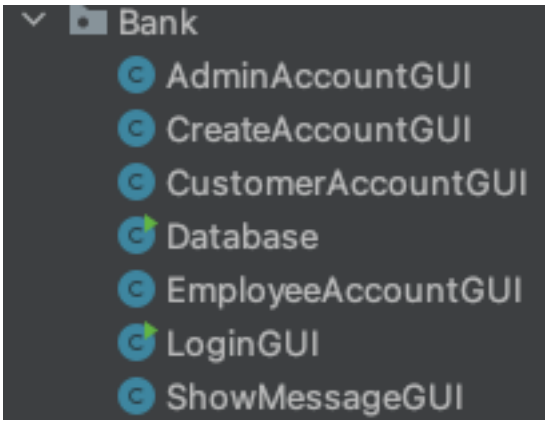
Table Of Contents

Introduction

This project aims to develop a versatile and user-friendly banking system with different sign-in interfaces and functionalities for three types of users: customers, employee, and administrators. This system is designed to streamline banking processes, enhance user experience, and ensure secure transactions. It provides a reliable and secure sign-in interface for customers, employees, and admins. It also enables easy management of customer accounts, banking transactions and administrative tasks.

System Architecture



Database

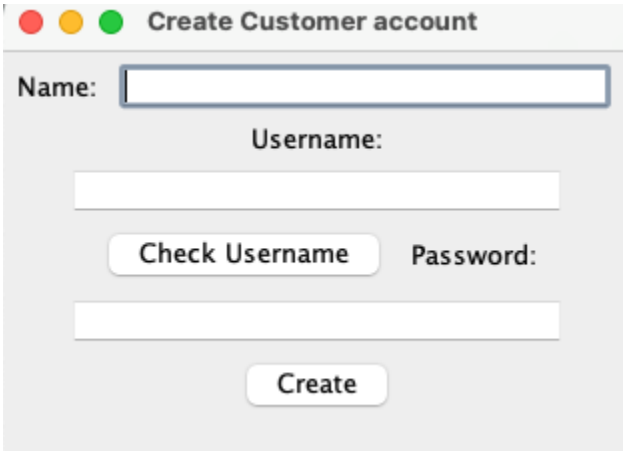
This class is charge of database interactions. It is only responsible for creating connections between project and MySQL.

LoginGUI

This class is a central component of the banking system user interface, which is designed to handle user authentication and navigation between different types of user accounts: customers, employees, and administrators. It also provides create customer account in this class. Method authentication (String username, String password, String type), first check type. If type is “customer”, pull data from customer table. If type is “employee”, pull data from employee table. If type is “admin”, pull data from admin table.

