# UNIVERSITY OF MUMBAI

A PROJECT REPORT ON

# “CashCraft: Mastering Your Financial Journey “

SUBMITTED BY

**Ms. Durga Adhikari Chhetri**

Under the guidance of

**Prof. Shreya Bhamare**

**Vishnu Waman Thakur Charitable Trust’s**

**VIVA INSTITUTE OF TECHNOLOGY**

**Shirgaon, Virar (East)**

**2023-24**

**Vishnu Waman Thakur Charitable Trust's**

# VIVA INSTITUTE OF TECHNOLOGY

**Shirgaon, Virar (East)**

## CERTIFICATE

**This is to certify that**

**Ms. Durga Adhikari Chhetri**

**Has satisfactorily completed the project entitled**

**CashCraft: Mastering Your Financial Journey**

**Towards the partial fulfillment of the**
**MASTER OF COMPUTER APPLICATION (MCA)**
**As laid by University of Mumbai.**

| | | |
|---|---|---|
| **Principal** | **External Examiner** | **Internal Guide** |

# ACKNOWLEDGMENT

We extend our heartfelt gratitude to everyone who contributed to the successful completion of the mini-project, "CashCraft: Mastering Your Financial Journey." This endeavor wouldn't have been possible without the collaborative efforts and support from various individuals and resources.

We express our sincere appreciation to Prof. Shreya Bhamare, whose guidance, mentorship, and valuable insights significantly enriched this project. Your continuous support and encouragement were instrumental in shaping the project's success.

We are immensely thankful to our peers and colleagues who offered their assistance and shared their expertise throughout the development process. Their input and constructive criticism were invaluable in refining the project.

Lastly, we acknowledge the support of our family and friends for their unwavering encouragement and understanding during the project's journey.

# ABSTRACT

The CashCraft project revolutionizes the realm of financial transactions by introducing a unique fintech application. This application empowers users to seamlessly transfer money without relying on traditional banking systems. The project targets financial inclusion by offering an accessible solution for individuals without conventional bank accounts.

CashCraft is a financial technology application designed to facilitate money transfers between users. It eliminates the dependency on traditional banking by allowing individuals to create accounts and perform transactions without the need for a bank account. Inspired by the ease of railway card systems, the project introduces a hybrid model that combines digital convenience with the familiarity of physical deposit locations.

Key features of the application include a simple account creation process, user-friendly interfaces, transaction tracking, and the ability to deposit funds at physical organization centers. Users can easily transfer money to others using the same application, simplifying the process and enhancing accessibility to financial services.

Future enhancements aim to introduce admin controls for security, advanced security measures, expanded deposit options, and additional financial services. These improvements align with the project's vision to constantly evolve, ensuring user-centricity, security, and reliability.

The project is built upon Agile methodologies, fostering flexibility, continuous improvement, and a customer-centric approach. The iterative development process allows for quick adaptations to evolving user needs, ultimately aiming to redefine accessibility and simplicity in digital financial transactions.

CashCraft presents a user-centric solution that holds promise in offering financial accessibility to a wider audience. Its innovative approach, combined with a commitment to ongoing enhancements, positions the project at the forefront of modern financial technology.

# INDEX

# INTRODUCTION

## 1.1 Introduction

In an era of advancing financial technologies, the CashCraft project emerges as a pioneering solution, aiming to redefine the landscape of digital financial transactions. The project addresses the evolving needs of a rapidly changing society by introducing a revolutionary fintech application. CashCraft, titled "Mastering Your Financial Journey," stands at the forefront of innovation, offering a novel approach to monetary exchanges and accessibility in the financial domain.

CashCraft is driven by a vision to revolutionize the way individuals interact with and manage their finances. Traditional banking systems often present barriers to access for individuals without a bank account or those navigating cumbersome processes. Understanding these limitations, CashCraft presents an alternative—a user-centric application that facilitates seamless money transfers without the necessity of conventional banking structures.

At its core, CashCraft aims to democratize financial transactions by offering a simple, intuitive, and accessible platform for users to engage in money transfers. By harnessing the potential of financial technology, CashCraft enables individuals to create accounts easily and transfer money to others within the application. Inspired by familiar systems like the railway card, CashCraft introduces a hybrid model that amalgamates digital convenience with physical deposit locations, catering to a wider audience and enhancing accessibility.

This introduction serves as a gateway to a project that not only challenges the status quo but also envisions a future where financial inclusion is not confined by traditional banking norms. CashCraft endeavors to usher in a new era—one where financial transaction is not just efficient and secure but also accessible to all, irrespective of their banking affiliations.

### 1.1.1 Problem Definition

1. **Limited Banking Access**: A significant portion of the population lacks access to traditional banking services due to various reasons such as geographic constraints, lack of necessary documentation, or unavailability of nearby bank branches.

2. **Complexity in Money Transfers**: Existing banking systems often involve complex procedures for money transfers, which might be intimidating for users unfamiliar with banking processes.

3. **Exclusion of Unbanked Individuals**: A large number of individuals, especially in remote or underprivileged areas, do not possess traditional bank accounts. Consequently, they are excluded from the formal financial system, making it challenging for them to engage in digital transactions.

4. **Transaction Hurdles**: The conventional banking system sometimes presents obstacles in performing simple transactions, leading to inconvenience and delays for users.

5. **Limited Accessibility to Financial Services**: Lack of accessibility to banking facilities restricts the ability of individuals to engage in financial activities such as money transfers, checking balances, or accessing transaction histories.

### 1.1.2 Objectives

1. **Accessibility**: Addressing the exclusion of individuals without traditional bank accounts by providing a user-friendly alternative for financial transactions.

2. **Simplicity**: Simplifying the process of money transfers and account management to enhance user convenience.

3. **Innovation**: Introducing a hybrid model inspired by familiar systems to amalgamate digital convenience with physical deposit locations.

4. **Financial Inclusion**: Democratizing financial transactions by making them accessible to a wider audience, irrespective of their banking affiliations.

5. **Security**: Ensuring robust security measures to safeguard user information and transactions.

### 1.1.3 Scope of the Project

1. **Account Creation**: Users can create accounts by providing basic details, eliminating the need for a traditional bank account.

2. **Money Transfer**: Facilitating seamless money transfers between users within the application.

3. **Balance and Transaction History**: Enabling users to monitor account balance and transaction history.

4. **Deposit and Withdrawal**: Initial testing includes cash deposits at physical organization centers, with withdrawals available post-admin login.

5. **Enhancements**: The project seeks to explore admin controls, enhanced security features, additional deposit methods, and potential expansion of financial services.

## 1.2 Technical Details

### 1.2.1 Overview of Front End

The front-end of application primarily deals with the user interface and user interactions. It involves a Java-based graphical user interface (GUI) developed using Swing or AWT.

1. **Graphical User Interface (GUI)**: Developed using Java Swing or AWT. Contains various components like text fields, labels, buttons, and password fields for user input and interaction. Provides an intuitive layout for users to interact with your application.

2. **Styling and Presentation**: Includes setting fonts, colors, backgrounds, and layouts for different components. Possibly involves using image icons or backgrounds to enhance the visual appeal.

### 1.2.2 Overview of Back End

The backend of application deals with data storage, retrieval, and processing. It involves interaction with the MySQL database using Java Database Connectivity (JDBC).

