IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 9, April 2025
Impact Factor: 7.67

Bank Management System

Prof. Samgir K.R[1] and Nikhil Prashant Kumbhar[2]
Department of Computer Engineering[1,2]
Navsahyadri Education Society's Group of Institution (Polytechnic), Pune, India

Abstract: *This project explores the design and development of a command-line based Bank Management System using the C programming language within the WSL (Windows Subsystem for Linux) environment. The system replicates essential banking functionalities such as account creation, balance inquiries, deposits, withdrawals, transaction history, and account deletion. A key feature of this system is the implementation of an Admin Module, which allows authorized personnel to perform advanced tasks like managing all user accounts, performing secure deletions, and viewing transaction logs. The main objective is to create a secure, efficient, and user-friendly terminal-based banking solution that not only enhances user interaction but also integrates fundamental programming concepts such as file handling, structures, and modular programming in C. This system also demonstrates real-world application of security mechanisms through authentication and restricted access for administrative functions.*

Keywords: Bank Management

I. INTRODUCTION

Banking systems play a vital role in managing financial transactions securely and efficiently. As digital banking becomes increasingly common, even basic banking operations can benefit from automation through programming. This project introduces a terminal-based Bank Management System developed using the C programming language and executed in a Linux-like environment via WSL (Windows Subsystem for Linux).The system replicates core banking features such as account creation, balance checks, deposits, withdrawals, transaction history tracking, and account deletion. To ensure administrative control and secure operations, the system includes a dedicated Admin Module with authentication protocols. This module enables secure access to account management tools such as viewing all accounts, editing or deleting user data, and reviewing activity logs.

By combining essential features with an admin interface, this project not only simulates a simplified banking environment but also provides hands-on experience with key programming concepts such as data structures, file handling, modular coding, and system security. It serves as a foundational project for students and developers aiming to understand and implement real-world logic using C in a Linux terminal setup.

II. LITERATURE REVIEW

Previous implementations of banking systems using C have mainly focused on simulating user-level transactions such as deposits, withdrawals, and balance inquiries through structure-based file handling. These systems demonstrate the effectiveness of using low-level programming to mimic real-life banking operations in a simple, resource-friendly manner.

Traditional banking software often utilizes robust databases and graphical interfaces; however, in educational or constrained environments, command-line and file-based systems offer an effective alternative. They allow developers to focus on the underlying logic and data management techniques without the overhead of advanced UI frameworks.

Moreover, incorporating an Admin Module into such systems addresses a commonly overlooked aspect—security and control. Studies and implementations in small-scale systems have shown the importance of separating general user functions from administrative privileges. Admin-controlled features like account editing, deletion, and full-system logs align with the principle of role-based access, a concept rooted in enterprise-level security models.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 9, April 2025
Impact Factor: 7.67

III. METHODOLOGY

The Bank Management System was developed using structured programming in C and executed within the WSL (Windows Subsystem for Linux) environment using the GCC compiler. The program follows a modular architecture, with clearly defined functions handling different banking operations, improving maintainability and scalability.

Key modules in the system include:

User Operations:

- Create Account – Gathers customer details and generates a unique account number.
- View Account – Retrieves and displays account information.
- Deposit – Adds a specified amount to the user's balance.
- Withdraw – Deducts funds from the user's account, ensuring sufficient balance.
- Transaction History – Logs all deposits and withdrawals for review.
- Delete Account – Erases all data related to a selected account.

Admin Module:

- Login Authentication – Requires a secure username and password to access admin privileges.
- View All Accounts – Displays a list of all users and account details.
- Search and Modify – Allows account lookup by number or name, with options to edit or delete.
- Audit Logs – Provides insight into transaction history and user activities.

All operations are accessible through a main menu loop, where the system handles user input and routes it to appropriate functions. Data is stored using structure-based serialization in files, ensuring persistence between sessions. Careful use of file pointers, error checking, and input validation enhances the system's reliability and user experience.

The design of the admin layer ensures restricted access, separating critical operations from regular user interactions and introducing basic role-based control.

