CCP – Proposal

PF (CT-175)

Team Name:

CodeCubed

Group Members:

Arbish Tehseen (CT-25053)

Aatiqa Batool (CT-25053)

Saita Ahuja (CT-25053)

Discipline : BCIT

Teacher: Sir Muhammad Abdullah

1. Project Title:

Bank Management System.

2. Project Description:

The Bank Service Management System is a C-based project designed to automate and simplify basic banking operations. It includes modules for Account Management, ATM Services, Security, Help Desk, and Admin Panel, offering users a complete banking experience. Customers can create accounts, perform deposits, withdrawals, transfers, and check balances, while the admin can view, delete, or reset accounts

The system ensures secure access through password authentication and provides a help desk for common queries. Its goal is to replace manual banking with an efficient, secure, and user-friendly digital solution while applying key programming fundamentals like arrays, structures, loops, and functions.

3. Project Methodology:

3.1 Dataset:

The system stores all banking data internally using arrays:

Account details: names, account numbers, and passwords

Financial data: account balances

Transaction history: last 10 transactions per account.

Other: data admin credentials and counters for total accounts

3.2 Tools and Technologies:

Language: C Language

Libraries: stdio.h for input/output operations like printf, scanf.

string.h-for string operations like strcmp and strcpy.

Environment: Dev C++

3.3 Algorithm:

Model Type: Modular Procedural System

Modules/Layers: Admin, ATM Services, Helpdesk, Additional Features

Functions/Logic: User and admin validation, transaction processing, account management, interest calculation, and reporting

3.4 Objectives:

- Develop a **secure** and **reliable bank management syste**m to handle customer accounts and transactions.
- Implement **ATM services** including withdrawals, deposits, balance checks, and money transfers.
- Create an **admin panel** to add, delete, and manage customer accounts securely.
- Maintain **transaction history** for each amount to track, deposit, withdrawals, and transfers.
- Provide a **helpdesk module** to guide users on banking operations included.
- **Additional features** like interest calculation, highest balance tracking, and account summaries.

3.5 Timeline

WEEK	TASK
Week 1	Problem definition, listing features, and dataset preparation (arrays for accounts, balances, transactions)
Week 2	System design, planning program flow, and designing modules (Admin, ATM, Helpdesk, Additional Features)
Week 3	Coding individual modules: Admin panel, ATM services, Helpdesk, and Additional Features
Week 4	Integration of all modules, testing of transactions, account management, and additional features
Week 5	Debugging, improving functionalities, and finalizing user interaction
Week 6	Documentation, report writing, and preparation for demonstration
Week 7	Final testing, verification of all modules, and polishing the system
Week 8	Final presentation preparation

3.6 Goals

- To **learn and apply programming concepts** in a real-world banking system.
 To **understand modular system design and data management**.
- To **gain experience in handling secure transactions and user authentication**.
- To **develop problem-solving and project development skills**.

3.7 Expected Outcomes

- A **working Bank Management System** that allows users to deposit, withdraw, transfer money, and check balances.
- A **secure admin panel** to manage customer accounts safely.
- A **helpdesk module and transaction history** to guide users and track all account activities.

4. Justification - Why it is a Complex Computing Problem:

Multiple Interdependent Modules:

- The system has several modules (accounts, transactions, ATM services, helpdesk, security monitoring) that must work seamlessly together, increasing complexity.

Data Security and Integrity:

- Handling sensitive banking data requires secure storage, authentication, and transaction validation, which adds computational and algorithmic complexity.

Real-time Operations:

- Operations like withdrawals, deposits, and ATM transactions need to be processed efficiently in real-time without errors or data loss.

Error Handling and Fault Tolerance:

- The system must anticipate and manage errors, such as invalid transactions or system crashes, to ensure reliability.

Integration of Diverse Data Types:

- Combines numerical, textual, and potentially image-based data (for ATM receipts or customer verification), requiring careful data processing and storage.

Algorithmic Complexity:

- Implementing secure authentication, transaction validation, and modular workflow involves algorithm design and logical sequencing.

6. Industrialization/Commercial Product Potential:

Market Demand:

- Banks, credit unions, and financial institutions constantly need efficient and secure management systems to handle customer accounts, transactions, and ATM services.

Scalability for Businesses:

- The system can be adapted for small to medium-sized banks and extended to larger financial networks, making it commercially viable.

Revenue Potential:

- Can be packaged as a software product for banks or fintech startups, with potential for licensing, subscription, or service-based revenue.

Integration with Modern Technologies:

- Can be enhanced with online banking, mobile apps, cybersecurity modules, and AI-based fraud detection, increasing industrial appeal.

This project has the potential to be developed into a **practical software system for banks or financial institutions**. It can manage **customer accounts, transactions, and ATM services** in a secure and organized way. The modular design allows it to be **easily expanded**, so additional features like **customer support, reporting, or online banking** can be added in the future. With further improvements, it could become a **reliable tool for small and medium-sized banks**, helping them **manage operations efficiently and safely**.