1. **Database Management System**: Relational database managed using MySQL Workbench. Contains tables and schema necessary to store and manage user information, passwords, and other relevant data.

2. **Java Database Connectivity (JDBC)**: Handles connections between the Java application and the MySQL database. Executes SQL queries and manages data retrieval, insertion, deletion, and updates in the database.

3. **Business Logic and Data Processing**: Includes logic for user authentication, password updates, and potentially other business rules related to financial transactions.

# SYSTEM STUDY AND PLANNING

## 2.1 System Study

### 2.1.1 Existing System

The existing system typically revolves around traditional banking structures or established digital payment platforms that necessitate conventional bank accounts for financial transactions. It might encompass:

1. **Traditional Banking Infrastructure**:
   - Bank-Centric Transactions: Users rely on traditional banks for financial activities.
   - Account Dependency: Transactions often require users to possess bank accounts.

2. **Digital Payment Platforms**:
   - Bank Account Dependency: Most digital platforms require users to link their bank accounts for transactions.
   - Limited Accessibility: Accessibility is constrained for individuals without bank accounts.

### 2.1.2 Disadvantages of Existing System

1. **Exclusion of Unbanked Individuals**:
   - Financial Inaccessibility: Individuals without bank accounts are excluded from mainstream financial services.
   - Accessibility Barrier: Accessing digital payment platforms or conducting online transactions is challenging.

2. **Complex Procedures**:
   - Lengthy Processes: Transactions often involve extensive paperwork or lengthy procedural steps.
   - Dependence on Banks: Heavy reliance on traditional banking systems creates bureaucratic hurdles.

3. **Geographical Limitations**:
   - Geographical Constraints: Limited access to banking services in certain areas or regions.

- Physical Presence: Users might need to physically visit banks or specific centers for transactions.

4. **Limited Innovation and Flexibility**:
   - Innovative Constraints: Existing systems may lack adaptability to evolving technologies or user needs.
   - Stagnation in Services: Limited improvements or enhancements over time hinder service advancements.

### 2.1.3 Proposed System

The proposed system, CashCraft, aims to revolutionize financial transactions by offering an alternative way for users to transfer money without traditional bank dependencies. Key features include:

1. **Bankless Transactions**:
   - Allows users to create accounts and transfer money without requiring traditional bank accounts.
   - Simplifies the process, making it more accessible to a wider audience.

2. **Accessibility and Convenience**:
   - Facilitates depositing money at physical organization centers, enhancing accessibility.
   - Provides a straightforward process for money transfers, creating a user-friendly experience.

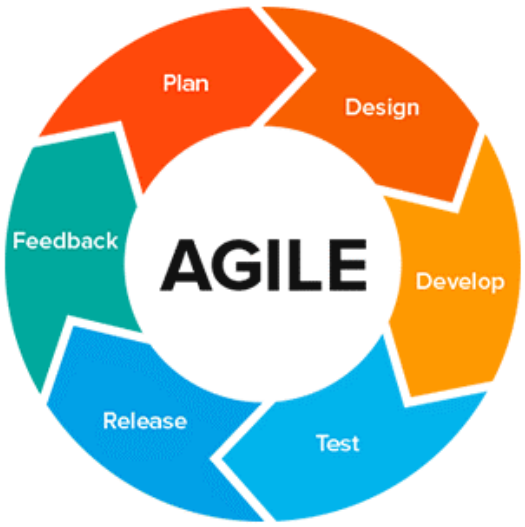
## 2.2 System Planning and Schedule

### 2.2.1 S/W development Model

**Agile Development Model:**

The meaning of Agile is swift or versatile. "Agile process model" refers to a software development approach based on iterative development. Agile methods break tasks into smaller iterations, or parts do not directly involve long term planning. The project scope and requirements are laid down at the beginning of the development process. Plans regarding the number of iterations, the duration and the scope of each iteration are clearly defined in advance.

The advantages of this model are as follows –




1. **Iterative Development**:
   - Divides the project into small, manageable increments or sprints.
   - Each sprint delivers a functional piece of software, allowing continuous delivery.

2. **Flexibility and Adaptability**:
   - Embraces changes in requirements throughout the development process.
   - Accommodates evolving needs and priorities through continuous iterations.

3. **Customer-Centric Approach**:
   - Prioritizes delivering value to users through incremental releases.
   - Incorporates feedback from stakeholders and customers to refine the product.

4. **Collaboration and Communication**:
   - Encourages cross-functional teamwork and communication among team members.
   - Regular meetings ensure alignment and awareness of progress and challenges.

The Agile model suits the CashCraft project by enabling adaptability to changes in financial service requirements, focusing on user needs, and facilitating continuous improvements based on feedback.

# SYSTEM DESIGN

## 3.1 Software Requirement Specification (SRS)

### 3.1.1 Introduction to SRS

CashCraft is a bankless transaction platform allowing users to create accounts, transfer money, and manage transactions seamlessly.

1. **Functional Requirements**:
   - User authentication for secure logins.
   - Account management to view balances and transactions.
   - Secure money transfer between registered users.
   - Provision for physical cash deposits and admin-enabled withdrawals.
2. **Non-functional Requirements**:
   - Responsive performance handling multiple requests.
   - Reliable system uptime and regular data backups.
3. **System Features**:
   - Account management module for balance and transaction views.
   - Secure transfer module for fund transactions.
4. **Constraints**:
   - Initial deposits limited to physical centers.
   - Deposit and Withdrawals accessible post-admin login.

### 3.1.2 Technological Requirements

#### 3.1.2.1 Hardware to be Used

1. **Computing Devices**:
   - Desktops or Laptops: To develop, run, and test the application.
2. **Memory and Storage**:
   - Adequate RAM and storage based on the application's requirements.

#### 3.1.2.2 Software/tools to be Used

1. **Programming Language**:
   - Java: Used for developing the backend and GUI components.
2. **IDE (Integrated Development Environment)**:
   - Eclipse: Used for Java development, providing an environment for coding, debugging, and testing.

3. **Database Management System**:
    - MySQL: Database system used to store and manage application data. MySQL Workbench as the graphical tool to manage the MySQL database.
4. **Java Libraries/Frameworks**:
    - AWT and Swing: Utilized for GUI development in Java.