IV. SYSTEM FRAMEWORK

The Bank Management System is organized into a set of modular components, each responsible for a specific aspect of functionality. This layered architecture enhances clarity, ease of maintenance, and scalability. The system consists of the following key modules:

Input Module

Responsible for collecting user information such as account holder name, account number, and transaction amounts. It ensures that the entered data follows the expected format and constraints.

Processing Module

Handles the core logic of the system. This includes balance updates, input validation, account verification, and execution of banking operations based on user or admin commands.

Storage Module

Uses file handling techniques to persist user and transaction data across sessions. File operations (fopen, fread, fwrite, etc.) are performed using structure-based serialization to maintain data integrity and accessibility.

Output Module

Displays account details, transaction results, and feedback messages to users through the command-line interface, ensuring clear communication and interaction.

Admin Module

Introduces a separate, secure layer of functionality for authorized system administrators. This module includes:

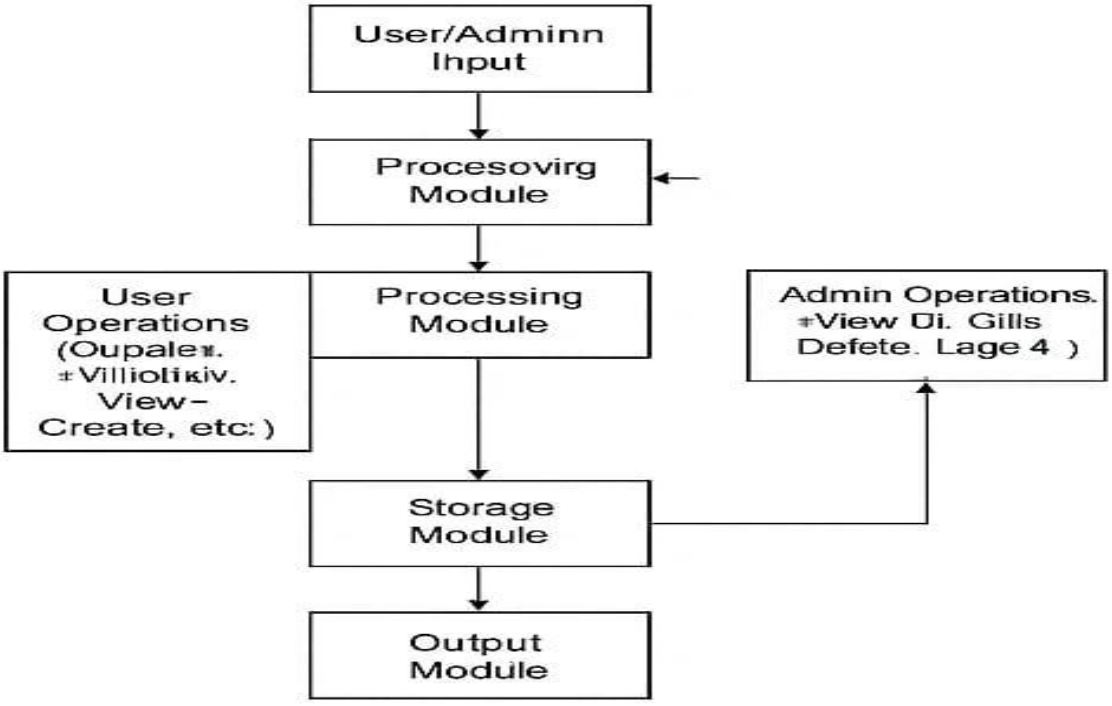
Authentication – Validates admin credentials before granting access to privileged features.

System-Wide Access – Provides visibility over all accounts and transactions.

Data Management – Allows for modification and deletion of user accounts.

Audit Features – Logs sensitive activities and system operations for review.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 9, April 2025
Impact Factor: 7.67



V. IMPLEMENTATION

The Bank Management System was implemented entirely in the C programming language, using a modular design to separate various banking operations. The program was compiled and tested in the WSL (Windows Subsystem for Linux) environment using the GCC compiler. The implementation involves a combination of structured data types, file handling, menu-driven interaction, and secure logic control.

