Advance Java Case Study –
Bank ATM
Management System

Advanced Java Case Study – Bank ATM Management System

Table of Contents

Contents

Advanced Java Case Study – Bank ATM Management System

Welcome!

Welcome to the Advance Java Case Study exercise for Bank ATM Management System in a bank.

This case study exercise includes six user stories. You need to perform the activities mentioned in the user stories and submit your results.

Before working on the case study exercise, take look at the Architecture Overview for Bank ATM Management System in a bank.

Lab Set up :- It is expected that you use the set up you have created while going through your learning journey for each of the individual courses.

Architecture Overview

1. Introduction – Architecture Overview

1.1 Purpose

The purpose of the Architecture Overview, as represented by the diagram below, is to:

- Communicate a conceptual understanding of the target IT system
- Provide a high-level shared vision of the architecture and scope of the proposed IT system
- Explore and evaluate alternative architectural options
- Enable early recognition and validation of the implications of the architectural approach
- Facilitate effective communication between different communities of stakeholders and developers
- Facilitate orientation for new people who join the project

1.2 Scope

This Architecture Overview is documented for Bank ATM Management System in a bank application.

1.3 Intended Audience

This document is intended for Developers, Architects, Technical Advisory Board, Project Manager, and Designers.

1.4 Overview

The Architecture Overview diagram represents the governing ideas and candidate building blocks of an IT system and enterprise architecture. It provides an overview of the main conceptual elements and relationships in an architecture, including candidate subsystems, components, nodes, connections, data stores, users, and external systems.

2. Architectural Goals

2.1 Strategic Drivers

The strategic drivers are as follows:

- Using a layered architecture, which ensures parallel development of multiple layers. Each layer is independent of other layers, whichmeans that we can change the technology of one layer without effecting other layers. For example, we can easily replace JSP in the presentation layer with ReactJS.
- Ensures low coupling between the components. Ensuring high cohesion, which enables that the responsibilities of a given element are strongly related and highly focused.
- Ensure separation of concerns and distribution of responsibilities.