## 3.2 Detailed life Cycle of the Project

### 3.2.1 Modules

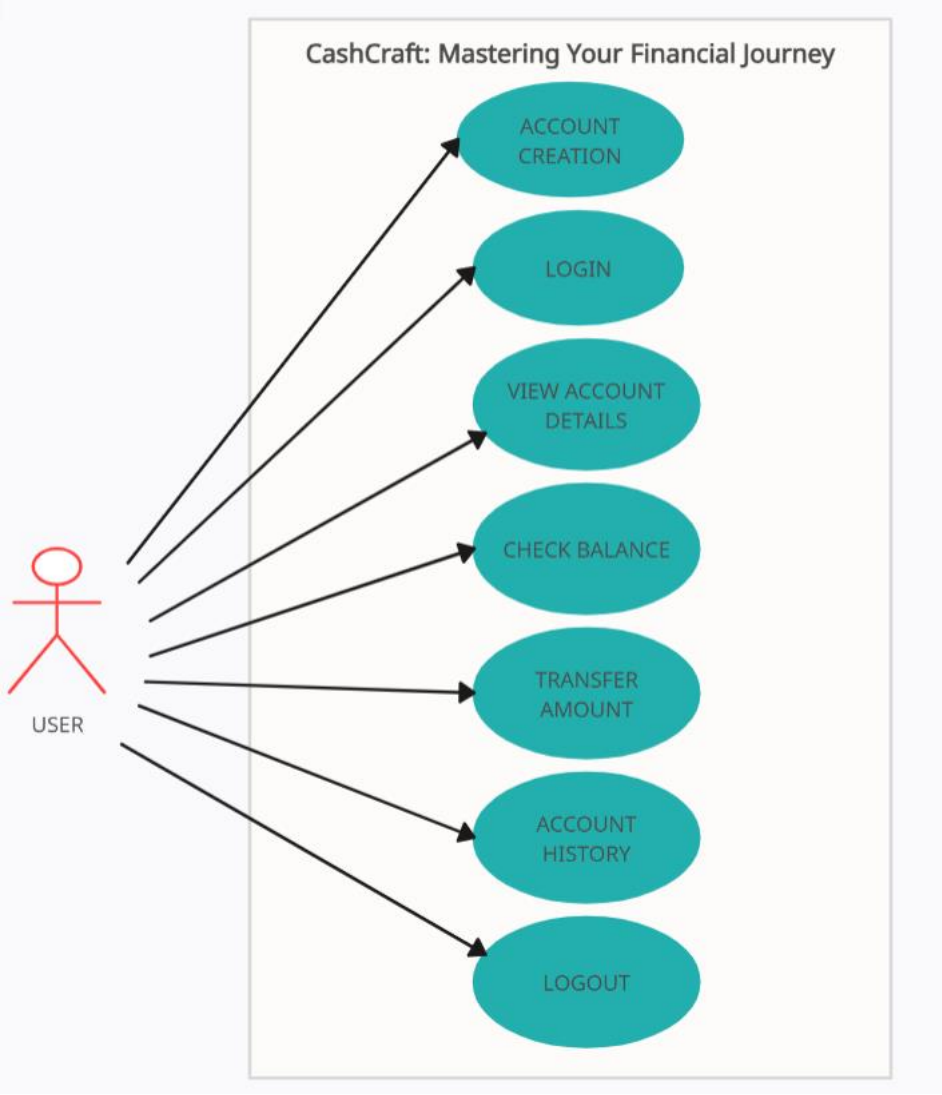
1. **Account Creation Module**:
    - Functionality: Allows users to register by providing basic details.
    - Features: Input fields for user information (name, email, dob, password). Generates a unique PIN for user authentication upon successful registration. Basic validation and storage of user data in the database.
2. **Login Module**:
    - Functionality: Validates user credentials for authentication.
    - Features: Verification of user PIN and email to access account features. Restricts access without proper authentication.
3. **Transaction Module**:
    - Functionality: Manages different transaction-related actions.
    - Features: Buttons to navigate to other key modules.
4. **Account Details Module**:
    - Functionality: Presents user account-related information.
    - Features: User profile display with stored details. Options to edit or delete profile information.
5. **Check Balance Module**:
    - Functionality: Allows users to view their account balance.
    - Features: Real-time retrieval and display of account balance information.
6. **Deposit Module**:
    - Functionality: Placeholder for future integration with physical centers.
    - Features: To be utilized for depositing money in-person at specific locations (after admin login implementation).
7. **Withdraw Module**:
    - Functionality: Placeholder for future integration with admin controls.
    - Features: Reserved for controlled and secure money withdrawals (after admin login

implementation).

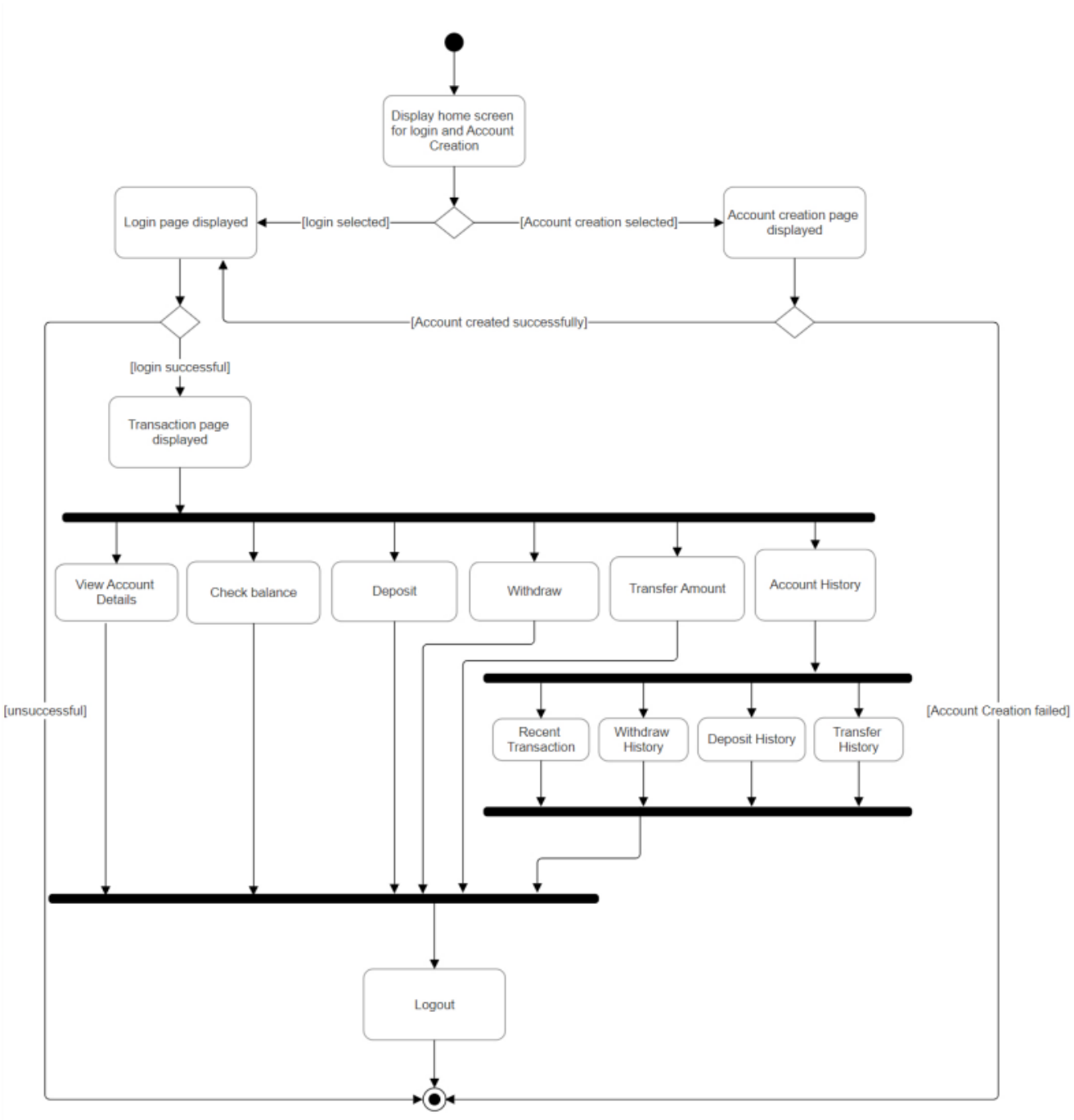
8. **Transfer Amount Module**:
   - Functionality: Enables seamless fund transfers between users.
   - Features: Facilitates secure and verified money transactions between registered users.
9. **Account History Module**:
   - Functionality: Displays the transaction history for user reference.
   - Features: Options to view Recent transactions, Deposit history, Withdrawal history, Transfer history.

