**Bank Management System**

**PROJECT SYNOPSIS**

A OOPs Based Bank Management System

**BACHELOR OF COMPUTER APPLICATIONS**

**SUBMITTED BY:**  **GUIDED BY:**

Mukesh Shrivastva (21401014) Mr. Ashutosh Kumar Singh

Rohit Kumar (21401027)



**SARDAR VALLABHBHAI PATEL COLLEGE (S.V.P)**

**BHABUA (KAIMUR)**

**INDEX**

|  |  |
| --- | --- |
| **SR No.** | **TOPIC** |
| 1 | Executive Summary |
| 2 | Introduction |
| 3 | Objective and Scope |
| 4 | Methodology |
| 5 | Expected Outcomes |
| 6 | Conclusion |
| 7 | Appendices or References |

# Executive Summary

## This report presents a simple banking system developed using C++. The system offers essential functionalities such as account creation, deposit, withdrawal, and account information retrieval. While the system is functional, it lacks advanced features like transaction history, interest calculations, and robust security measures. The code serves as a basic framework for understanding fundamental concepts of programming, file handling, and OOP in the context of a simplified banking application.

## Introduction

The primary goal of this project was to develop a functional, albeit rudimentary, banking system using the C++ programming language. This system aimed to demonstrate the application of core programming concepts like file handling, object-oriented programming (OOP), and user interaction. While not designed for real-world banking operations, the system provides a foundational structure for understanding the fundamental aspects of building a basic banking application.

**Objective and Scope**

The objective of this project was to create a program that allows users to perform the following basic banking operations:

**Account Creation:** Users can generate new accounts with unique account numbers, storing their name, CNIC, phone number, email address, and initial deposit amount.

**Deposit:** Existing account holders can deposit money into their accounts.

**Withdrawal:** Existing account holders can withdraw money from their accounts (subject to sufficient balance).

**Account Information Retrieval:** Users can view the details of their account, including their name, CNIC, phone number, email address, and current balance.

The scope of this project focuses solely on these core functionalities. The system does not incorporate advanced features such as transaction history, interest calculation, loan management, online banking capabilities, or robust security measures.

**Methodology**

The banking system was developed using the following methodologies and concepts:

**Object-Oriented Programming (OOP):** The program uses the bank class to encapsulate data related to each account (account number, name, CNIC, phone number, email, and balance) and methods for managing account operations (creation, deposit, withdrawal, and information retrieval).

**File Handling:** The system utilizes text files for data storage. Account information is stored in a file named data.txt, where each line represents an account and contains the details separated by tabs.

**Random Number Generation:** The system employs srand(time(0)) and rand() functions to generate unique account numbers.

**User Interface:** The program features a simple menu-driven interface using cout and cin for user interaction. The menu allows users to choose between creating accounts, logging in, and exiting the system.

**Expected Outcomes**

This project aimed to develop a program that meets the following expectations:

**Functionality:** The program should enable users to create accounts, deposit and withdraw money, and check their account information.

**Data Management:** The system should store account data in a structured way, ensuring the integrity of data.

**User Friendliness:** The user interface should be intuitive and easy to navigate.

**Conclusion**

This report has presented a basic banking system designed to showcase fundamental programming concepts in C++. The system demonstrates basic file handling, OOP, and user interaction through a menu-driven interface. While functional, the system is limited in scope and lacks advanced features such as security, transaction history, interest calculations, and more sophisticated data storage methods. It serves as a foundation for further development, where more complex functionalities and real-world considerations can be incorporated to build a more robust and feature-rich banking application.

**Appendices or References**

* **The C++ Programming Language** by Bjarne Stroustrup.
* **ISO C++ Official Documentation**: isocpp.org.
* **cppreference.com**: cppreference.com.
* **C++ Core Guidelines**: isocpp.github.io.
* **C++ Primer** by Stanley B. Lippman, Josée Lajoie, and Barbara E. Moo.
* **The C++ Standard Library** by Nicolai M. Josuttis.
* **C++ Templates: The Complete Guide** by David Vandevoorde and Nicolai M. Josuttis.
* **Modern C++ Design** by Andrei Alexandrescu.
* **Accelerated C++** by Andrew Koenig and Barbara E. Moo.
* **Clean Code in C++** by Stephan Roth.
* **C++ Crash Course** by Josh Lospinoso.
* **Design Patterns** by Erich Gamma, Richard Helm, Ralph Johnson, John Vlissides.