JAVA ATM INTERFACE

Low 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

Introduction

What is Low-Level Design Document?

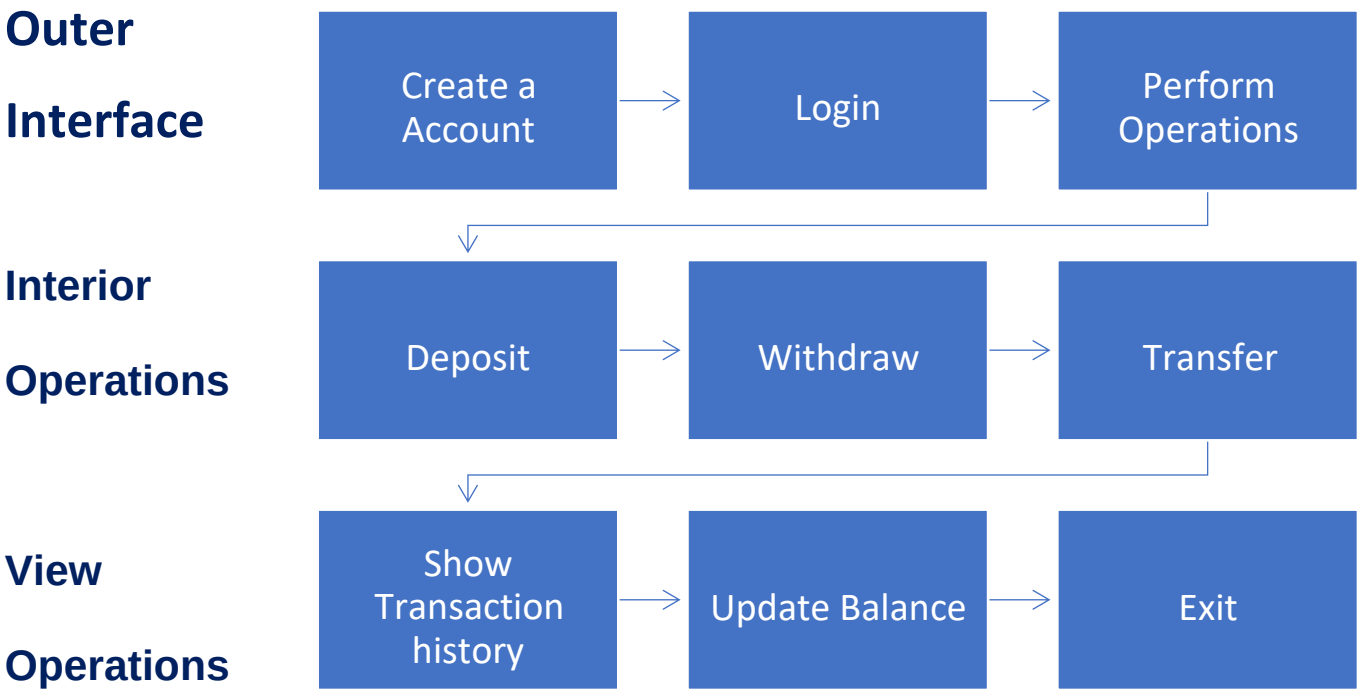
The goal of LLD or a low-level design document is to give the internal logical of the actual program code for Java Atm Interface. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, the constraints under which it must operate and how the system will react to external stimuli.

The main objective of the project is to Console prompt to simulate the functionality of an Automated Teller Machine (ATM).

Scope

Low-level design (LLD) is a component-level design process that follows a step-by-step refinement process. This process can be used for designing data structures, required software architecture, source code and ultimately, performance algorithms. Overall, the data organization may be defined during requirement analysis and then refined during data design work.

Architecture



Class Description

ATM Interface

First opening prompt allows user to create a new account and Login to their respective account.

ATM

This class contains methods to create an account and perform login operations. It also includes necessary error handling mechanisms to check the validity of the account and returns an appropriate success message.

Account Holder

This class contains a statically linked list that stores all the bank account objects along with their required details, such as first name, last name, age, user ID, PIN, and balance. Additionally, this class handles the task of validating the entered details during login and transfer operations. It returns a Boolean value of true if the details are valid; otherwise, it returns false. Furthermore, it is responsible for retrieving the account information if the details are valid.

Account

This class encapsulates various operations related to banking, including deposit, withdraw, transfer, show transactions, and show balance. These operations allow users to interact with their accounts and manage their funds effectively which provide users with the necessary functionality to manage their funds, perform transactions, and keep track of their financial activities.

Bank

This class serves as a blueprint for creating account objects in a banking system. It holds essential information about an account holder, including their first name, last name, age, user ID, PIN, and balance. By encapsulating all this information within the account object, the class provides a convenient way to create, access, and manage account-related details. It allows for proper organization and manipulation of account information within the banking system.

Bank Data

This class encompasses methods for file handling operations, specifically for storing account details in a text file, reading information from the text file, and updating the balance. By incorporating these file handling methods, the class enables the persistence of account details in a text file. It ensures that the data is stored securely and can be accessed or modified as required. Additionally, it provides the functionality to read and update the balance, allowing for accurate financial transactions and account management.

Unit cases

Test Case Description	Pre-Requisite	Expected Result
Verify whether the Account Exists when user logins	Account has to be created before login	Prompts user with error if account not and asks them to create new account
Check that the deposit operation increases the account balance by the deposited amount.	Application accessible	Balance has to updated accordingly and negative deposits should be caught
Verify that the withdrawal operation deducts the specified amount from the account balance.	Application is accessible	Balance has to be withdrawn from the particular account and insufficient balance error has to be handled
Verify fund transferred correctly from one user to another	Application is accessible	Ensure that funds are transferred correctly from one account to another. Test for invalid recipient account and verify appropriate error handling.
Verify account Balance display	Application is accessible	Ensure that the account balance is displayed correctly. Verify that the balance is updated accurately after each transaction.
Verify Valid inputs	Application is accessible	Test for invalid data input and verify proper error handling and error messages.