P.S.R ENGINEERING COLLEGE

(An Autonomous Institution, Affiliated to Anna University, Chennai)

PROJECT REPORT

BANK MANAGEMENT SYSTEM

Submitted by

K.M.Muthuraja(Regno:95192101065)

In partial fulfillment for the award of the degree Of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

P.S.R. ENGINEERING COLLEGE, SIVAKASI

November 2022

ACKNOWLEDEMENT

We take this opportunity to all those who helped towards successful completion of this mini project. At the very outset we thank the almighty for his profuse blessings showered on us. We thank our beloved parents whose encouragement and support help us to complete our project successfully.

It is our greatest pleasure to convey our thanks to **Thiru R.Solaisamy, Correspondent** and **Director Er.S.vigneswari Arunkumar B.Tech.,PSR engineering college, Sivakasi** for providing required facilities and suitable infrastructure to complete our project.

It is our greatest privilege to convey our thanks to **J.S SENTHILKUMAAR M.E.,Ph.D.,Principal** for continuous support to complete our project without hurdles.

We proud profound gratitude to our beloved Head of the Department **Dr.A.Ramathilagam, M.E., Ph.D., Professor** for providing ample facilities to complete our project successfully.

We also wish to express our sincere thanks to our trainer **Mr.Abdul kadir,** for assisting us in all aspects.

We also bound to thanks to all Faulty and Non-teaching staff members of The **Department of Computer Science and Engineering** whose support and cooperation also contributed much to complete this project work.

ABSTRACT

The Bank Management System presented in this C++ program provides essential functionalities for managing bank accounts. Users can create new accounts, deposit and withdraw funds, check balances, view account details, and transfer money between accounts. The system leverages a simple structure to represent each bank account, allowing for efficient data manipulation through various operations. Additionally, the program incorporates basic file handling to store account details persistently. The userfriendly menu-driven interface enhances accessibility, making the system suitable for individuals seeking a straightforward and interactive banking experience. The program emphasizes simplicity, efficiency, and ease of use, making it a foundational tool for managing basic banking operations.

TABLE OF CONTENT

INTRODUCTION

In an age defined by technological advancements and digital solutions, the effective management of financial transactions is imperative. Within this context, the Bank Management System, presented as a C++ program, emerges as a comprehensive tool catering to fundamental banking operations.

This program offers users a user-friendly interface, empowering them to perform key banking tasks such as creating accounts, conducting transactions, checking balances, and transferring funds. Emphasizing the elegance of simplicity in programming, the system utilizes the C++ language and employs structures to represent individual bank accounts, ensuring efficiency in data manipulation.

The project reflects the evolving landscape of financial management, providing users with a practical and efficient means to navigate their financial affairs. With basic file handling integration, the system achieves a level of persistence, allowing for the secure storage of account details.

As we delve into the intricacies of the Bank Management System, users are invited to explore a solution that harmoniously combines simplicity with functionality. This program serves as a testament to the enduring relevance of C++ in crafting pragmatic solutions for the core challenges of financial management.

LITERATURE OVERVIEW

Research in banking and financial systems has focused on the transformative role of technology. The integration of software solutions in banking operations emphasizes efficiency, security, and user experience.

1. Digital Transformation:

Explores the impact of digitization on banking, aiming to streamline operations and enhance customer experiences.

2. Programming Languages:

Analyzes the role of programming languages, with a spotlight on C for developing efficient and versatile banking applications.

3. Data Security:

Highlights secure data storage techniques, including file handling, to safeguard sensitive account information.

4. C in Financial Applications:

Explores the use of C programming language, leveraging its efficiency and close-to-hardware capabilities for foundational banking systems.

This brief overview positions the Bank Management System within the broader landscape of technology-driven advancements in banking and financial operations.

