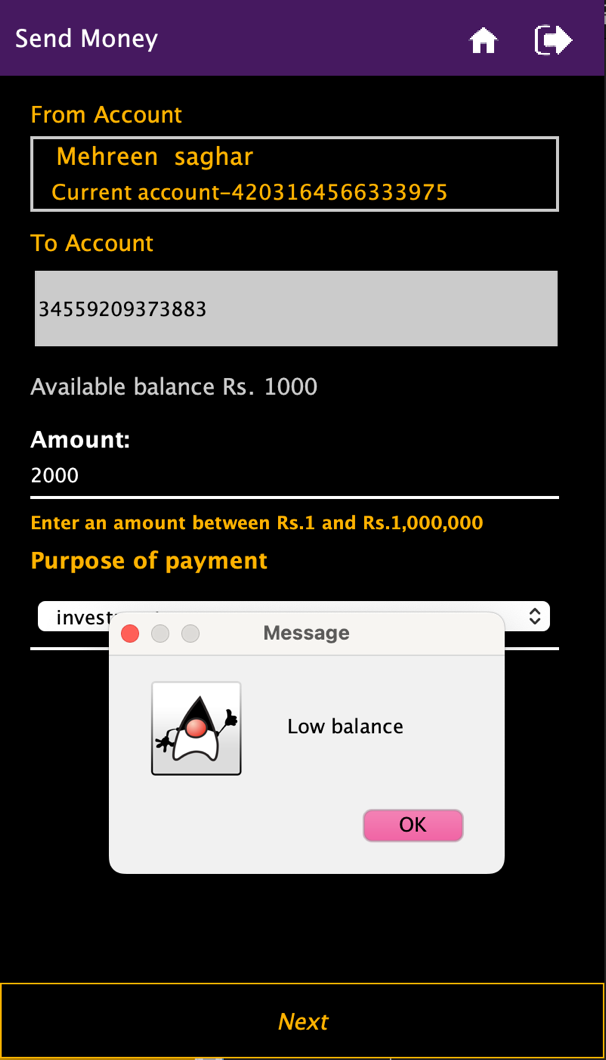
Test Case 2: Verify that the user is not able to log in with invalid credentials.



Test Result: Pass

**7.3.2 Transactions**

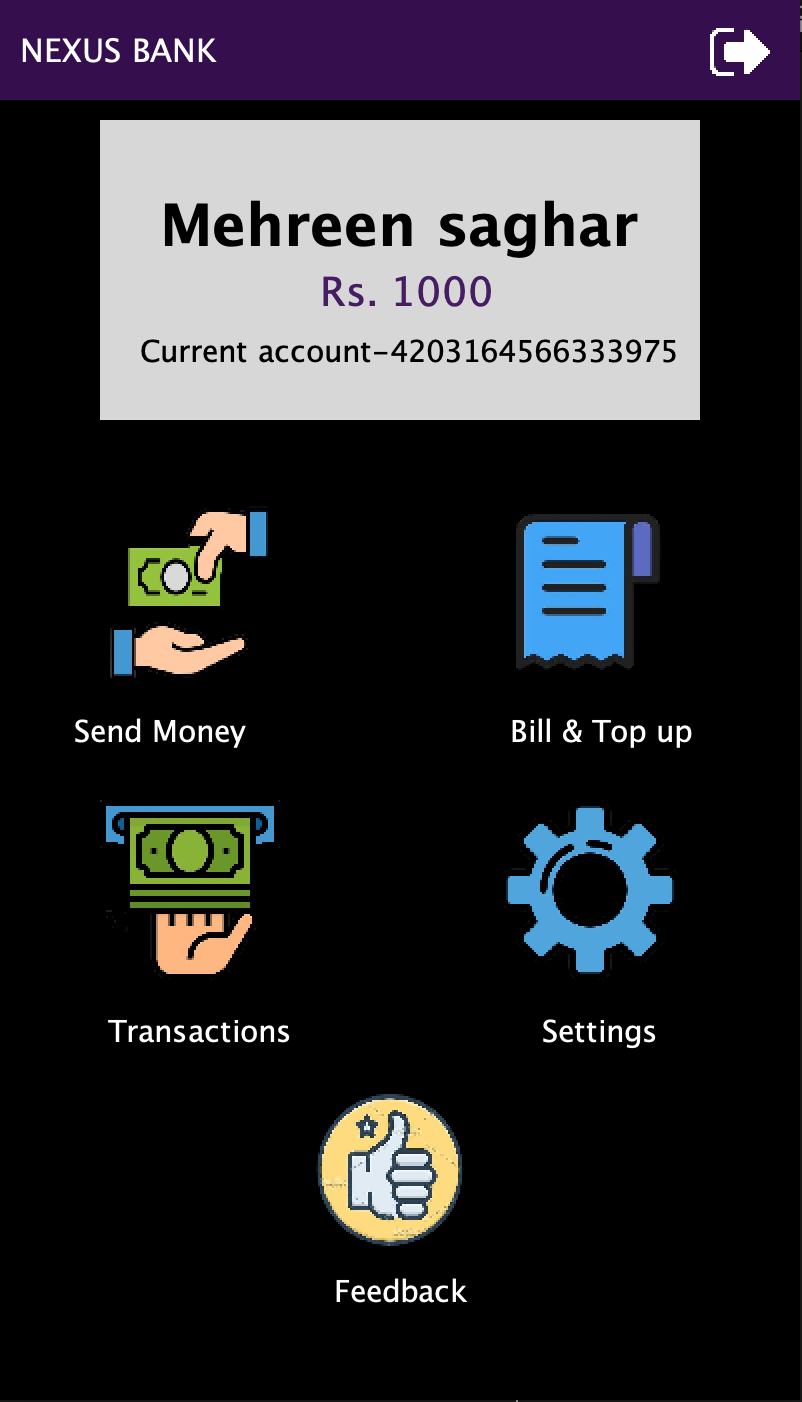
Test Case 1: Verify that user cannot send an amount greater than balance in send money, bill payment and withdraw.

