SWIFT BANK

Fast, Simple and Secure Banking !!

Team Members:

Rupesh Saini.

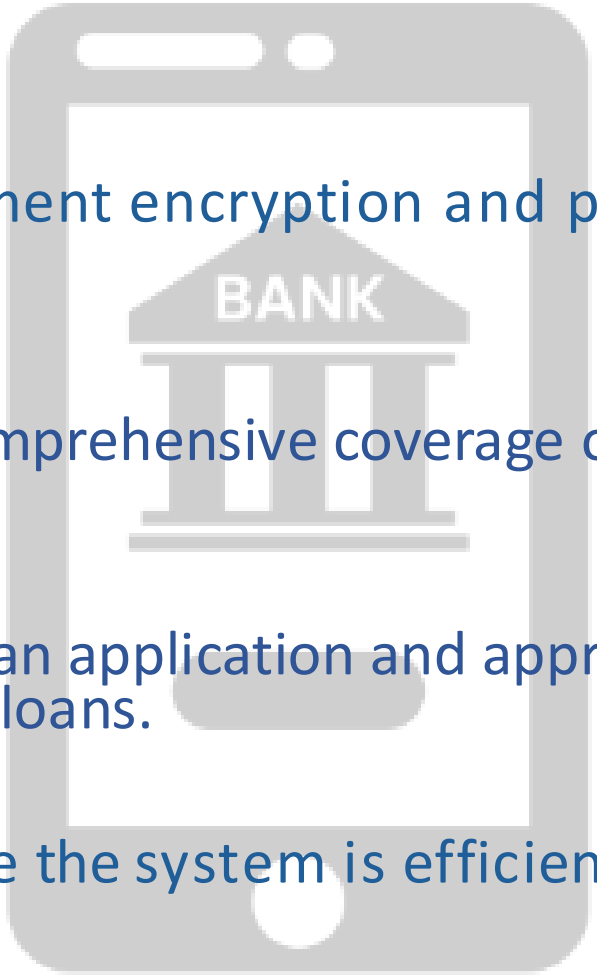
Shubhangi Tyagi.

Project Overview

- Swift Bank is a cutting-edge, secure digital banking solution focused on enhancing financial transaction.

- It offers a safe and seamless platform for users to manage accounts and conduct transactions.

Objectives



- **Secure Transactions**: Implement encryption and protection for financial transactions.
- **Insurance Services**: Provide comprehensive coverage options for home, life, motor, and general insurance.
- **Loan Management:** Simplify loan application and approval for home, car, education, business, and property-backed loans.
- **System Optimization**: Ensure the system is efficient and scalable to handle more users and transactions.

Functional Requirements

- **User Authentication:** Secure sign-up, log-in, and log-out.
- **Loan Management:** Users can view their loan applications.
- **Insurance Management:** Users can renew their insurance policies.
- **Balance Updates**: Update account balance after each transactions.

Non-functional Requirements

- **Compliance:** Ensure that the system complies with relevant financial and insurance regulations and standards.
- **Scalability:** The system should be able to scale to accommodate more users and transactions as demand increases.
- **Reliability**: The program validates inputs to ensure error-free operations.

System Architecture

- User Input: Users interact with the system through a menu-based interface.
- Processing Unit: The core logic handles user actions:
    - Deposit: Adds the specified amount to the account balance.
    - Withdraw: Validates balance before allowing withdrawal.

- Data Management: Updates and maintains account balance during transactions
- Output: Displays transaction results .
- System Exit: Exits the Program.

Technology Used

- **Programming Language**:
- **Platform**: Command-Line Interface (CLI)
- **Tools/Compiler**: GCC (GNU Compiler Collection)

Data Requirements

- **Type of Data:**

  User Information: Credentials such as username and password for secure access.

  Account Details: Includes account number, balance, and transaction history.

- **Data Sources:**

  Data is entered directly by users through the **Command-Line-Interface (CLI)**.

- **Data Storage and Format**:

  Data is stored and managed in structured **text-based files** (e.g.,.txt) for easy access and processing.

Assumptions and Constraints

- Users provide accurate data.
- Users have a stable internet connection.

Constraints:

- Accuracy depends on data quality.
- Latency may occur with large real-time data.

Timeline and Milestones

- **Phase 1: Requirement Gathering and Design** (4th Nov to 7th Nov)
  Milestone 1: Completed requirement gathering and finalized the design.

- **Phase 2: Development** (8th Nov to 16th Dec)
  Milestone 2: Completed development with all core features.

- **Phase 3: Testing** (17th Dec to 18th Dec)
  Milestone 3: Successfully completed testing, with all tests passed.

- **Phase 4: Review and Feedback**(19th Dec)
  Milestone 4: Incorporated feedback and made necessary adjustments

- **Phase 5: Final Presentation and Submission** (20th Dec)
  Milestone 5:

References

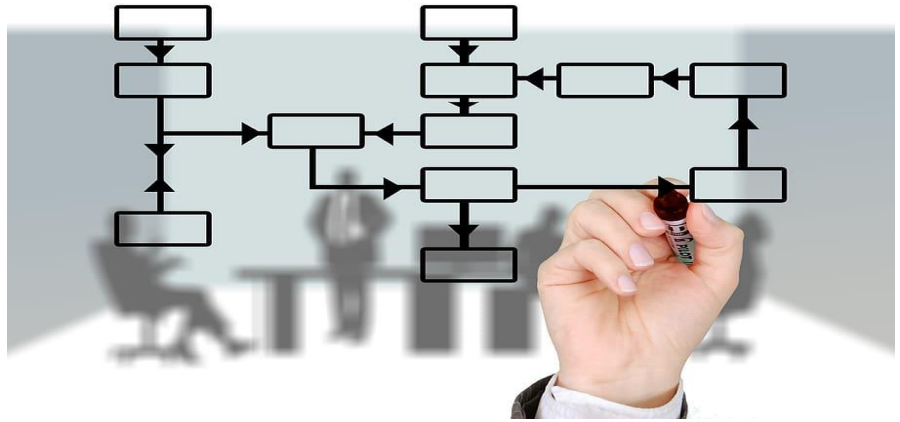
- Let Us C by Yashavant Kanetkar.

Additionally, for websites ,

- GeeksforGeeks

- TutorialsPoint

Appendices



- ***Flowchart For USER Login***

Start → User enter credentials → Validate credentials→ Success

- ***Activity Diagram***

User Activity:

Log in → Choose transaction → Enter details → Confirm → Transaction completed.