JAVA ATM INTERFACE

High Level Design

Domain: Java

Stuti Radhanpura

Date: 25.04.2024

Document Version Control

Date issued	Version	Description	Author
25/04/2024	1.1	First Draft	Stuti Radhanpura

Contents

Abstract

The ATM interface provides an abstraction for interacting with an Automated Teller Machine (ATM). It serves as a medium for users to perform various banking operations, such as account creation, login, balance inquiry, cash withdrawal, fund transfer, and transaction history retrieval. The interface abstracts the underlying complexities of the ATM system and presents a user-friendly interaction platform.

Key Features and Functionalities:

1. Account Creation: Allows users to create a new bank account by providing essential details such as name, age, and initial deposit amount.

2. Login: Authenticates users by verifying their credentials, such as user ID and PIN, to ensure secure access to their accounts.

3. Balance Inquiry: Enables users to check the available balance in their accounts, providing them with up-to-date financial information.

4. Cash Withdrawal: Facilitates the withdrawal of cash from the user's account, ensuring proper validation and deduction of the requested amount from the account balance.

5. Fund Transfer: Enables users to transfer funds from their account to another account within the same bank or to external accounts in other banks, ensuring secure and accurate transactions.

6. Transaction History: Allows users to view a record of their recent transactions, including details such as date, time, type of transaction, and amounts involved.

The ATM interface abstracts the underlying implementation details, such as data storage, security protocols, and transaction processing. It provides a simplified and user-friendly experience while ensuring the integrity and security of banking operations. The interface serves as a bridge between users and the ATM system, enabling them to carry out essential banking tasks conveniently and efficiently.

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
ATM	Automated Teller Machine
Database	Collection of the Information
IDE	Integrated Development Environment
UI	User Interface

Product Description

ATM 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 ATM. Upon successful authentication, the user gains access to various operations commonly found in an ATM. 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 Software development 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 ATM 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 ATM 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 ATM interface project.

Hardware Requirements

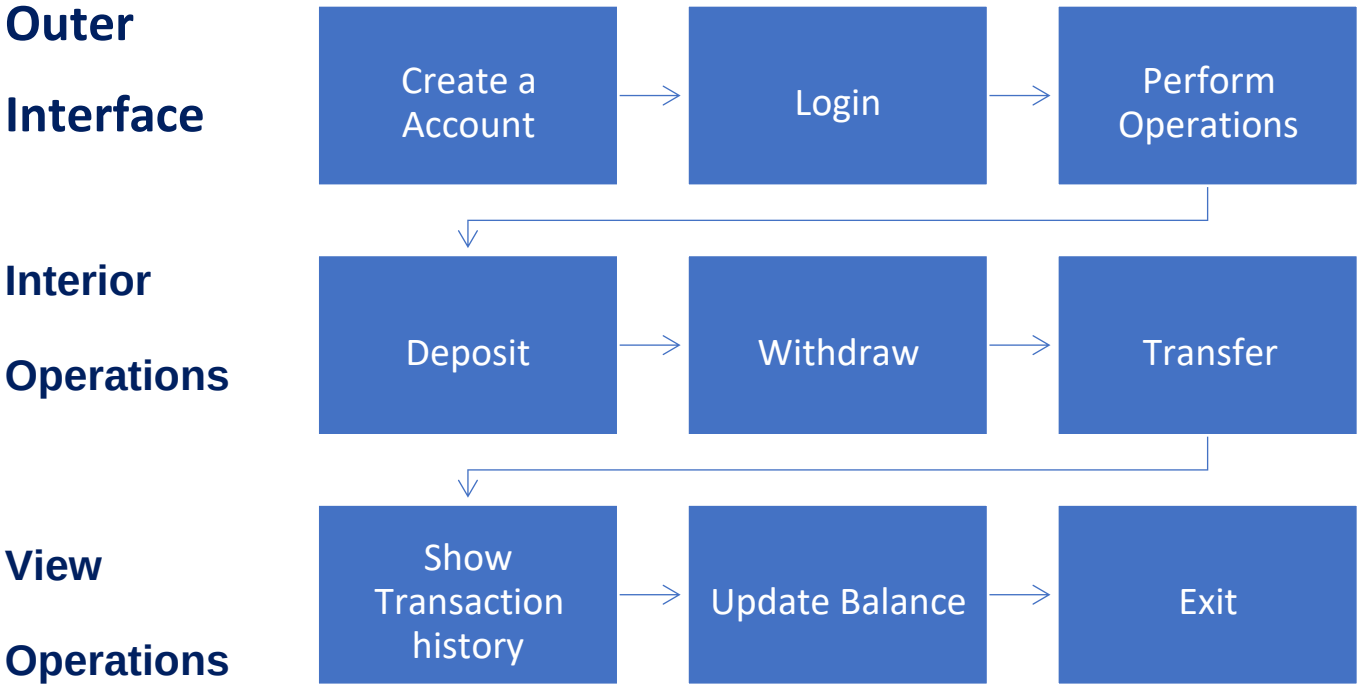
- A computer system capable of running the Java development environment and the chosen IDE (such as Visual Studio Code). The system should meet the minimum system requirements for the selected software..
- Minimum 1.10 GHz processor or equivalent.
- Between 1-2 GB of free storage
- Minimum 512 MB of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

