Banking Application in Java

(Console Based Application)

High Level Design

Domain : JAVA

Creator : Mohamed Tharif

Date : 1st Feb 2024

Document Version Control :

Date Issued	Version	Description	Author
1st Feb 2024	1.0	Initial Version	B Mohamed Tharif

Abstract :

The Banking Application is a console-based Java program designed to provide basic banking functionalities such as account creation, transaction management, and data persistence through file handling. The application is structured around key classes, each serving a specific role in the banking system.

Key Classes:

1. **Banking Class:**
    - Responsible for managing the overall flow of the application.
    - Orchestrates interactions between different components.
    - Handles user input, decision-making, and control flow.
2. **NewAccount:**
    - Facilitates the creation of a new bank account.
    - Gathers user details, validates inputs, and creates a new entry in the CSV file.
    - Ensures the integrity of user data during account creation.
3. **ExistingAccount:**
    - Manages existing bank accounts.
    - Performs withdrawal and deposit operations.
    - Provides functionality for updating account details.
4. **CheckExisting:**
    - Verifies the existence of a bank account.
    - Implements logic to check for account presence in the CSV file.
    - Supports decision-making in account creation and transaction processes.
5. **WriteAccount(WriteData and UpdateData):**
    - Manages reading and writing data to files.
    - Ensures the persistence of account information.

- Implements file handling operations for creating, updating, and retrieving data.

The architecture emphasizes modularity and separation of concerns, allowing for easy maintenance and extensibility. The application utilizes CSV files for data storage, enhancing portability and simplicity. Components are designed to interact seamlessly, creating a cohesive system that caters to basic banking operations.

Security considerations include the handling of sensitive data, and error handling mechanisms ensure robustness against unforeseen issues. The system provides a user-friendly console interface for input and feedback.

Introduction

What is High-Level Design Document?

The goal of this HLD or a high-level design document is to add the necessary detail to the current project description to represent a suitable model for coding. This document is also intended to help detect contradictions prior to coding and can be used as a reference manual for how the modules interact at a high level.

The HLD will:

Present all of design aspects and define them in

detail Describe all user interfaces being

implemented Describe the hardware and software

interfaces Describe the performance requirements

Include design features and architecture of the project

List and describe the non-functional attributes such as security, reliability, maintainability, portability, reusability, application compatibility. resource utilization, serviceability

Scope:

The HLD documentation presents the structure of the system, such as database architecture, application architecture (layers), application flow (Navigation), and technology architecture. The HLD uses non-technical to mildly technical terms which should be understandable to the administrators of the system.

General Description

Definitions

Term	Description
BankingSystem	Banking Operations
Database	Collection of the Information
IDE	Integrated Development Environment
UI	User Interface

Product Description

Banking interface is basic command prompt which allows user to perform various task such as deposit, withdraw, transfer, show transactions and show balance

Problem Statement

The program prompts the user to enter their user ID and PIN to access the functionalities of an Banking. Upon successful authentication, the user gains access to various operations commonly found in an Banking. The available operations in this project include viewing transaction history, making withdrawals, deposits, transfers, and quitting the program.

Proposed solution

Using all the standard techniques used in the life cycle of a Softwaredevelopment by using UML Diagrams such as constructing use case diagram, class diagram, sequence diagram and activity diagram to perform the required operations.

Further improvements

This project can be further improved by adding an swing interface to access in local machine or by developing a web interface and deploy the software as web app.

Data requirements

Data requirement completely depend on our User using the interface. This program stores information entered by user to store and store it in the text file.

Tools used

The Banking interface is developed using Java programming language, along with the tools such as GitHub for version control and collaboration, and Visual Studio Code as the integrated development environment (IDE). Java provides the necessary features and libraries to implement the core functionality of the Banking interface. GitHub enables the team to manage and track code changes, collaborate on the project, and ensure version control. Visual Studio Code offers a user-friendly and feature-rich environment for writing, testing, and debugging the Java code. Together, these tools facilitate the development, management, and maintenance of the Banking interface project.

Hardware Requirements

- A computer system capable of running the Java development environment and the chosen IDE (such as Visual Studio Code). The system should meet the minimum system requirements for the selected software..
- Minimum 1.10 GHz processor or equivalent.
- Between 1-2 GB of free storage
- Minimum 512 MB of RAM
- 3 GB of hard-disk space

Constraints

The front-end must be user friendly and should not need any one to have any prior knowledge in order to use it.