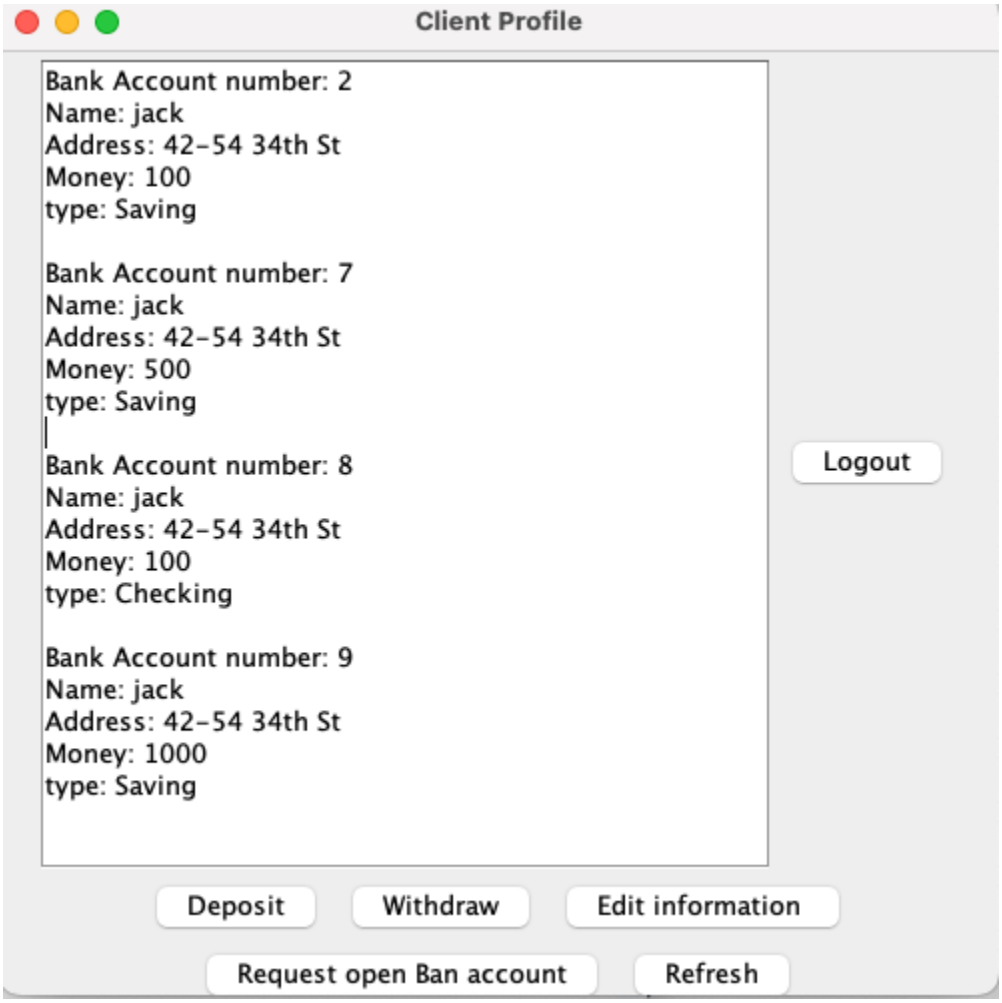
And then compare input username and password with data from database. If all correct, then login to account.

CreateAccountGUI



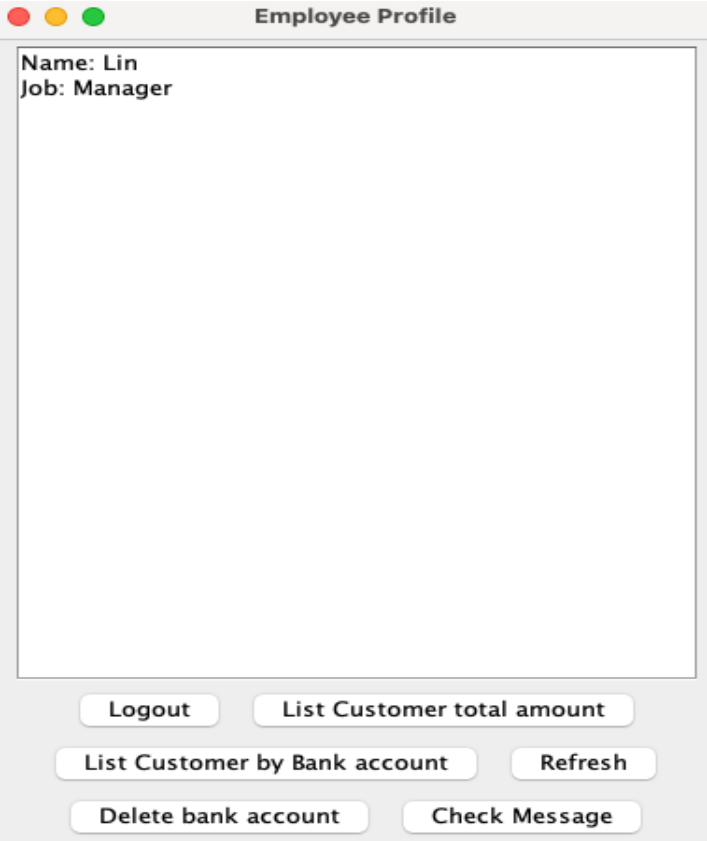
If user click create account at login page, this GUI will pop up to let you create customer account. Function “Check Username” can pull all customer username from database to compare with input username. If match, system will tell user to change username. User can’t create account until system tell user it is good username to use.