### 3.2.2 Object Oriented Analysis & Design Diagram

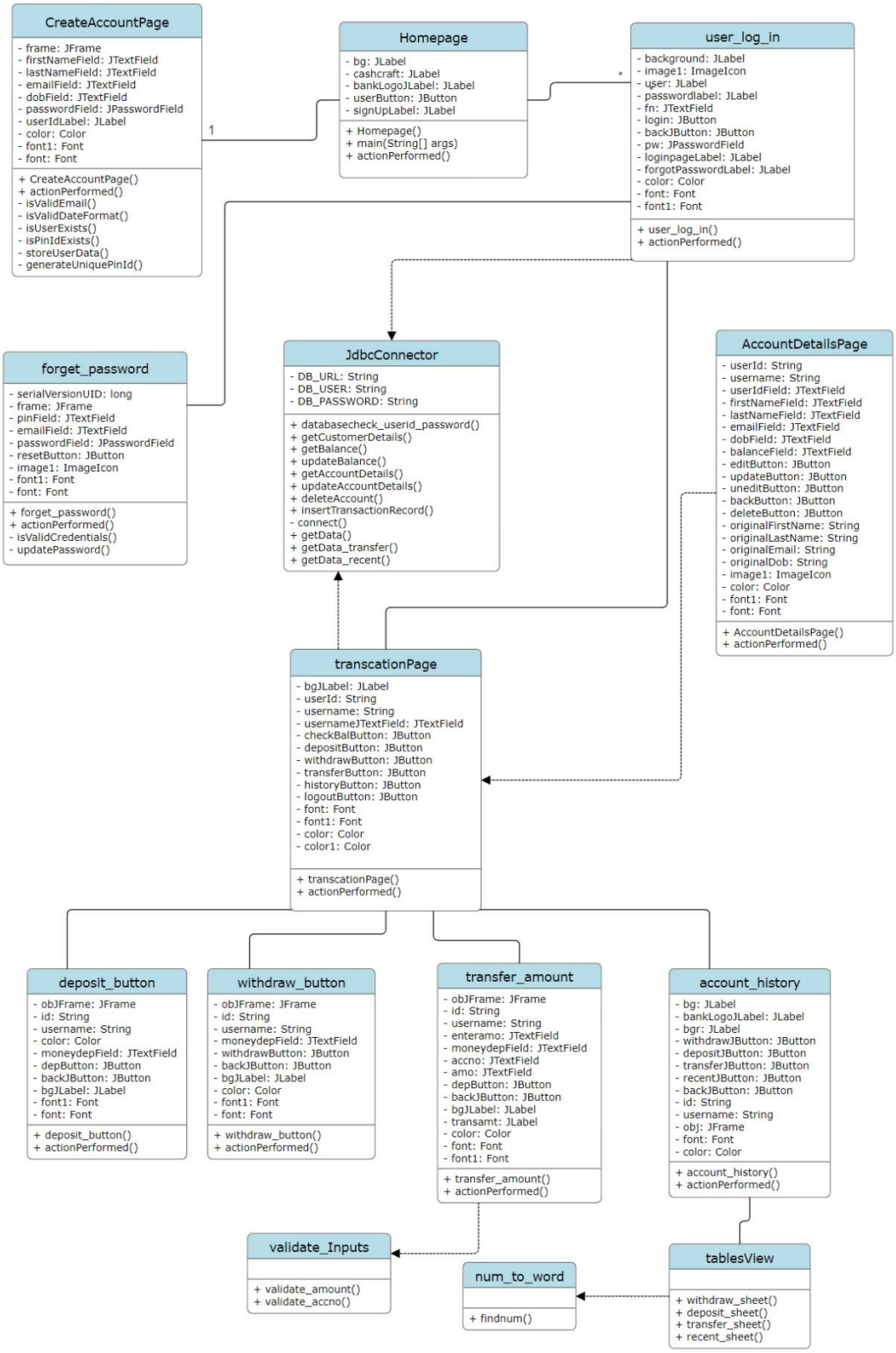
#### 3.2.2.1 Use Case Diagram



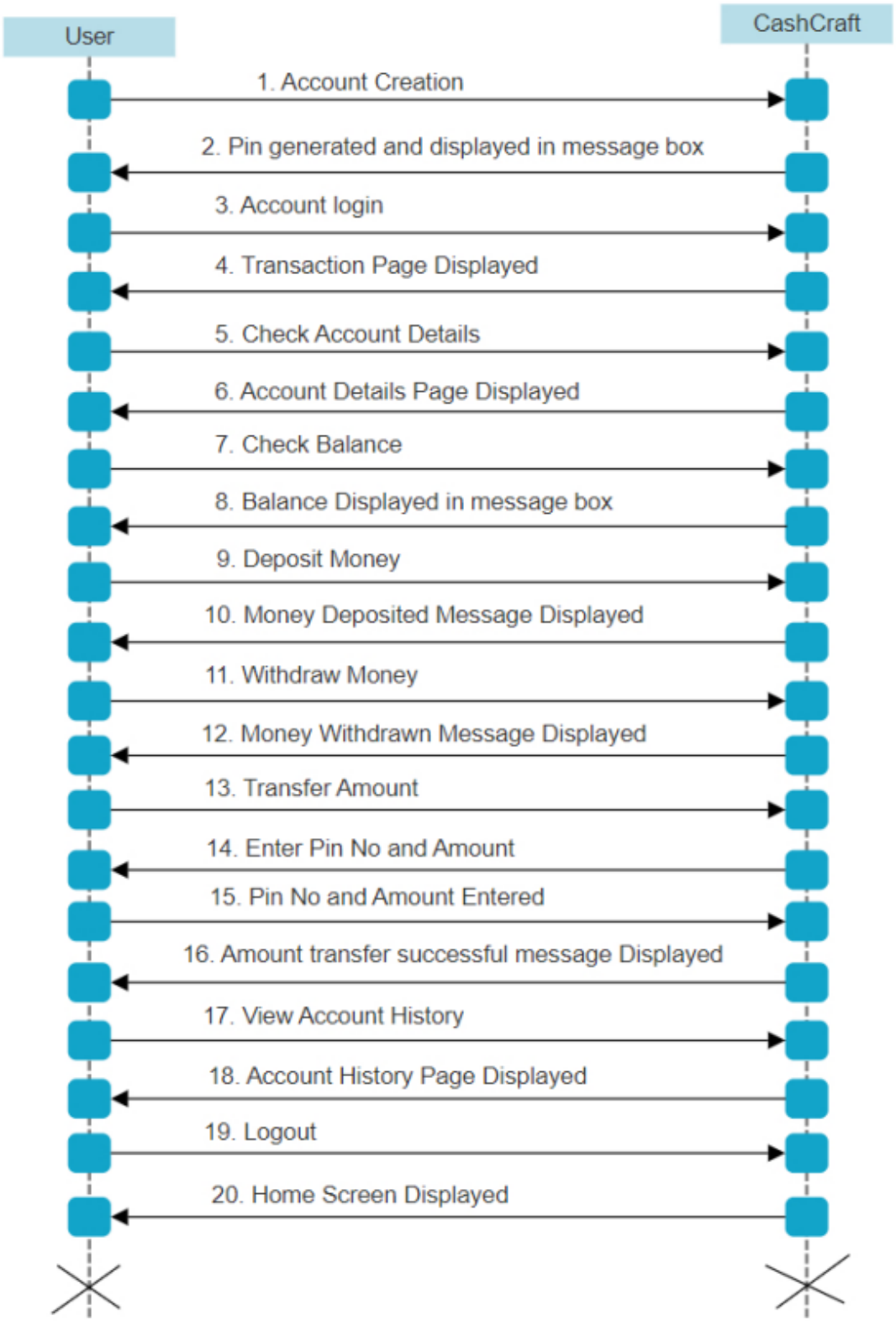
### 3.2.2.2 Activity Diagram



## 3.2.2.3 Class Diagram

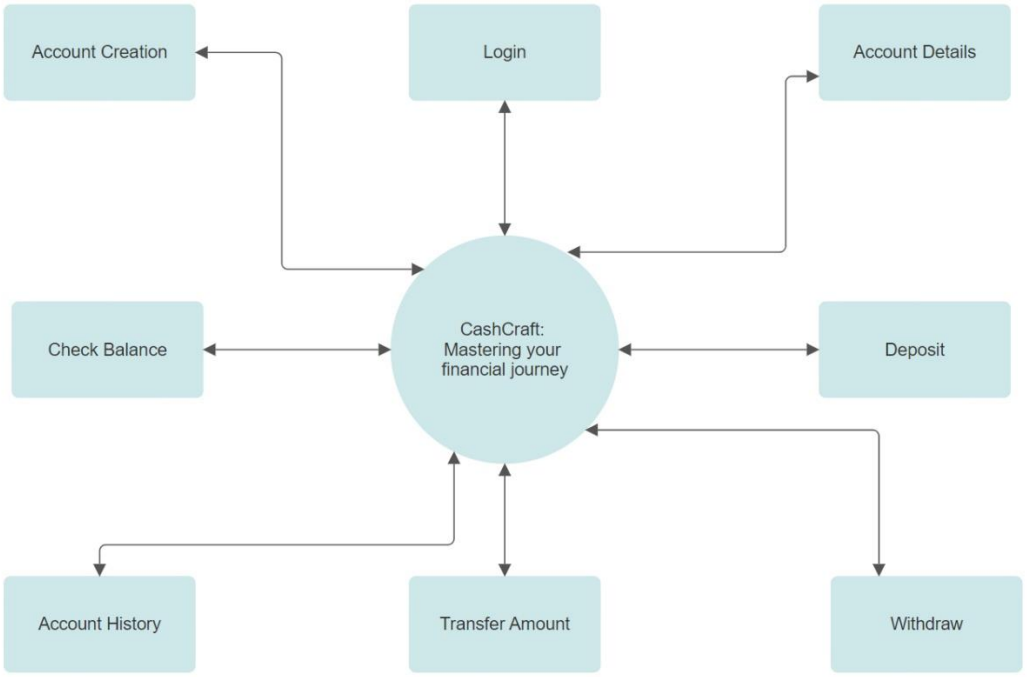


### 3.2.2.4 Sequence Diagram



### 3.2.2.5 Data Flow Diagram

#### Zero Level DFD




## 3.2.3 Database

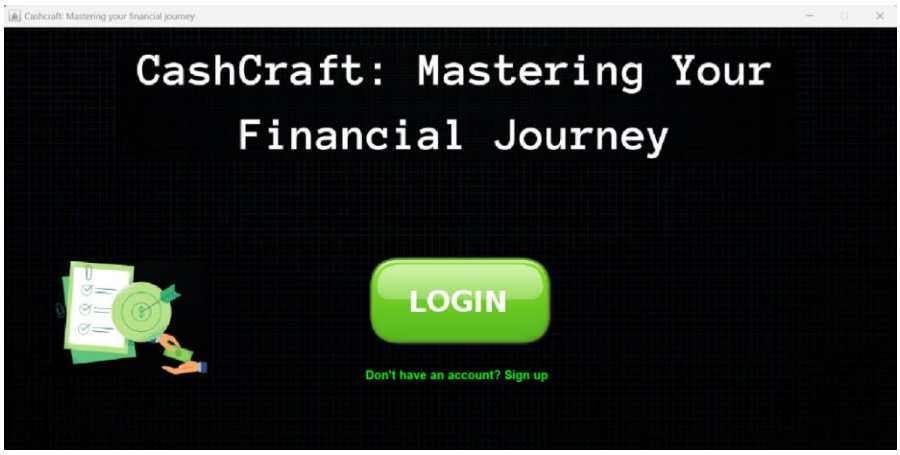
### 3.2.3.1 Database Table

| account_details | |
