K.RAMAKRISHNAN COLLEGE OF TECHNOLOGY (AUTONOMOUS), TRICHY.

SIMPLE BANKING APPLICATION

PRESENTED BY
GANGAISELVAN.C
2303811710421302

SUPERVISOR
Mrs.Valli priyadharshini
M.E,(Phd),
AP/CSE

PRESENTATION OVERVIEW

1. Objective
2. Project Introduction
3. Problem Statement
4. Methodologies (Programming concepts relevant to problem statement)
5. Architecture of the proposed system
6. List of Modules
7. Merits
8. Results and Discussion
9. Queries

OBJECTIVE

The primary objective of a simple banking application is to provide users with a seamless and secure platform to manage their basic financial activities. The application should enable users to create accounts, log in securely, and perfor fundamental banking operations such as viewing account balances, transferring funds, and monitoring transaction history. A user-friendly interface is essential to ensure accessibility for individuals with varying levels of technical expertise. Additionally, the application must prioritize security features like encryption and multi-factor authentication to safeguard user data and transactions.

PROJECT INTRODUCTION

The Simple Banking Application is designed to replicate the fundamental operations of a bank in a digital environment. It allows users to create and manage their accounts, conduct financial transactions, and maintain records seamlessly. The application is intended for small-scale banks or educational purposes to understand the fundamentals of banking software development.

PROBLEM STATEMENT

With the increasing digitization of services, small banks or educational institutions need a lightweight and simple banking application to manage their operations effectively.

Many existing solutions are complex and costly, making them inaccessible to smaller entities.

The challenge is to develop a cost-effective, user-friendly, and secure banking application to address this need.

METHODOLOGIES

1. Object-Oriented Programming (OOP):

   For structuring the application into classes such as Account, Transaction, etc.

2. Data Structures:

   Arrays, lists, or dictionaries to store account details and transaction records.

3. File Handling or Database Management:

   To persist account and transaction data securely.

4. Input Validation:

To ensure valid inputs during transactions.

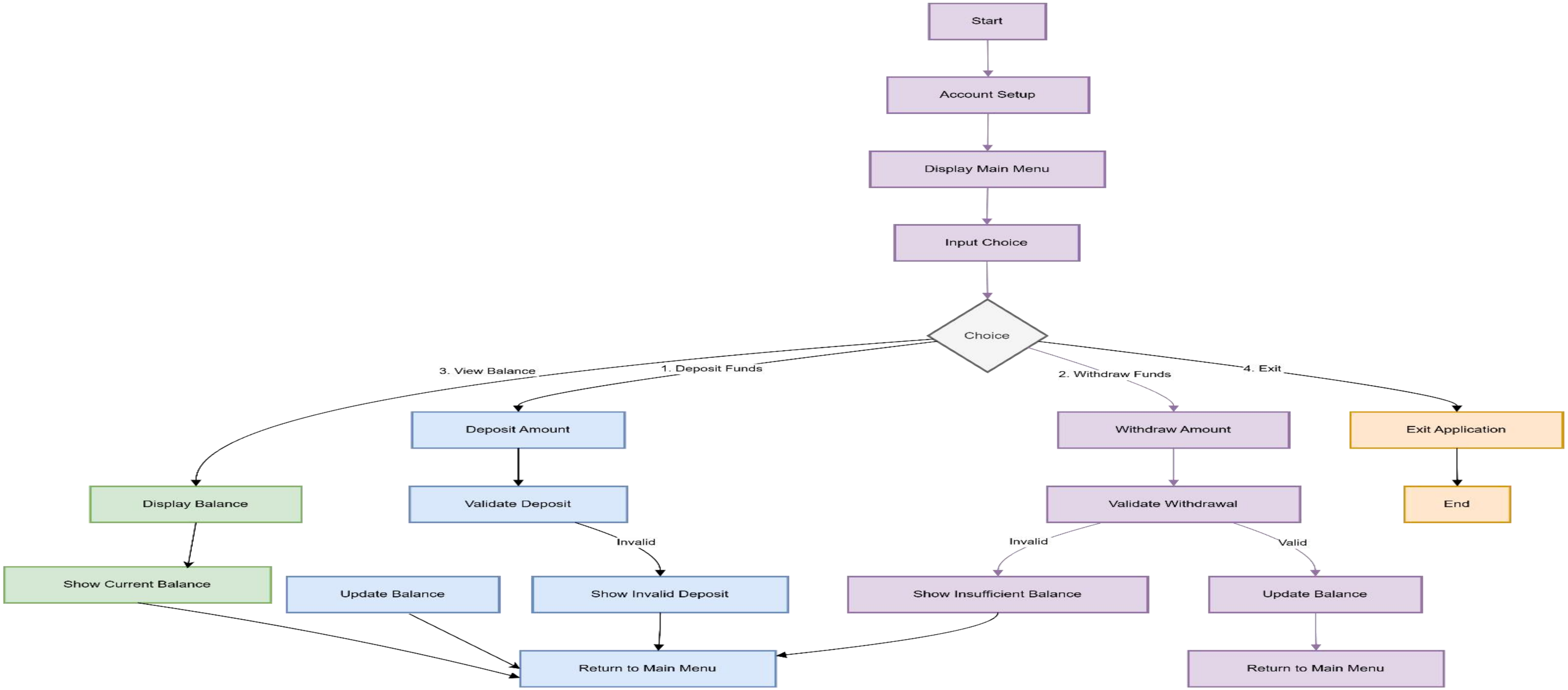
5. Error Handling:

