Bank Account Management System

Made By:-

Ayush Jain (Batch 61)

Piyush Bisht (Batch 62)

Abstract

This project implements a console-based Account Management System using the C programming language and MySQL database. The system focuses on two primary operations: creating new user accounts and managing existing account information. During account creation, the program collects and validates user inputs such as personal details, contact information, identification numbers, and account type. Strong validation techniques, including regular expressions and logical checks, ensure accurate and error-free data entry. All account records are securely stored in a MySQL database, enabling efficient retrieval and modification. The project demonstrates the integration of C with database systems, systematic input validation, modular programming, and the practical use of SQL for maintaining persistent account records. This provides a simple and reliable solution for basic account creation and data management.

Problem Definition

In many real-world scenarios, tasks and operations are performed manually, leading to inefficiency, errors, and inconsistent results. The absence of a structured, automated system creates challenges such as data inaccuracy, difficulty in retrieval, slow processing time, and poor user experience.

Because of these limitations, users or organizations face delays in decision-making, reduced productivity, and a lack of reliability in overall operations.

Therefore, there is a need for a systematic, automated, and user-friendly solution that can handle data securely, ensure accuracy, improve efficiency, and simplify operations.

The goal is to develop a software system that provides reliable data management; ease of use, faster execution, and improved performance compared to existing manual or unstructured methods.

Implementation

The Banking System is implemented using the C programming language along with MySQL as the backend database. The goal of the implementation is to provide a reliable system for creating and managing bank accounts using secure input handling, structured modules, and database connectivity.

1. Programming Language (C):
   The system is developed in C due to its efficiency and low-level hardware control. Key features include:
   - Structures (struct) for storing user/account data
   - File and database handling for persistent storage
   - Modular programming using multiple .c and .h files
   - Regex-based validation for user inputs
   - Menu-driven design using loops and conditionals
   - These techniques ensure reliability, clarity, and reusability.
2. Database Integration (MySQL)
   MySQL is used for storing account information.
   The implementation includes:
   - Creating tables for customer data
   - Inserting records during account creation
   - Login validation through SQL queries
   - Updating user information using UPDATE statements
   - Using MySQL C API functions such as:
     I. database connection
     II. executing SQL commands
     III. fetching results via MYSQL_RES and MYSQL_ROW
     IV. safely closing connections

   This provides structured and persistent data management.

3. System Modules
   A. Account Creation Module Handles:
      - Input collection (Name, Email, Phone, Password, DOB)
      - Regex validation
      - Auto-account number generation
      - Insertion into the MySQL database

   B. Account Management Module
      - Allows users to:
        I. Log in via account number & password
        II. View stored account details
        III. Modify fields (name, phone, email, password)

IV. Instantly reflect changes in MySQL

4. Input Validation
    - The system ensures correct and secure inputs using:
    - regex.h for email/phone validation
    - fgets() for safe input
    - Buffer-clearing techniques to remove unwanted newline characters
    - Logical validation such as ensuring age ≥ 18 based on DOB
    - This ensures no invalid or insecure data enters the database.
5. Error Handling
    - The implementation includes:
    - Detection of invalid entries
    - Clear messages for database connection failures
    - Retry loops instead of abrupt termination
    - Menu error handling for incorrect choices
    - This results in a stable, user-friendly application.
6. Code Organization
    - The project is modularized into separate files for clarity:

I. Account_management.c
II. account_creation.c
III. account_creation.h

This structure simplifies debugging and improves maintainability.

7. User Interface
    - The system uses a simple terminal-based user interface that allows:

I. Selecting operations
II. Entering data
III. Receiving immediate feedback
IV. Its straightforward design makes the application easy to test and operate.

Testing and Result

```
-----------------Welcome To Bank-----------------


01) Create New Account
02) Check Existing Account
03) Exit

Select an option: 1


-----------------Registration Starts!---------------------

Enter your Name: Shalin Verma
Enter your Gender: male
Enter your date of birth (yyyy-mm-dd): 2006-10-16
Enter your Aadhar Number :123467890456
Enter your PAN Number :IPUTY2312F
Enter your Phone Number: 4567123489
Enter your email: shalin.verma@gmail.com
Choose the type of Account (SAvings or Current): savings
Create Password (exactly 6 characters):

-----------------Registration complete!----------------------
-------------Your account-------------


                Account Number  : 879035590016
                Account Holder Name  : Shalin Verma
                Account Type    : Savings
                Account Balance : 0.00
```

```
01) Create New Account
02) Check Existing Account
03) Exit

Select an option: 2
Enter your Account Number :879035590016
Enter your Account Password :


------------------ ACCOUNT INFORMATION ------------------

1) Account Holder : Shalin Verma
2) Phone Number   : 4567123489
3) Email    : shalin.verma@gmail.com
4) Password   : ******
--- Balance   : 0.00   ---


A: Change Account Information
B: Deposit Money
C: Withdraw Money
X: Return to main menu

: b
Enter The Amount balance: 10000
```

```
01) Create New Account
02) Check Existing Account
03) Exit

Select an option: 2
Enter your Account Number :879035590016
Enter your Account Password :


------------------ ACCOUNT INFORMATION ------------------

1) Account Holder : Shalin Verma
2) Phone Number   : 4567123489
3) Email    : shalin.verma@gmail.com
4) Password   : ******
--- Balance   : 10000.00  ---


A: Change Account Information
B: Deposit Money
C: Withdraw Money
X: Return to main menu

: c
Enter the Withdraw Amount From balance: 5000

01) Create New Account
02) Check Existing Account
03) Exit

Select an option: 3
```

Conclusion

The Banking System project successfully demonstrates how C programming and MySQL can be combined to create a functional and secure account management application. The system allows users to create new accounts, store validated information, log in securely, and manage their details efficiently. Through the use of modular programming, structured data handling, regex-based validation, and database integration, the project achieves reliability, accuracy, and ease of use.