|---|---|
| user_id | INT |
| first_name | VARCHAR |
| last_name | VARCHAR |
| email | VARCHAR |
| dob | DATE |
| password | VARCHAR |
| pin_id | INT |
| balance | INT |
| created_at | TIMESTAMP |

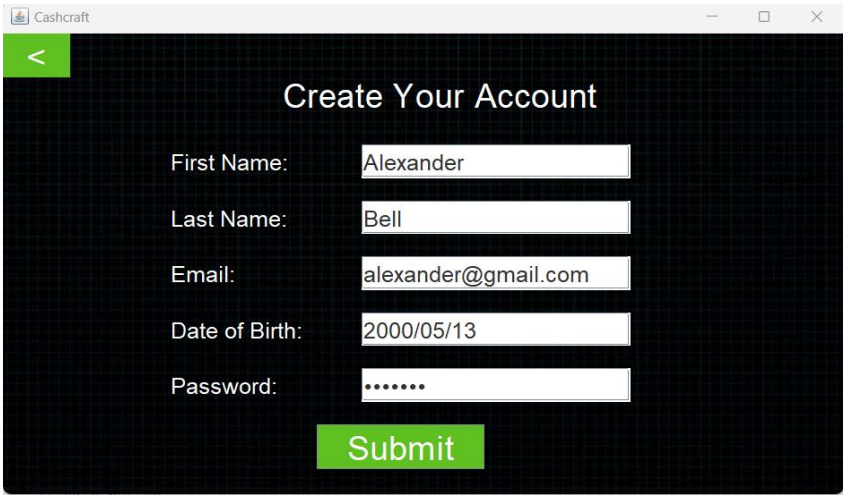
| Transactions | |
|---|---|
| NO | INT |
| ACCNOSENDER | INT |
| TYPETRANSCATION | VARCHAR |
| AMOUNT | INT |
| ACCNORECEIVER | INT |
| TDATE | DATE |
| TTIME | TIME |
| TDT | DATETIME |