Key components of the implementation include:

Structures:

User and transaction data are stored in struct types, which group account details (name, account number, balance, etc.) into manageable units.

File Handling: Persistent data storage is achieved using standard file I/O operations (fopen, fwrite, fread, fclose, etc.). All user account data and transaction history are saved in binary files for speed and security.

Menu Interface:

The system is navigated using a terminal-based menu, which continuously loops until the user chooses to exit. This makes the application intuitive and easy to use.

Admin Module:

The system includes an Admin Login section protected by a hardcoded username and password. Upon successful authentication, the admin can:

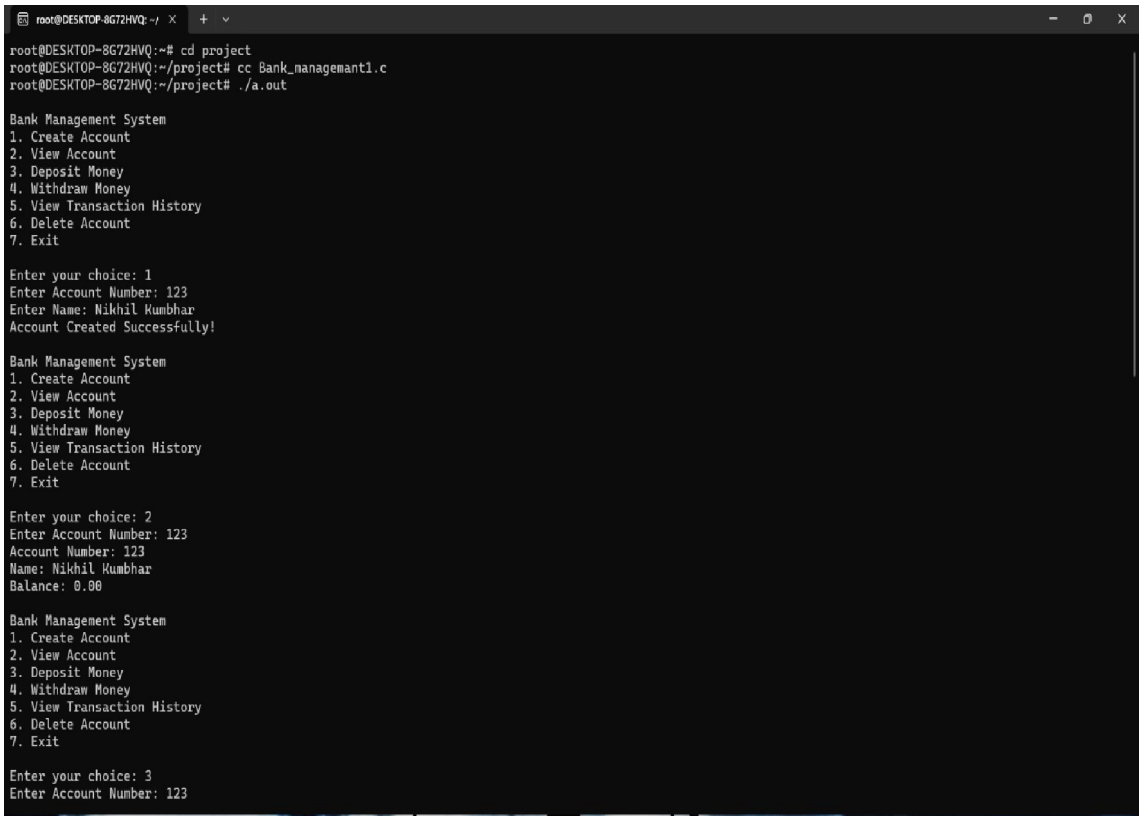
- View all accounts
- Search and edit account data
- Delete user accounts
- Review transaction logs for audit purposes
- Error Handling:

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 9, April 2025
Impact Factor: 7.67