Design Details

Process Flow



Event Log

User Login:

Event: User successfully logged in.

Description: The user entered their user ID and PIN, and the system verified the credentials. The user has gained access to the ATM interface functionalities.

View Account Balance:

Event: User requested to view account balance.

Description: The system retrieves the account balance associated with the logged-in user and displays it on the screen.

Withdrawal:

Event: User initiated a withdrawal.

Description: The user entered the withdrawal amount, and the system verified the availability of funds. If the balance is sufficient, the requested amount is deducted from the account balance and dispensed as cash.

Deposit:

Event: User initiated a deposit.

Description: The user inserted cash or a check into the ATM. The system verifies the authenticity and validity of the deposit and adds the deposited amount to the account balance.

Transfer:

Event: User initiated a transfer.

Description: The user entered the recipient's account number and the transfer amount. The system validates the recipient's account and verifies the availability of funds. If successful, the specified amount is transferred from the user's account to the recipient's account.

Transaction History:

Event: User requested transaction history.

Description: The system retrieves and displays the transaction history for the user's account, including details such as date, type of transaction (withdrawal, deposit, transfer), and transaction amount.

Incorrect PIN Entry:

Event: User entered an incorrect PIN.

Description: The system compares the entered PIN with the stored PIN for the user's account. If the entered PIN does not match, an error message is displayed, and the user is prompted to re-enter the PIN.

Insufficient Funds:

Event: User attempted a transaction with insufficient funds.

Description: The system checks the account balance to ensure there are enough funds for the requested transaction. If the balance is insufficient, an error message is displayed, and the transaction is not processed.

Error Handling

Error handling is a crucial aspect of the ATM interface to ensure smooth operation and provide appropriate feedback to users in case of errors or exceptional situations. The system should be designed to handle various types of errors effectively.

Firstly, the interface should handle invalid user login attempts. If a user enters an incorrect user ID or PIN, the system should display an error message indicating the invalid credentials and prompt the user to retry.

Secondly, error handling should be implemented for situations where there are insufficient funds. When a user requests a withdrawal or initiates a transfer, the system should check if their account balance is sufficient. If not, an error message should be displayed, informing the user about the insufficient funds and preventing the transaction from proceeding.

Additionally, the interface should perform proper validation and error handling for user input. This includes validating fields such as withdrawal amount, transfer amount, or account details. If any input is invalid or out of bounds, the system should display appropriate error messages to guide the user in providing correct information

Performance

he interfaces should respond quickly to user inputs, providing near-instantaneous feedback. Actions such as button presses, menu navigation, and transaction processing should have minimal latency, ensuring a seamless user experience.

Reusability

The code written and the components used should have the ability to be reused with no problems.

Conclusion

All in all, overall project architecture, design details, used technologies and performance were explained in detail. The MITVP will give the traffic volume predictions instantly and has the potential to help various government organizations, agencies and etc.