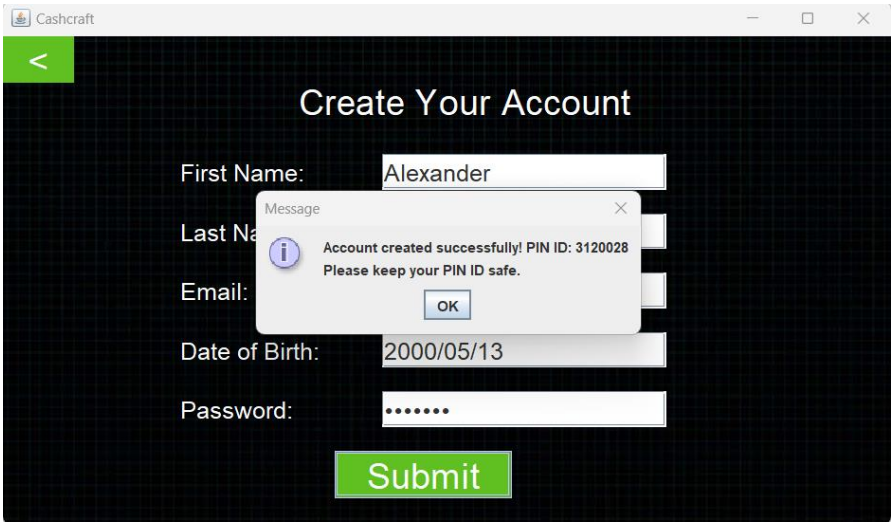
### 3.2.4 I/O Screen Layout

**Homepage:** It's the main entry point of the application. This is the first page that displays when the application is opened/started.

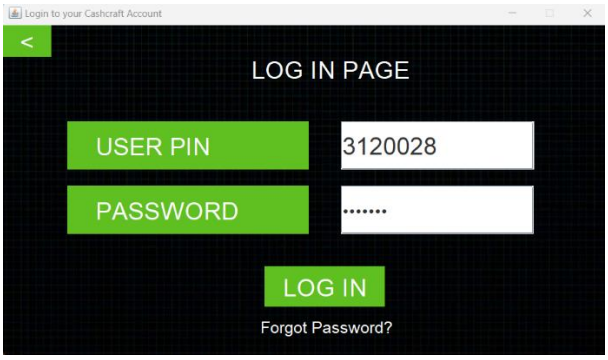


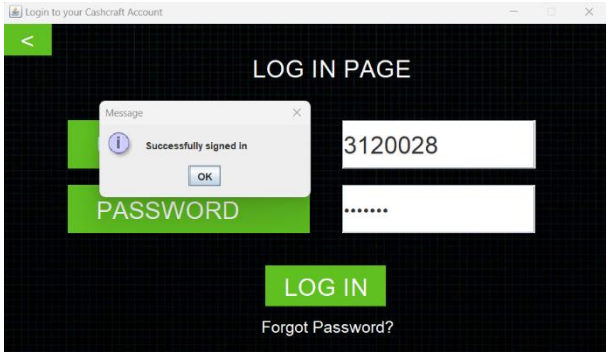

**Account Creation:** User needs to create an account for the use of the application.



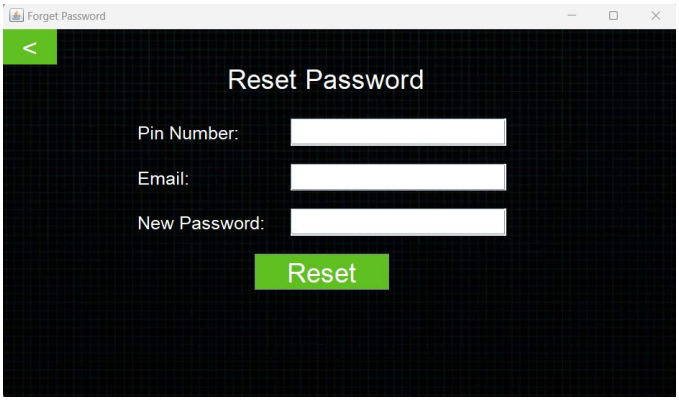



**Login Page:** Once the account is created user needs to login to the account with generated pin and password for authentication.