To manage exceptions such as invalid account numbers or insufficient balance.

6. Modular Programming:

Breaking the system into smaller, manageable modules for clarity and maintainability.

ARCHITECTURE OF THE PROPOSED SYSTEM



LIST OF MODULES

MODULE DESCRIPTON

1. Account Management Module:

Create, update, or delete bank accounts.

2. Transaction Module:

Handle deposits, withdrawals, and transfers

3. Balance Inquiry Module:

Display the current balance of an account.

MERITS

1. User-friendly interface for easy navigation.

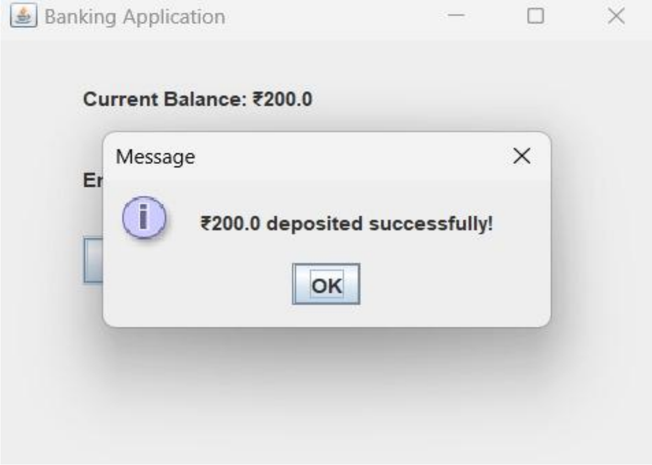
2. Lightweight and cost-effective solution.

3. Secure handling of user data and transactions.

4. Modular design enhances scalability and maintenance.

5. Suitable for educational purposes and small-scale implementations.

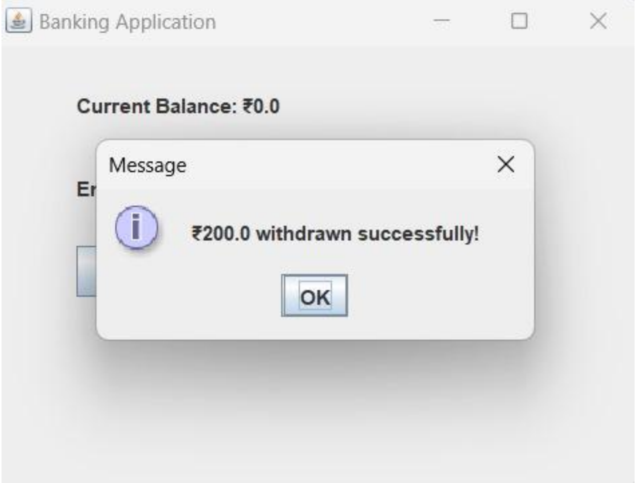
RESULTS AND DISCUSSION



Current Balance: ₹200.0

Er

₹200.0 deposited successfully!



Current Balance: ₹0.0

Er

₹200.0 withdrawn successfully!

QUERIES ?