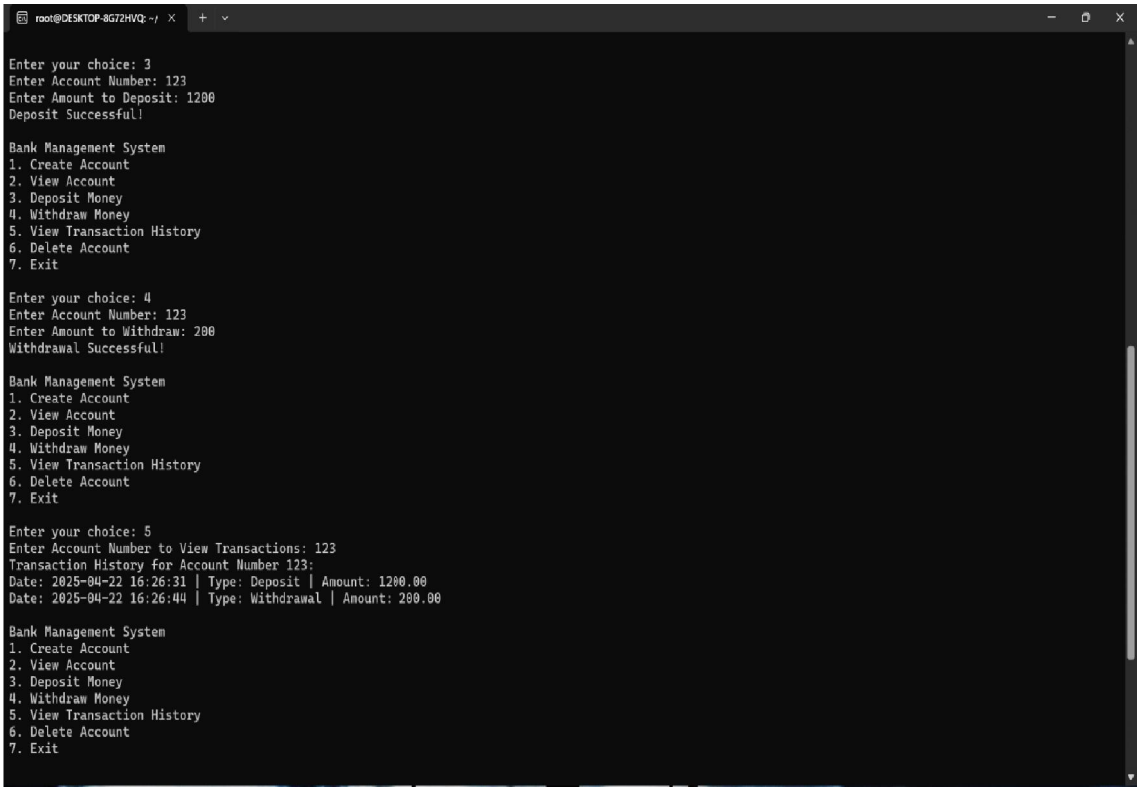
Defensive programming techniques such as file existence checks, input validation, and conditional checks help ensure smooth and safe execution.

Test Cases:

The system has been tested with multiple use cases, including account creation, balance verification after transactions, edge cases like insufficient balance on withdrawal, and admin-level data deletions.