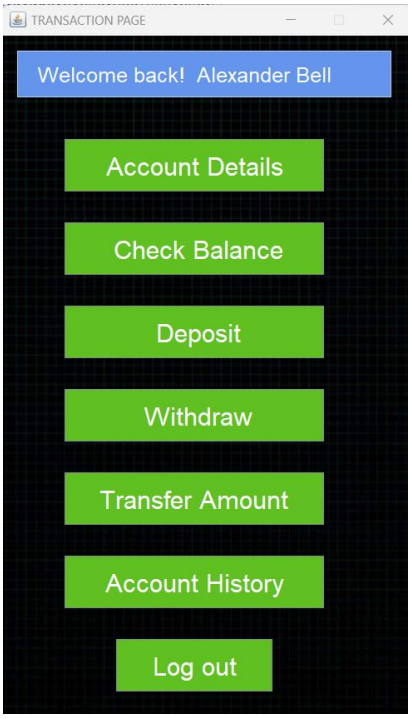


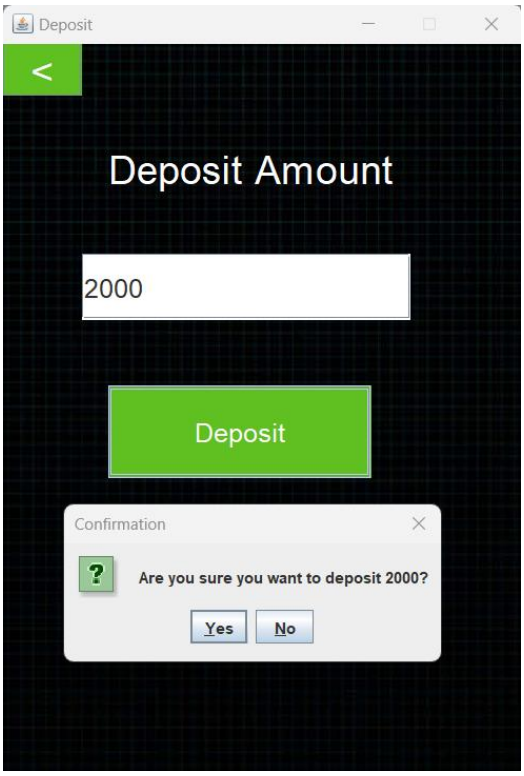

**Reset Password Page:** User can reset the password if they forget.

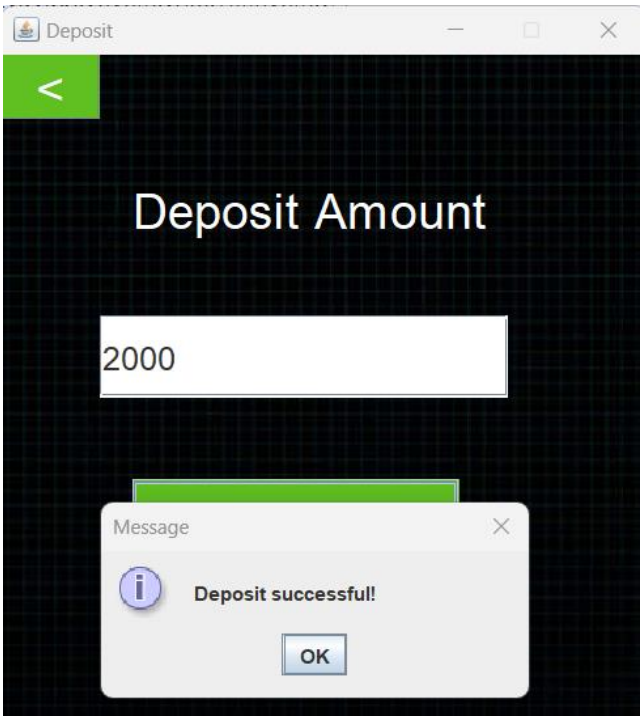



**Transaction Page:** This is the main page from where user can perform all activities listed in it.

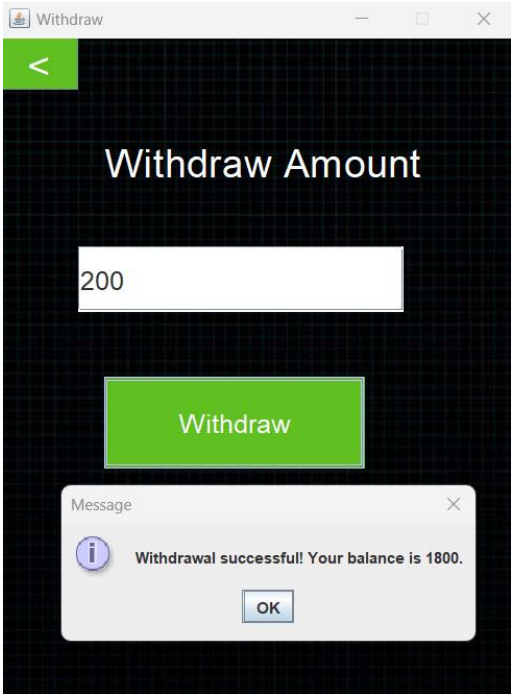


**Deposit Page:** The amount is deposited in user account. It's a Placeholder for future integration with physical centers.

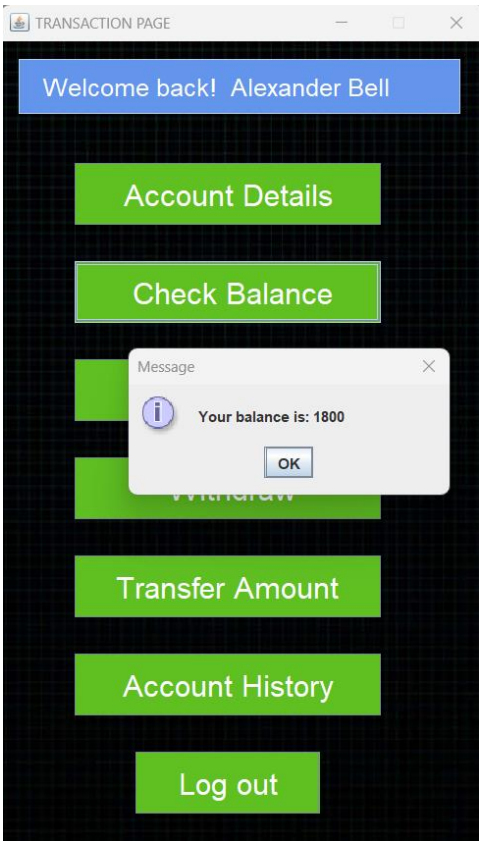


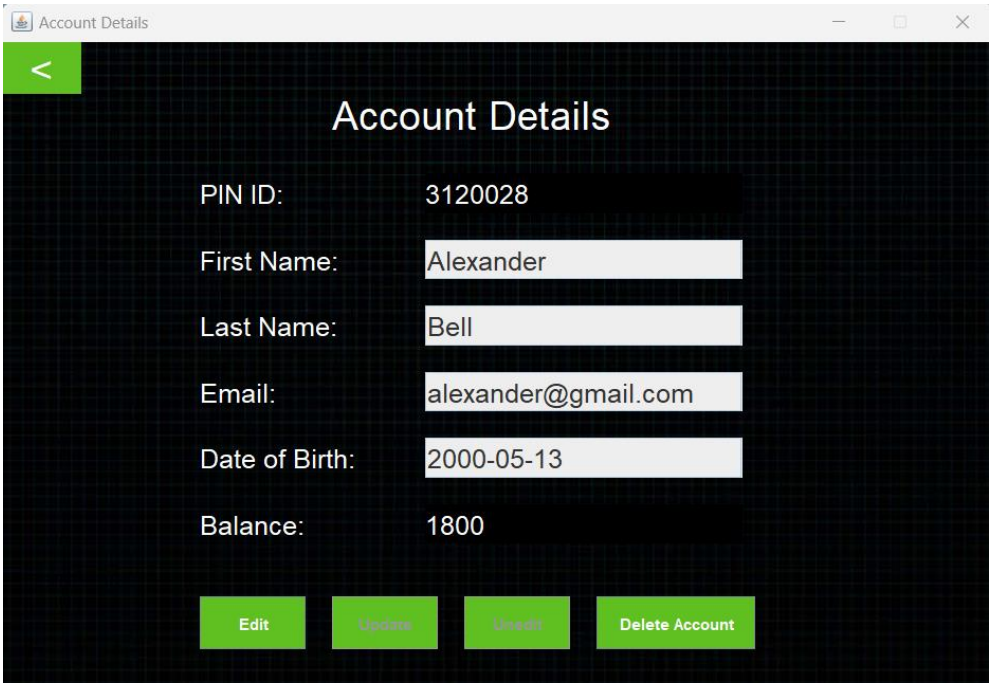

**Withdraw Page:** The amount is withdrawn from user's account. It's a Placeholder for future integration with physical centers.



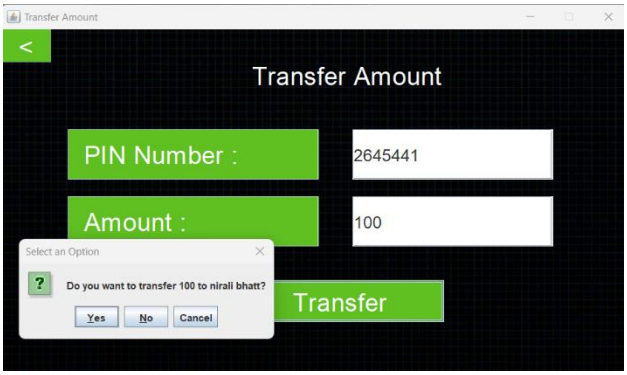
**Check Balance:** The user can check its account balance.