CustomerAccountGUI



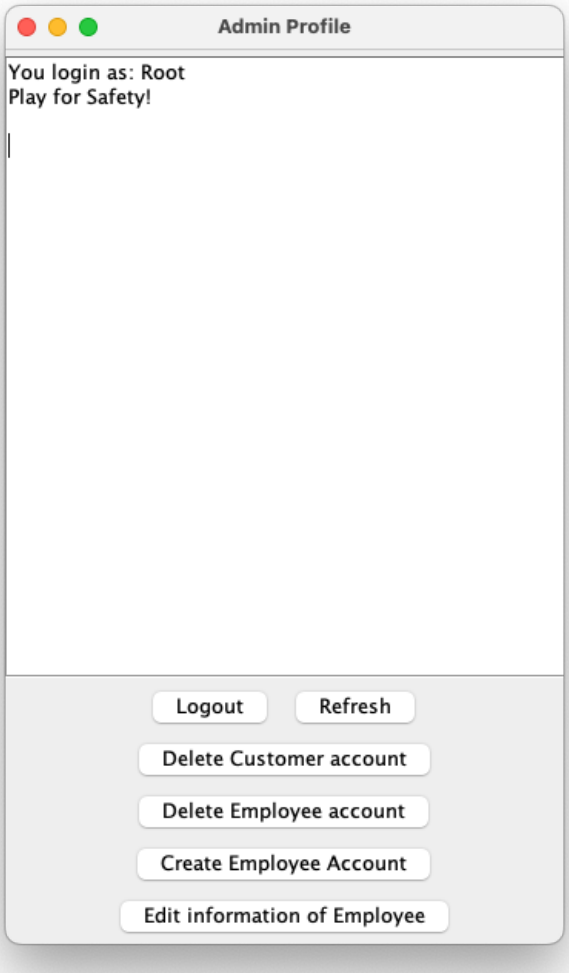
This is customer profile. System will display all bank account information on board. Customer can deposit or withdraw money. After operation, just click refresh button to refresh information on board. It can edit information to change password and address. It can also send message to employee to request opening new bank account.

EmployeeAccountGUI



Employee can list all customer information which include customerID, name, address, and total money. It can also list all bank account information which include bank accountID, owner, customerID, money, and address. Employee can delete any bank account. It can also check message that send by customer, and then approve the request to let customer to open new bank account.

AdminAccountGUI



Admin can delete customer or employee account. It can also create employee account and edit information of employee account.

Technical Framework

The Graphical User Interface is created by Javax Swing Framework. JFrame is used to create the main window for graphical user interface. Jbutton is used to create clickable buttons in GUI. JOptionPane provides a way to create standard dialog boxes. JTextArea is used to display content from database.

Database Management: MySQL
Admin account

- Account_ID
- Admin_name
- Usernmae
- Password

Employee account

- Account_ID
- Name
- Usernmae
- Password
- Job title

Message Box

- Message_ID
- Content
- Account_ID (FK to customer account)
- Employee_ID (FK to employee account)

Customer account

- Account_ID
- Customer_name
- Username
- Password
- Address
- Permission

Bank account

- Bank_account_ID
- Money
- Type
- Account_ID (FK) to customer account

Achievement

This project successfully integrates diverse functionalities for different type of user groups. It also provides each user type with an interface tailored to their needs and permissions. The use of Java Swing to create a graphical user interface makes the system accessible and intuitive for users, improving the overall user experience. With separate authentication pathways that ensures users access only their authorized areas.

Future Work

In the future, program should add encryption for password storage like hash algorithm. For the data transmission, it should be encrypted by TLS or SSL. For more user account security, additional layers of security such as multi-factor authentication should be added. Continued development on the GUI can make project more responsive and aesthetically pleasing. With increasing number of users, the system needs optimizations to handle high-volume transactions and maintain performance. Project can utilize scalable cloud database services, which implement efficient database management, and apply load balancing techniques.

Conclusion

Banking system provides comprehensive and user-friendly experience for customer, employees, and administrators. It emphasizes efficient management of banking operations and enhanced user interaction. The system’s adaptable architecture allows for future enhancement and integration.