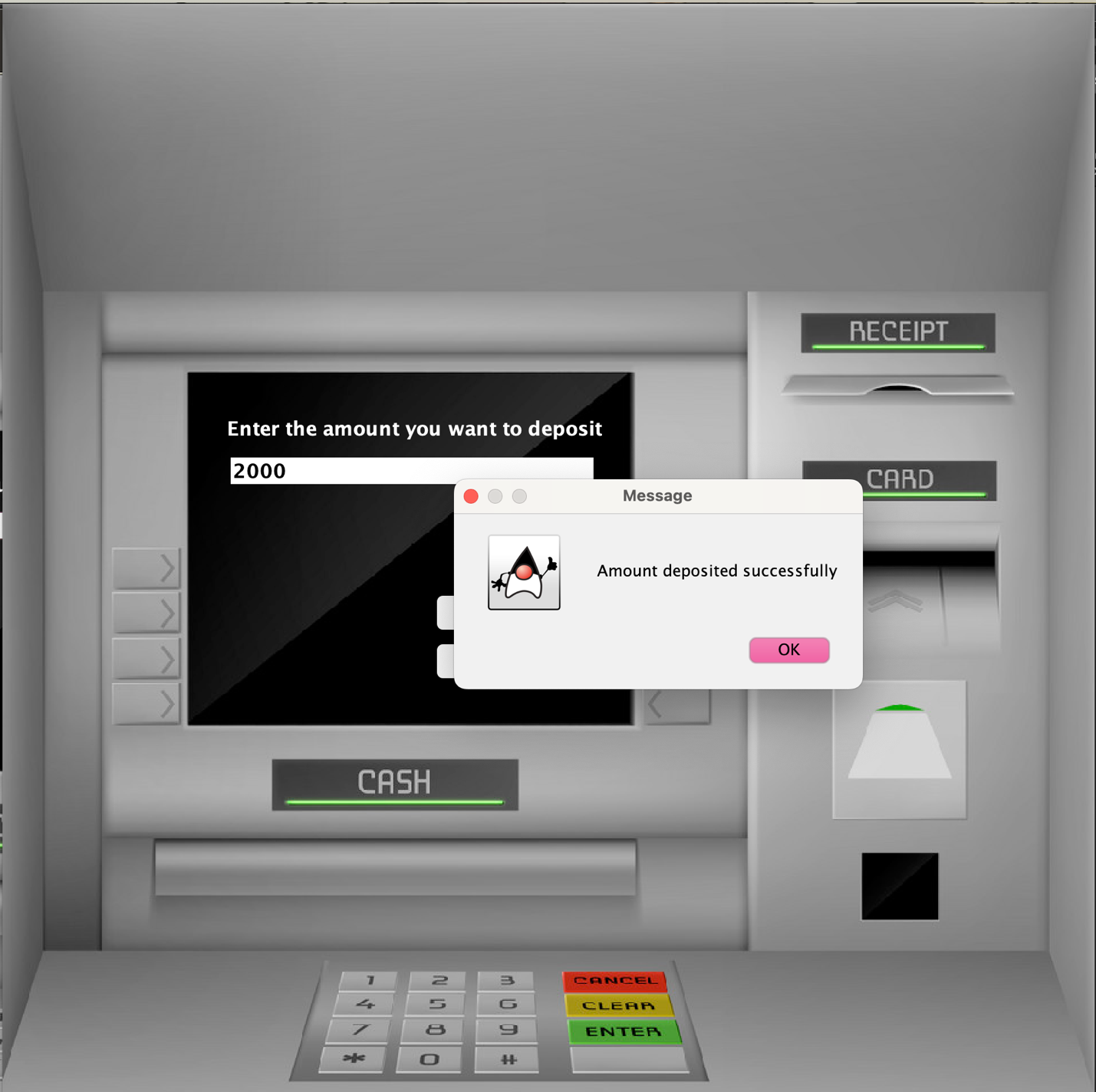


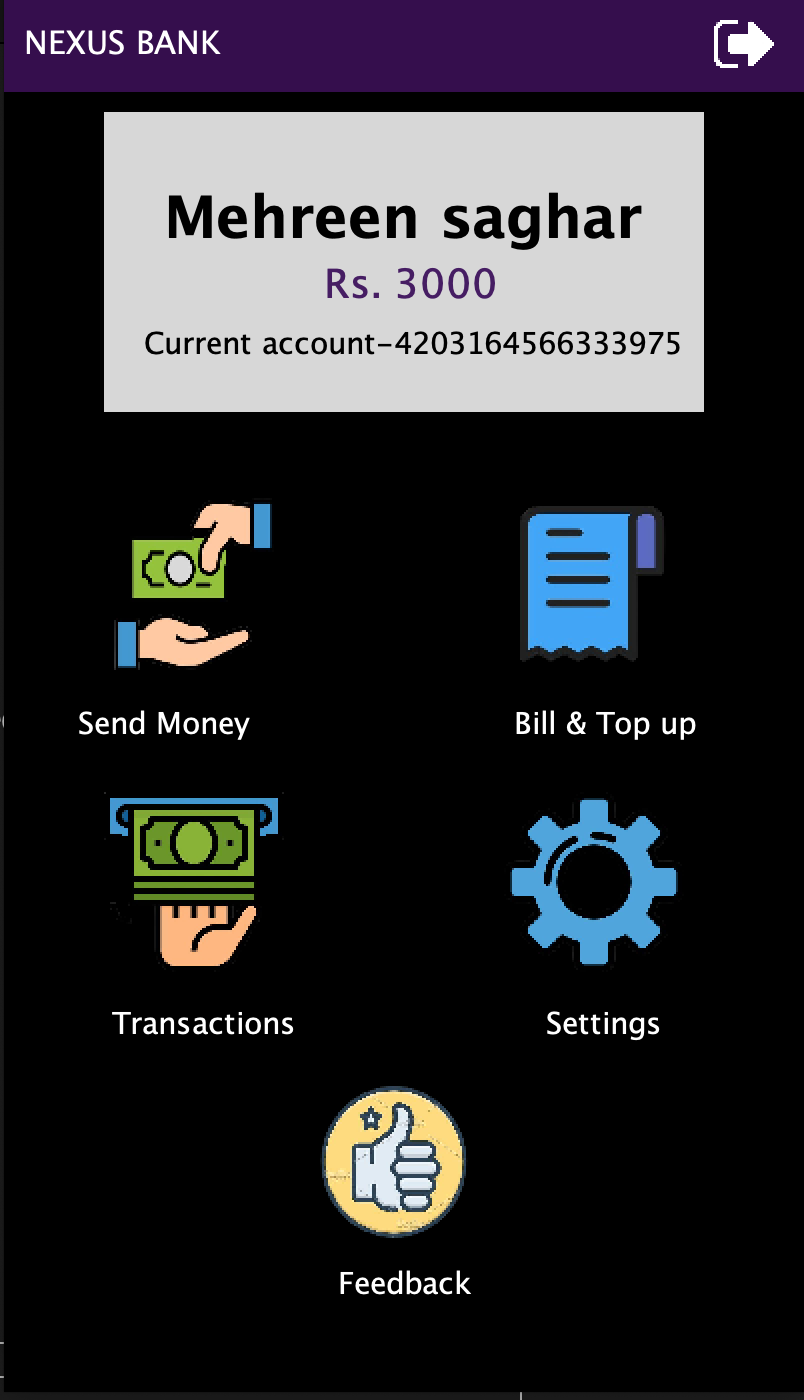
Test Result: Pass

**7.3.2 Update balance**

Test Case 1: Verify that with every transaction the balance is being updated.