Overall, the implementation proves that even a console-based application can offer effective data management and user interaction when supported by proper backend organization and input validation techniques.

Future Work

Although the current system performs essential banking functions such as account creation and account management, several enhancements can further improve its usefulness and scalability:

- **Transaction Management:**
  Adding deposit, withdrawal, and fund transfer features to handle real-time transactions.
- **Balance Tracking:**
  Maintaining and updating account balances with transaction history.
- **Password Encryption:**
  Storing encrypted passwords instead of plain text for stronger security.
- **Admin Panel:**
  Introducing an admin login for managing all user accounts and monitoring system activity.
- **Graphical User Interface (GUI):**
  Converting the terminal-based interface into a GUI using GTK or a web interface for better usability.
- **Error Logging System:**
  Saving runtime errors or failed login attempts in log files for debugging and security audit.
- **Multi-Threading Support:**
  Allowing multiple operations simultaneously for future scalability.
- **Improved Validation Rules:**
  Enhancing regex patterns and adding more detailed checks (stronger passwords, better email verification, etc.)
- **Cloud Database Integration**
  Migrating the database to cloud-based MySQL servers for remote access.

References

A. MySQL C API Documentation
   MySQL Developer Reference Manual
   https://dev.mysql.com/doc/c-api/en/

B. GNU C Library Documentation
   Information on standard C functions, regex, string handling, and memory management
   https://www.gnu.org/software/libc/manual/

C. Regex Manual – POSIX Regular Expressions
   Pattern matching and validation rules
   https://pubs.opengroup.org/onlinepubs/9699919799/basedefs/regex.h.html

D. W3Schools – SQL Tutorial
   Basic SQL queries used in the project (SELECT, INSERT, UPDATE)
   https://www.w3schools.com/sql/

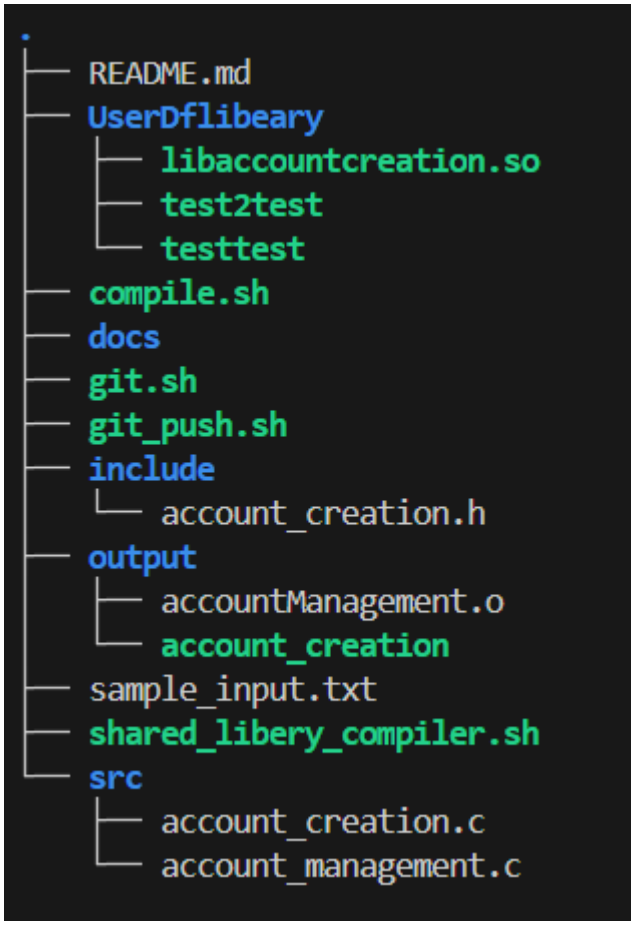
E. Linux Man Pages
   Reference for system-level functions used (termios, fgets, etc.)
   https://man7.org/linux/man-pages/

Appendix

A. Software Requirements

I. GCC Compiler
II. MySQL Server & MySQL Client
III. Linux Os
IV. Terminal/Command Prompt

B. Project Directory structure



Databases table Structure

Field	Type	Null	Key	Default	Extra
account_no	bigint	NO	PRI	NULL	
name	varchar(100)	NO		NULL	
gender	enum('Male','Female','Other')	NO		NULL	
date_of_birth	date	NO		NULL	
Aadhar_no	varchar(20)	NO	UNI	NULL	
Pan_no	varchar(20)	NO	UNI	NULL	
phone	varchar(15)	NO		NULL	
email	varchar(100)	YES		NULL	
balance	decimal(30,2)	NO		0.00	
account_type	enum('Savings','Current')	YES		Savings	
password_hash	varchar(100)	NO		NULL	
created_at	timestamp	YES		CURRENT_TIMESTAMP	DEFAULT_GENERATED
updated_at	timestamp	YES		CURRENT_TIMESTAMP	DEFAULT_GENERATED on update CURRENT_TIMESTAMP
no_of_user	int	NO	UNI	NULL	auto_increment

Compilation command

```
gcc -o UserDflibeary/libaccountcreation.so -fpic -shared src/account_creation.c -lmysqlclient
gcc -c src/account_management.c -o output/accountManagement.o -Iinclude
gcc -o UserDflibeary/test2test output/accountManagement.o -laccountcreation -LUserDflibeary
LD_LIBRARY_PATH=UserDflibeary ./UserDflibeary/test2test
```