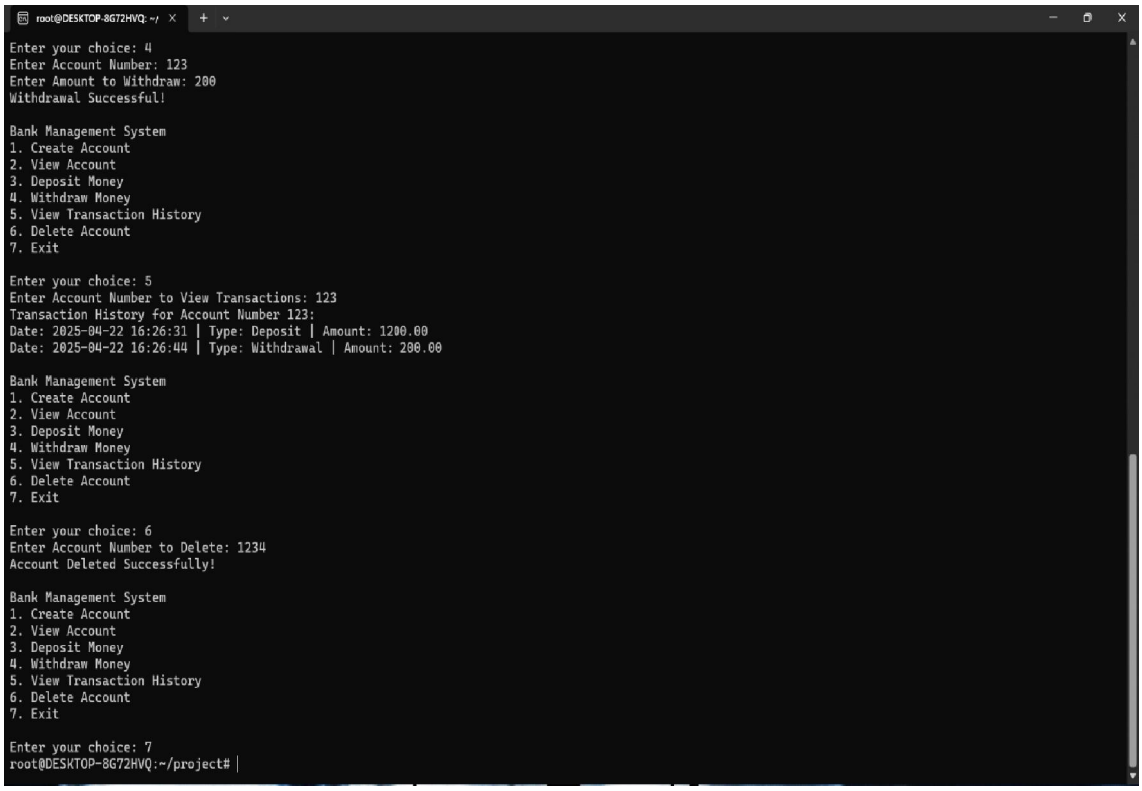


IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 9, April 2025
Impact Factor: 7.67



DOI: 10.48175/IJARSCT-25760

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 9, April 2025
Impact Factor: 7.67



VI. CONCLUSION

The Bank Management System developed using C programming language and executed within the WSL (Windows Subsystem for Linux) provides a reliable and efficient command-line application for simulating essential banking operations. The system supports a range of features including account creation, deposits, withdrawals, balance inquiries, and transaction tracking—delivered through a user-friendly, menu-driven interface.

One of the project's significant achievements is the integration of an Admin Module, offering secure access to sensitive operations such as modifying or deleting accounts and viewing system-wide logs. This not only improves control and transparency but also introduces a level of role-based access similar to real-world banking applications.

The project successfully demonstrates the application of key programming concepts such as structures, file handling, modular design, and authentication in C. It serves as an excellent educational model for understanding real-life systems in a controlled, terminal-based environment. With further enhancements—such as encryption, database integration, and GUI—this project has the potential to evolve into a more comprehensive banking software.

REFERENCES

[1]. Yashavant Kanetkar. (2004). Let Us C. BPB Publications. – A foundational book for learning C programming, with practical examples and exercises.

[2]. Brian W. Kernighan & Dennis M. Ritchie. (1988). The C Programming Language. Prentice Hall.– The definitive guide to the C language by its original creators.

[3]. TutorialsPoint. File Handling in C. Available at: https://www.tutorialspoint.com/cprogramming/c_file_io.htm – A practical resource for understanding file operations in C.

[4]. Linux Man Pages – File I/O. Available at: https://man7.org/linux/man-pages/ – Official Linux documentation for system calls and file handling.

IJARSCT
International Journal of Advanced Research in Science, Communication and Technology
International Open-Access, Double-Blind, Peer-Reviewed, Refereed, Multidisciplinary Online Journal
ISSN: 2581-9429
Volume 5, Issue 9, April 2025
Impact Factor: 7.67

[5]. Stack Overflow Community Discussions. – Community-driven help and solutions for common C programming challenges during development.

[6]. GeeksforGeeks. Bank Management System using C. – Examples and tutorials related to building basic C applications, especially for student projects.