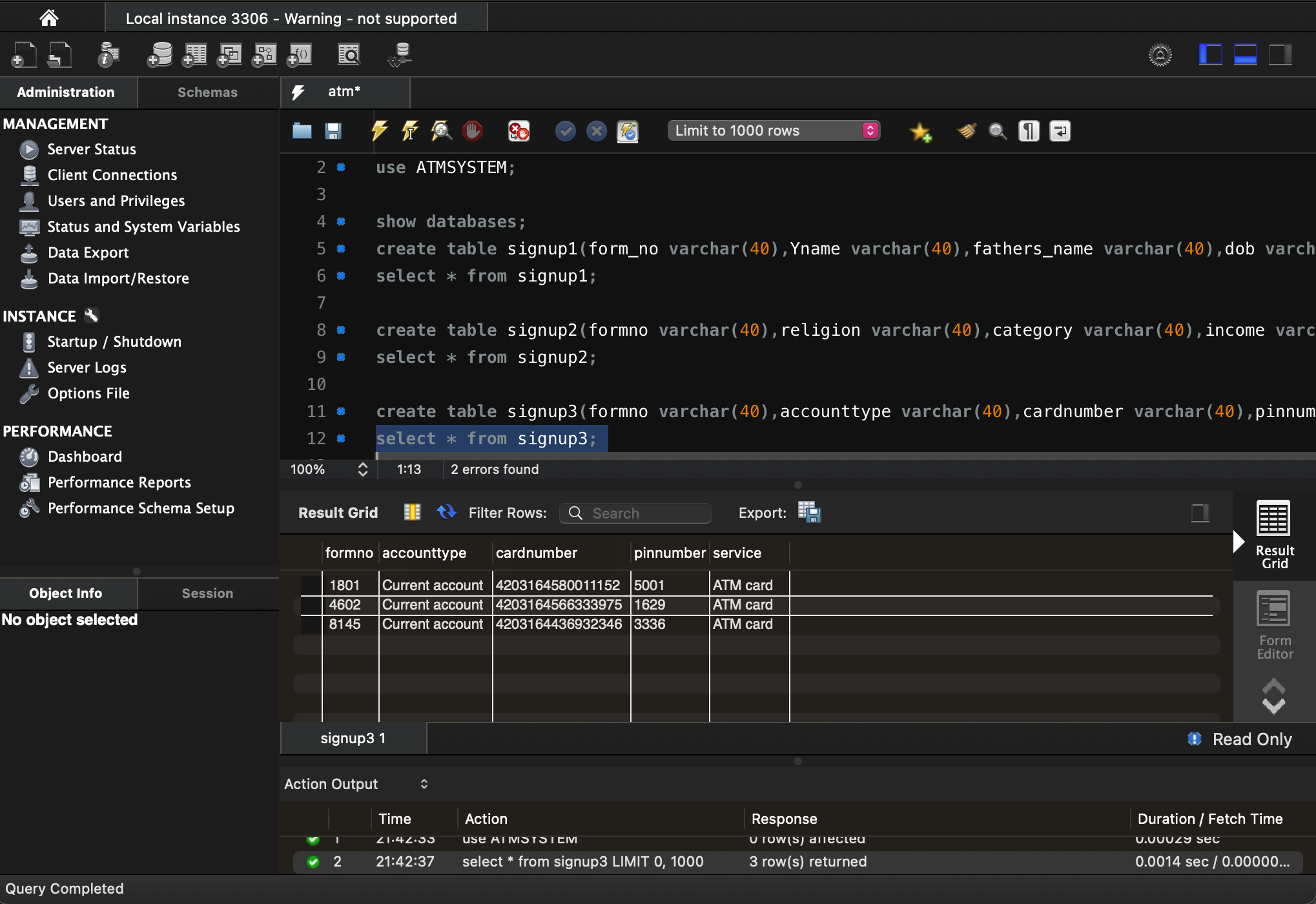
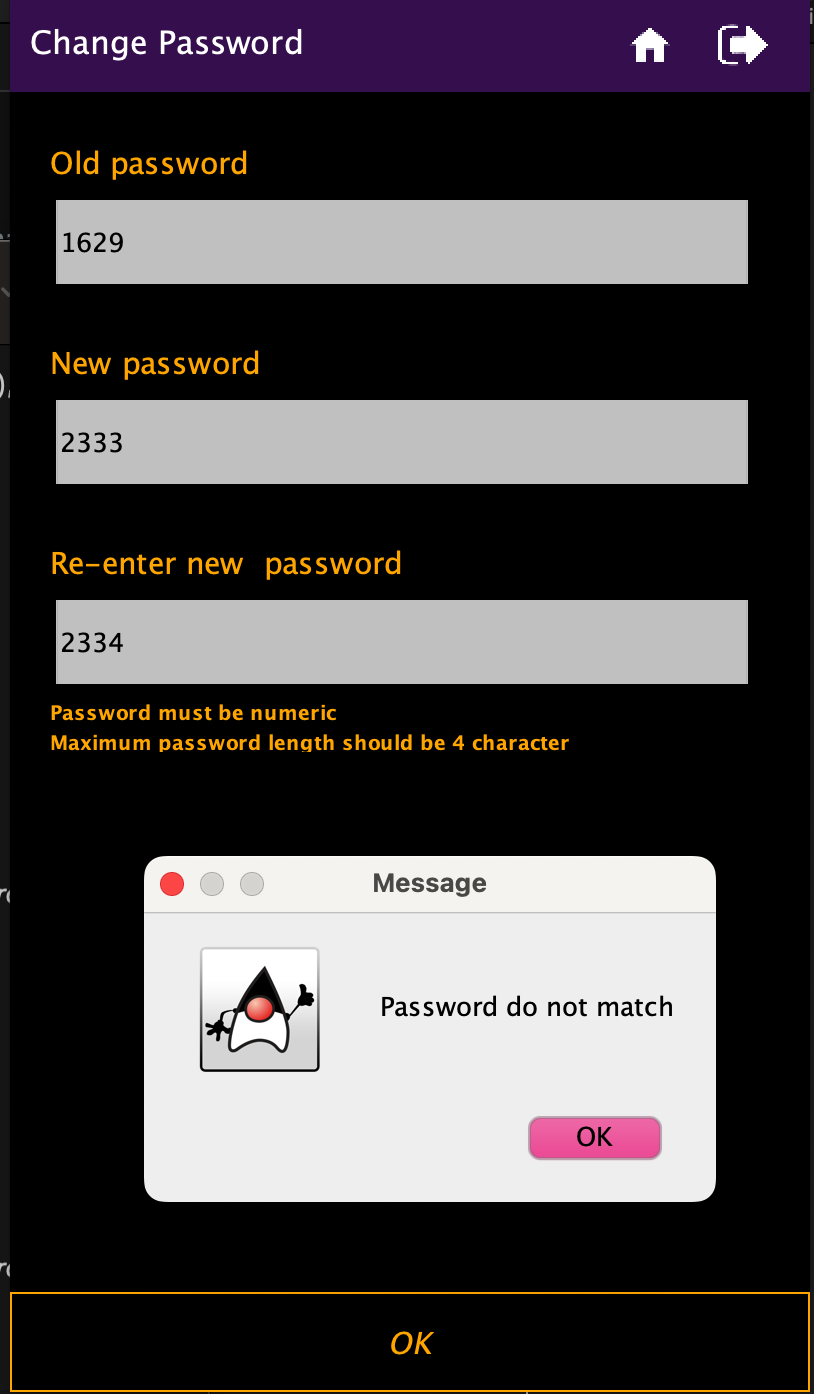


****

Test Result: Pass

**7.3.2 Change password**

Test Case 1: Verify that the password is being changed in the data base.



Test Result: Pass