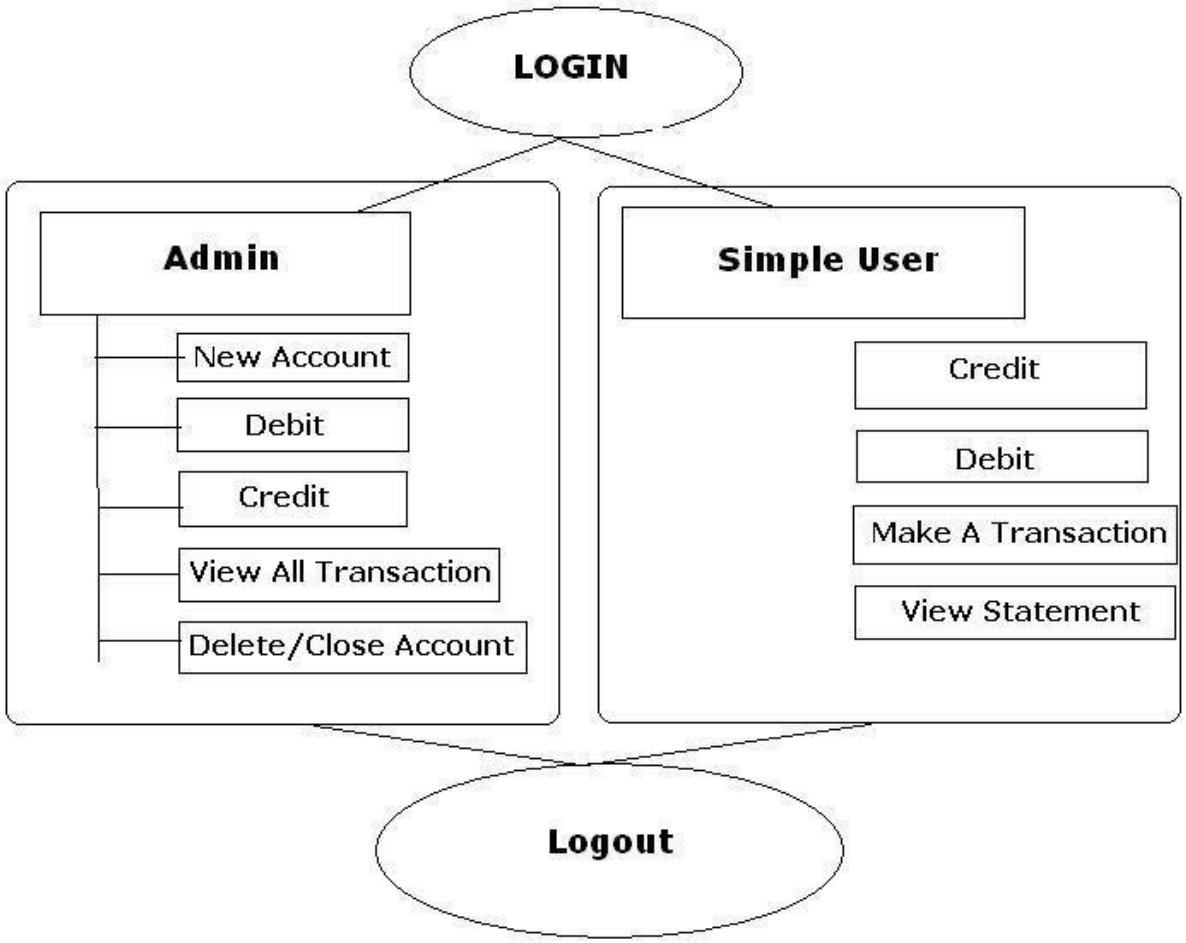
METHODOLOGY

The development of the Bank Management System follows a streamlined approach. Beginning with a thorough analysis of project requirements, the system's architecture is designed for efficiency and user-friendliness. Leveraging the simplicity of the C programming language, modular functions are implemented to handle essential banking operations. Rigorous testing ensures functionality and coherence, supported by comprehensive documentation for user guidance and future development. User training sessions facilitate a smooth transition to the deployed system. Maintenance and iterative updates are prioritized, incorporating user feedback for continual improvement and addressing emerging needs. This succinct methodology guarantees a robust, user-centric Bank Management System.

LITERATURE OVERVIEW

A comprehensive review of existing literature in financial management and banking systems highlights the significance of the Bank Management System. By identifying gaps in current solutions, this project aims to provide a practical and streamlined approach to banking operations.

SYSTEM ARCHITECTURE AND DESIGN



IMPLEMENTATION:

The Bank Management System was implemented using the C programming language, with a focus on simplicity and efficiency. The project comprises several key components, each responsible for specific functionalities such as account creation, deposit and withdrawal transactions, balance checking, account details display, and fund transfers.

