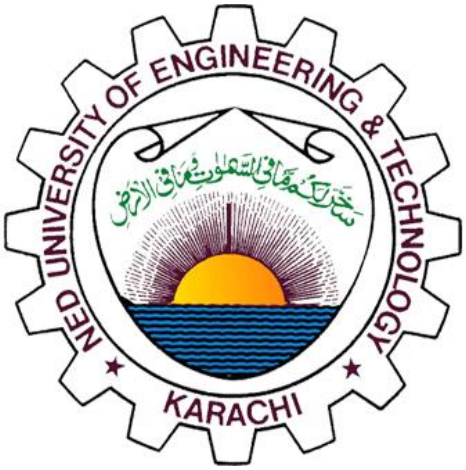

NED University of Engineering and Technology

Department of Computer Science and Information Technology

Complex Computing Problem (CCP)

Proposal

Programming Fundamentals (CT-175)

BANKING SYSTEM WITH ATM AND PASSWORD ENCRYPTION

Group Name: CODESTORM

Aryan Ahmed CT-176

Arslan Ahmed CT-184 **GROUP LEADER**

Muhammad Sami CT-185

Instructor: Sir Abdullah

1. Project Title

BANKING SYSTEM WITH ATM AND PASSWORD ENCRYPTION

2. Project Description

The Banking System with ATM and Password Encryption is a secure application that simulates core banking and ATM operations such as account creation, balance inquiry, deposit, withdrawal, and fast cash. It ensures data security by encrypting user passwords using xor cipher, preventing unauthorized access.

3. Project Methodology

3.1 Key Features:

- **User Authentication:** Secure login using encrypted passwords.
- **ATM Functionalities:** Balance inquiry, cash withdrawal, fast cash, and deposit operations.
- **Account Management:** Create, update, and login user.
- **Password Encryption:** Utilizes hashing or encryption techniques (xor cipher) to safeguard credentials.
- **Error Handling:** Prevents invalid transactions such as overdrafts or incorrect PIN entries.

3.2 Objectives:

- To design a reliable and user-friendly banking system with ATM functionalities.
- To ensure data security using password encryption and secure authentication.
- To demonstrate the application of encryption algorithms in financial systems.
- To provide users with a simulation of real-time banking operations.

3.3 Technologies Used:

- **Programming Language:** C language
- **Encryption:** Xor Cipher
- **Interface:** Command-line interface (CLI)

3.4 Algorithm:

1. **Start**
2. Display main menu: Create Account */ Login*
3. **If** user selects *Register*:
    a. Input user details (name, account number, PIN/password, Email, balance).
    b. Encrypt the password using an encryption algorithm (xor cipher).
    c. Store encrypted password and user data in the file.
4. **If** user selects *Login*:
    a. Input user's name and password
    b. Encrypt entered password and compare with stored encrypted password
    c. **If** match found → grant access; else → display "Invalid Credentials"
5. Display ATM menu: *Balance Inquiry / Deposit / Withdraw / Fast Cash*.
6. **If** Balance Inquiry → display current balance
7. **If** Deposit → input amount, update balance, and record transaction
8. **If** Withdraw → check sufficient balance, deduct amount, and record transaction
9. **If** Fast Cash → display few amounts and subtract the choosen amount feom user's balance
10. Store all transactions in the file for record-keeping
11. **End**

3.5 Expected Outcome:

The final system will provide a secure, efficient, and user-oriented banking platform that demonstrates how encryption enhances data protection in financial transactions. It will serve as both an educational and practical example of implementing cybersecurity in banking applications.

3.6 Future Goal:

- **Implement Online Banking Access:** Extend the system to support web and mobile platforms for remote banking operations.
- **Add Biometric Authentication:** Integrate fingerprint or facial recognition for enhanced security.
- **Enhance Encryption Techniques:** Upgrade to advanced encryption standards like RSA or hybrid cryptography for stronger data protection.
- **Introduce AI-based Fraud Detection:** Use machine learning to monitor transactions and detect suspicious activities.

- **Enable Multi-Currency Support:** Allow transactions in different currencies with real-time exchange rate updates.
- **Generate Automated Reports:** Provide detailed financial and transaction reports for users and administrators.
- **Improve User Interface:** Develop a modern, intuitive graphical interface for better user experience.

4. Justification – Why is it a Complex Program:

This program is **complex** because it integrates multiple core programming concepts, system functionalities, and security mechanisms to simulate a real-world banking environment. It is not a simple linear application but a **multi-functional system** that handles user authentication, encrypted password management, transaction processing, and persistent data storage.

The complexity arises from:

1. **File Handling:**
   The program reads and writes user data to external files, ensuring data persistence across multiple sessions. This requires structured data management and error handling for file operations.
2. **Password Encryption:**
   The XOR-based encryption system adds a layer of security, requiring careful encryption and decryption during both registration and login processes.
3. **User Authentication and Account Management:**
   It validates login credentials, handles blocked accounts, and maintains secure access control for multiple users simultaneously.
4. **Transaction Processing:**
   It supports multiple ATM operations (balance inquiry, deposit, withdrawal, fast cash), each updating and saving data consistently in real time.
5. **Structured Data Representation:**
   The program uses **structures (structs)** and arrays to organize user records, making the data handling modular and efficient.
6. **Error and Input Handling:**
   The program must validate user inputs, handle invalid entries gracefully, and maintain logical consistency during transactions.
7. **System Integration:**
   It combines **file I/O, encryption, conditionals, loops, structures, and real-time**

time-stamping to produce realistic transaction slips — reflecting real banking processes.

5. Industrialization:

This project can be industrialized by integrating it with real banking networks and secure database servers to handle large-scale customer data and transactions. By upgrading the current system with advanced encryption algorithms, multi-factor authentication, and real-time networking, it can be transformed into a fully functional banking solution suitable for ATM terminals or digital banking platforms used in the financial industry.