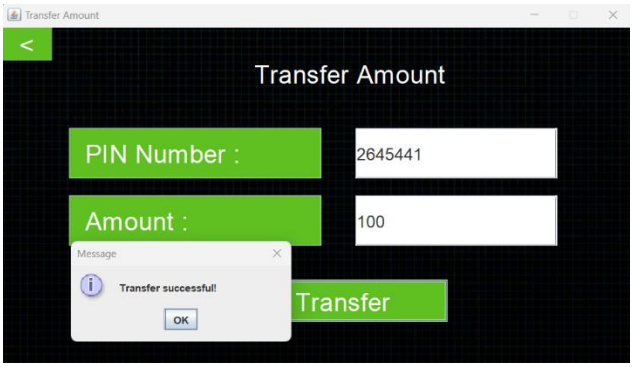



**Account Details:** Presents user account-related information. User can edit the information or delete account.

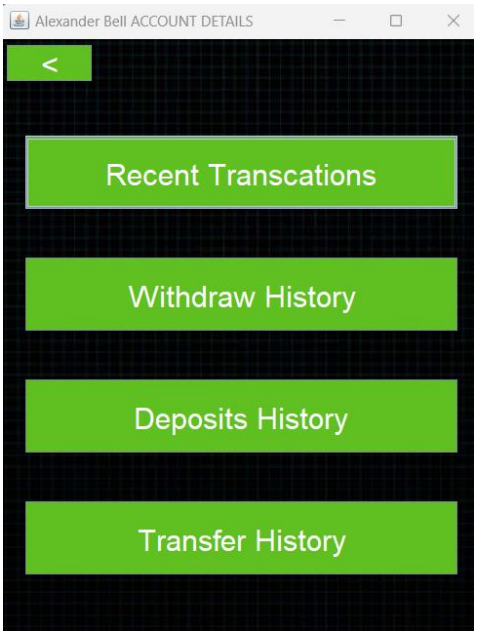


**Transfer Amount:** User can transfer amount to another user using same application.




**Account History:** User can view its transaction history.




RECENT HISTORY

| Transcation | Receivers Account No. | Amount | Amount In Words | DATE & TIME |
|---|---|---|---|---|
| Transfer | 2645441 | 100/- | one hundred rupees only/- | 2023-11-24 19:16:38.0 |
| Withdraw | 3120028 | 200/- | two hundred rupees only/- | 2023-11-24 19:06:19.0 |
| Deposit | 3120028 | 2000/- | two thousand rupees only/- | 2023-11-24 19:02:25.0 |

WITHDRAW HISTORY

| Amount | Amount In Words | DATE | TIME |
|---|---|---|---|
| 200/- | two hundred rupees only/- | 2023-11-24 | 19:06:19 |

DEPOSIT HISTORY

| Amount | Amount In Words | DATE | TIME |
|---|---|---|---|
| 2000/- | two thousand rupees only/- | 2023-11-24 | 19:02:25 |

TRANSFER HISTORY

| Receivers Account No. | Amount | Amount In Words | DATE & TIME |
|---|---|---|---|
| 2645441 | 100/- | one hundred rupees only/- | 2023-11-24 19:16:38.0 |

# TESTING

The project uses Agile model, the development process is divided into small increments called sprints. Each sprint typically lasts from one to four weeks, during which a small part of the overall project is completed.
Here are the key aspects of the Agile model:

1. **Iterative and Incremental Development**:
   - Sprints: Work is divided into time-bound iterations or sprints.
   - Incremental Deliveries: Functional pieces of the software are developed incrementally, allowing for continuous delivery of usable software.
2. **Flexibility and Adaptability**:
   - Adaptive to Change: Agile allows for changes in requirements, accommodating evolving needs and priorities.
   - Regular Feedback: Continuous feedback from stakeholders and customers is integrated into the development process.
3. **Collaboration and Communication**:
   - Cross-functional Teams: Developers, testers, designers, and stakeholders work collaboratively within a team.
   - Frequent Communication: Regular meetings like daily stand-ups ensure everyone is aligned and aware of progress and challenges.
4. **Emphasis on Customer Satisfaction**:
   - Customer-Centric Approach: Prioritizes delivering value to the customer through iterative releases.
   - Quick Response to Feedback: Iterative development allows for quick incorporation of feedback.
5. **Continuous Improvement**:
   - Retrospectives: Regularly held meetings to reflect on what went well, what didn't, and how to improve processes.

## 4.1 Methodologies used for testing

1. **Unit Testing**:
   - Explanation: Testing individual components or modules in isolation. In your banking application, you could test individual functions like validating PINs, updating passwords, or database interactions independently.

- Benefits: Helps in identifying bugs early, isolates issues in specific code segments, and facilitates easier debugging.

2. **Integration Testing**:
   - Explanation: Verifying interactions between different modules or systems within your application. For instance, testing how your UI interacts with your backend logic or how different modules communicate.
   - Benefits: Ensures that various components work together seamlessly, identifies interface defects, and validates the flow of data between modules.
3. **System Testing**:
   - Explanation: Evaluating the entire system against the specified requirements. It involves testing the complete application as a whole to ensure it meets the business needs.
   - Benefits: Validates end-to-end functionality, assesses whether the system meets user expectations, and identifies any inconsistencies.
4. **Regression Testing**:
   - Explanation: Re-running previously executed tests to ensure that new changes haven't adversely affected existing functionalities. For instance, after adding the forgot password feature, ensuring it didn't break the login functionality.
   - Benefits: Prevents unintended side effects from new changes and ensures that the system remains stable after modifications.

## 4.2 Types of Testing

1. **Black-box Testing**: This approach examines the software's functionality without considering its internal structure or code implementation. Testers verify that the software performs as expected by using inputs and observing outputs. It's beneficial for testing user interfaces, functionalities, and user experience. For instance, in your banking application, black-box testing can validate features like login, transactions, and account management without looking at the underlying code.

2. **White-box Testing**: This type of testing delves into the internal structure of the software. Testers examine the code, logic, and internal paths to ensure they work correctly. In your banking application, white-box testing might involve verifying database interactions, error handling mechanisms, and security protocols.

# CONCLUSION

This project is a financial technology (fintech) application that allowed users to transfer money without the need for a traditional bank account. Users can create an account by providing basic details and then transfer money to others who also use the same application. The application allows users to check their balance, view transaction history, and perform money transfers seamlessly.

To deposit money into the application, users visit a physical organization center where they can provide cash. The withdrawal feature is currently for testing purposes and will be available post-admin login. The project draws inspiration from the railway card system, where users can deposit money at a physical location, making it accessible to individuals who may not have a traditional bank account.

The application aims to be user-friendly, providing a simple and intuitive interface for users to engage in financial transactions. Future enhancements include implementing an admin login system for additional security and control, exploring various deposit options beyond physical centers, and potentially expanding the range of financial services offered by the application. Overall, your project addresses the need for a straightforward and accessible way for users to transfer money without relying on traditional banking channels.

# FUTURE ENHANCEMENTS

1. **Admin Login and Controls**:
   - Implement an admin login system for enhanced security and control over deposit and withdrawal functionalities.
   - Admins can manage user accounts, oversee transactions, and ensure the security of the system.
2. **Security Measures**:
   - Integrate advanced security features such as two-factor authentication, biometric verification, or encryption to enhance the overall security of the application.
3. **Expanded Deposit Options**:
   - Explore additional deposit options beyond physical organization centers, such as partnerships with retail outlets or digital methods.
4. **Financial Services Expansion**:
   - Consider expanding the application to offer additional financial services like bill payments, mobile recharges, or even microloans in the future.
5. **User Education and Support**:
   - Develop educational materials within the app to guide users on financial literacy and responsible money management.
   - Implement a robust customer support system for query resolution and assistance.

# REFERENCES


- https://www.railmitra.com/blog/indian-railways-smart-card
- https://www.javatpoint.com/banking-application-in-java
- https://coderspacket.com/banking-management-system-using-java-swing-and-mysql
- https://1000projects.org/online-banking-application-project.html
- https://www.geeksforgeeks.org/mini-banking-application-in-java/
- https://copyassignment.com/bank-management-system-project-in-java/
- https://codezips.com/java/bank-management-system-in-java-with-source-code/