Banking Management System

A Mini Project Report

Abstract

The **Banking Management System** aims to simulate the core functionalities of a modern banking environment using concepts from operating systems and computer networks. It provides role-based access control for multiple users including Customers, Bank Employees, Managers, and Administrators. Each role can perform specific operations such as account management, transaction processing, loan handling, and administrative control.

The system ensures **data consistency, synchronization, and secure communication** through file management, locking mechanisms, and process synchronization using system calls and socket programming.

1. Introduction

The Banking Management System is designed to emulate the operations of a real-world bank using a client-server model. Multiple users interact with the server concurrently, performing actions like deposits, withdrawals, transfers, and loan processing.

This project emphasizes secure, concurrent access and synchronization to prevent race conditions during simultaneous transactions. The server handles multiple clients using sockets and ensures correct file-level locking to maintain data integrity.

2. System Overview

2.1. Objectives

- Simulate real-world banking operations digitally.
- Provide secure and concurrent access to different users.
- Ensure data integrity and consistency during concurrent operations.
- Implement role-based access control for distinct user responsibilities.

2.2. Technologies Used

- **Programming Language:** C
- **Networking:** Socket Programming (TCP)
- **Concurrency:** Fork, Threads, Semaphores
- **OS Concepts:** File Locking, System Calls, Inter-process Communication

3. System Design

3.1. Architecture

The system follows a **client-server architecture**:

- The **Server** maintains the database and manages all user requests.
- The **Clients** (Customers, Employees, Managers, Administrators) connect to the server using TCP sockets and perform their operations remotely.

3.2. File Structure

- `customer.dat` – Stores customer account details.
- `employee.dat` – Stores employee details and assigned loan cases.
- `manager.dat` – Stores manager credentials and activity records.
- `admin.dat` – Stores administrator credentials and system-level info.
- `transaction.dat` – Logs all deposit, withdrawal, and transfer activities.
- `loan.dat` – Maintains loan application details and statuses.
- `feedbacks.dat` – Stores customer feedback for managerial review.

4. Functional Modules

4.1. Overview

The Banking Management System provides different functionalities based on user roles. Each user interacts with the system through a secured login mechanism, ensuring proper authorization.

5. Functionalities Explanation

5.1. Customer Functionalities

1. **Login System (One Session Per User):** Customers authenticate with unique credentials. Only one session per user is permitted for security.
2. **View Account Balance:** Displays the customer's current balance using locked read access to prevent inconsistent reads.
3. **Deposit Money:** Allows deposits, updates balance, and records the transaction.
4. **Withdraw Money:** Withdraws funds after verifying sufficient balance and logs the operation.
5. **Transfer Funds:** Transfers funds between accounts with atomic operations and locking.
6. **Apply for a Loan:** Customer submits loan details; stored in `loan.dat` for employee processing.
7. **Change Password:** Securely updates the user's encrypted password.
8. **Add Feedback:** Adds customer feedback to `feedbacks.dat` for managerial review.
9. **View Transaction History:** Displays past deposits, withdrawals, and transfers.
10. **Logout / Exit:** Ends the session safely, releasing all locks and closing the socket.

5.2. Bank Employee Functionalities

1. **Login System (One Session Per User):** Employee login with controlled single-session access.
2. **Add New Customer:** Registers a new customer with initial balance and credentials.
3. **Modify Customer Details:** Updates existing customer information.
4. **Process Loan Applications:** Reviews loan requests assigned by the manager.
5. **Approve / Reject Loans:** Approves or rejects loans and updates records accordingly.

6. **View Assigned Loan Applications:** Displays all loans assigned for processing.
7. **View Customer Transactions:** Allows employees to view customer transaction history for verification.
8. **Change Password:** Updates employee password securely.
9. **Logout / Exit:** Ends employee session and releases active resources.

5.3. Manager Functionalities

1. **Login System (One Session Per User):** Managers log in securely with restricted access.
2. **Activate / Deactivate Customer Accounts:** Controls account activity status for customers.
3. **Assign Loan Application Processes:** Distributes loan applications among employees.
4. **Review Customer Feedback:** Reads and evaluates feedback records for improvement.
5. **Change Password:** Updates manager password to maintain secure access.
6. **Logout / Exit:** Terminates the session cleanly and closes all open handles.

5.4. Administrator Functionalities

1. **Login System (One Session Per User):** Administrator logs in with the highest privileges.
2. **Add New Bank Employee:** Adds employee records in `employee.dat`.
3. **Modify Customer / Employee Details:** Updates or corrects customer/employee data.
4. **Manage User Roles:** Changes or assigns roles such as Customer, Employee, or Manager.
5. **Change Password:** Allows administrator to update their password securely.
6. **Logout / Exit:** Closes all open sessions and releases server resources.

5.5. Summary

Each role in the Banking Management System has distinct privileges. Access control ensures that customers, employees, managers, and administrators perform only authorized actions, maintaining the system's reliability and data consistency.

6. Technical Implementation

6.1. Concurrency and Synchronization

- File locking prevents race conditions during concurrent read/write.
- System calls like `read()`, `write()`, `lseek()` and `fork()` are used instead of library functions.
- Semaphores and mutexes ensure safe multi-client operations.

6.2. Socket Programming

- The server listens on a port (e.g., 8080) and accepts client connections.
- Each client connection is handled by a separate child process.
- Reliable communication is ensured using TCP sockets.

6.3. Database Management

- All user data is stored in binary flat files.
- Record structures are defined using `struct` for consistency.
- File offsets and locks maintain atomic updates.

7. Security Features

- Encrypted password storage for all user types.
- One active session per user policy.
- Role-based access control ensures secure data handling.
- File-level locking and synchronization prevent data corruption.

8. Conclusion

The Banking Management System provides a robust simulation of a real-world banking environment using socket programming and operating system concepts. It integrates multiple user roles with secure, synchronized data access. The system ensures correctness, consistency, and concurrent handling of multiple clients efficiently.

9. Future Enhancements

- Integration with SQL databases for large-scale storage.
- Development of a graphical user interface (GUI) for ease of use.
- Implementation of real-time notifications for transactions.
- Use of SSL/TLS encryption for secure socket communication.