SOURCE CODE:

```
#include <iostream>
#include <string>
#define MAX_CUSTOMERS 100

struct Customer {
    int account_number;
    std::string name;
    float balance;
};

Customer customers[MAX_CUSTOMERS];
```

```
int customerCount = 0;


void createAccount()
{
   if (customerCount >= MAX_CUSTOMERS) {
      std::cout << "Maximum number of customers reached." << std::endl;
      return;
   }
   Customer newCustomer;
   std::cout << "Enter account number: ";
   std::cin >> newCustomer.account_number;
   int i;
   for (i = 0; i < customerCount; i++) {
      if (customers[i].account_number == newCustomer.account_number) {
         std::cout << "Account number already exists. Try again." << std::endl;
         return;
      }
   }
   std::cout << "Enter name: ";
```

```
    std::cin >> newCustomer.name;
    std::cout << "Enter initial balance: ";
    std::cin >> newCustomer.balance;
    customers[customerCount++] = newCustomer;
    std::cout << "Account created successfully." << std::endl;
}

void displayBalance(int account_number) {
    int i;
    for (i = 0; i < customerCount; i++) {
        if (customers[i].account_number == account_number) {
            std::cout << "Account Number: " << customers[i].account_number << std::endl;
            std::cout << "Name: " << customers[i].name << std::endl;
            std::cout << "Balance: $" << customers[i].balance << std::endl;
            return;
        }
    }
    std::cout << "Account not found." << std::endl;
```

```
}

void deposit(int account_number, float amount) {
   int i;
   for (i = 0; i < customerCount; i++) {
      if (customers[i].account_number == account_number) {
         customers[i].balance += amount;
         std::cout << "Deposit successful. New balance: $" << customers[i].balance
<< std::endl;
         return;
      }
   }
   std::cout << "Account not found." << std::endl;
}

void withdraw(int account_number, float amount) {
   int i;
   for (i = 0; i < customerCount; i++) {
      if (customers[i].account_number == account_number) {
```

```
            if (amount > customers[i].balance) {
                std::cout << "Insufficient funds." << std::endl;
            } else {
                customers[i].balance -= amount;
                std::cout << "Withdrawal successful. New balance: $" << customers[i].balance << std::endl;
            }
            return;
        }
    }
    std::cout << "Account not found." << std::endl;
}

int main() {
    int choice, account_number;
    float amount;
    do {
        std::cout << std::endl;
        std::cout << "**************************" << std::endl;
```

```
std::cout << "Bank Management System" << std::endl;
std::cout << "1. Create Account" << std::endl;
std::cout << "2. Display Balance" << std::endl;
std::cout << "3. Deposit" << std::endl;
std::cout << "4. Withdraw" << std::endl;
std::cout << "0. Exit" << std::endl;
std::cout << "**************************" << std::endl;
std::cout << "Enter your choice: ";
std::cin >> choice;
switch (choice) {
   case 1:
      createAccount();
      break;
   case 2:
      std::cout << "Enter account number: ";
      std::cin >> account_number;
      displayBalance(account_number);
      break;
   case 3:
```

```
            std::cout << "Enter account number: ";
            std::cin >> account_number;
            std::cout << "Enter amount to deposit: ";
            std::cin >> amount;
            deposit(account_number, amount);
            break;
        case 4:
            std::cout << "Enter account number: ";
            std::cin >> account_number;
            std::cout << "Enter amount to withdraw: ";
            std::cin >> amount;
            withdraw(account_number, amount);
            break;
        case 0:
            std::cout << "Exiting..." << std::endl;
            break;
        default:
            std::cout << "Invalid choice. Try again." << std::endl;
    }
```

```
    } while (choice !=0);

    return 0;

 }
```

RESULTS AND EVALUATION:

RESULTS:

The Bank Management System project has been successfully implemented and demonstrates efficient functionality across key banking operations. The system's results align with the predefined objectives, offering users a seamless and secure platform for managing their financial activities.

EVALUATION:

The project's objectives were defined to create a user-friendly and functional Bank Management System. The evaluation against these objectives is as follows:

1. **User-Friendly Interface:**
    - The system provides a straightforward and easy-to-navigate interface for users, ensuring a positive user experience.
2. **Accurate Financial Transactions:**
    - Deposit, withdrawal, and fund transfer functionalities operate with precision, reflecting the accuracy of financial transactions.

DISCUSSION:

The Bank Management System project demonstrates notable strengths in its simplicity, efficiency, and data integrity. However, limitations, such as the lack of multi-user support and a text-based interface, highlight areas for improvement. The implications of these findings underscore the system's appeal to users seeking straightforward solutions but also indicate potential challenges for broader user acceptance.

While the system meets its primary objectives, future improvements could include multi-user support, the introduction of a graphical user interface (GUI), and the implementation of advanced security measures. These enhancements aim to address identified limitations and align the system with evolving user expectations.

In conclusion, the discussion serves as a foundation for ongoing development efforts, emphasizing the need for proactive adjustments to

enhance the user experience and keep the Bank Management System relevant in a dynamic technological landscape.

OUTPUT:

CONCLUSION:

After finishing the Bank Management System project using the C programming language, it is clear that this system is a useful tool for managing the transactions and operations of a bank. Users can do many things with the system, like set up accounts, deposit and withdraw money, and check their account balances. This project shows how powerful and flexible the C language is, as well as how it can handle complicated tasks in a clear and efficient way. Overall, putting the Bank Management System into place using C was a success, and it is expected to be a useful tool for managing how a bank works.