2.2 Architectural Drivers

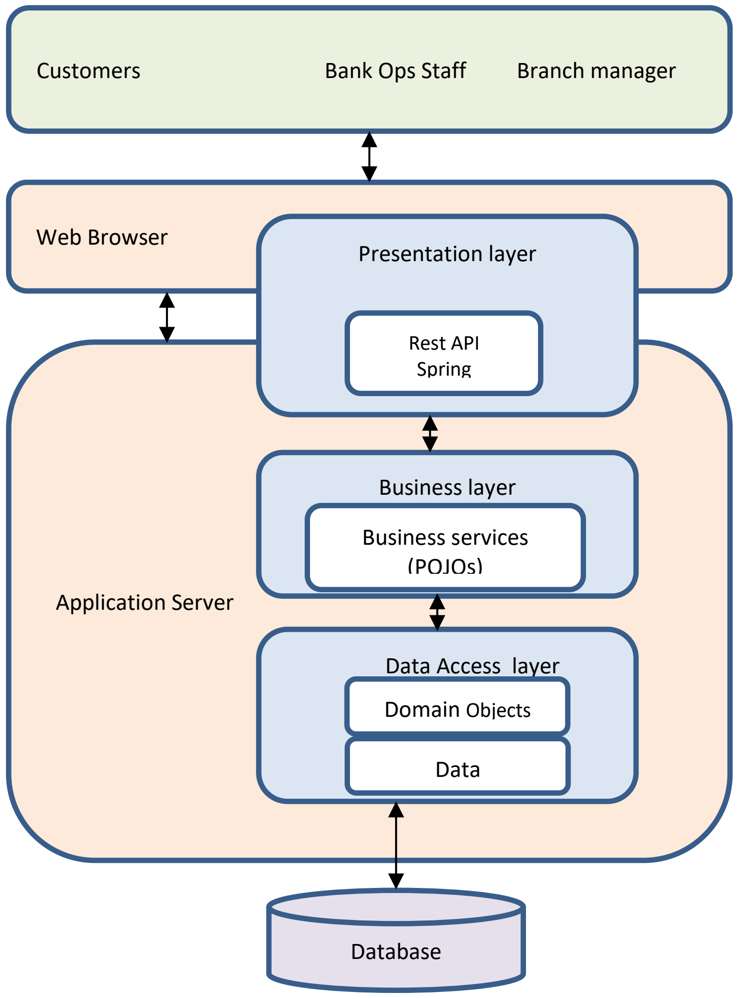
2.2.1 Technology Platform

Development platform	A Java-JEE/Stack built around: • Java Enterprise Edition • Spring Framework • Spring Boot • Spring Data MySQL • Eclipse or STS • JUnit
Application server	Apache Tomcat or any application server
Relational database server	MySQL

3. Architecture Overview

3.1 Application Layers

The diagram given below will provide a clear idea of the different layers of the application.



Advanced Java Case Study – Bank ATM Management System

Presentation Layer:

The presentation layer is responsible for the interaction with outside actors (via the User Interface).

The technology used in the presentation layer is html/jsp coupled with Spring Boot and Spring Rest.

Service Layer:

Service layer consists of business service components which implement the business logic of the application.

Data Access Layer:

This layer is responsible for interaction with the database and performs database-specific operations

(CRUD).

Persistence Layer:

This layer is not actually an individual layer. It is clubbed with Data Access layer and it uses Data MySQL to retrieve, save, and update data in the database.

Case Study Exercise

Case Study - Bank ATM Management System in a bank

General Description

The Bank ATM Management System in a bank application is used automating the daily tasks of a customer. Different types of services are made available through ATMs and are connected with the bank accounts. The customers can access the ATM with their debit cards authenticated by 8 digit card no, validity of card and cvv. 4 digit Pin is used to authenticate the operation.

User Stories

User Story 1.1: View Home Page

Description

As a user, I want to view the application home page before I am prompted to give my card details. I log in to have easy access to the features of the application and can quickly see relevant information.

The home page should contain the following:

1. Name of application
2. Reserved area for functions. Place holders should be placed instead of actual links for functions. Functions will be added through succeeding user stories.

3. Reserved area for default views. Nothing to be displayed for now. Views to be displayed will be determined by succeeding user stories.

Out-of-scope

Actual content of reserved areas is out of scope and will be specified by other stories.

Dependencies

None

Acceptance Criteria

1. All actions are routed through the home screen.
2. Home page displays application name.
3. Home page has visible areas for functions and default views.

User Story 1.2: Withdrawal

Description

As a user, I want to withdraw some money , subject to the adequate balance in the account.

Functional Requirements

Customer name to be displayed after the customer is recognized. At the end of transaction, balance is displayed, if user wishes to see. A transaction summary is displayed on screen, irrespective of success or failure of operation.

UI

1. User should receive feedback that a transaction was success or failure.
2. User should receive feedback if a transaction failed for any of the the following reasons:
   a. Not adequate balance – User should be alerted
   b. Missing required fields – User should be alerted as to which fields are missing
   c. Invalid field values – User should be alerted as to why a field value is wrong
3. In case of failure, a generic error message should be displayed

Validation

1. A transaction can be done in multiples, within the daily transaction limits of number/amount.
2. Each transaction will have a transaction number.

Out-of-scope

Dependencies

None

Acceptance Criteria

1. User can carry out transactions as it passes validation.
2. All functional requirements are met.

User Story 1.3: Deposit money

Description

As a user, I want to deposit some money, balance to be added in the account.

Functional Requirements

Customer name to be displayed after the customer is recognized. At the end of transaction, balance is displayed, if user wishes to see. A transaction summary is displayed on screen, irrespective of success or failure of operation.

UI

4. User should receive feedback that a transaction was success or failure.
5. User should receive feedback if a transaction failed for any of the following reasons:
    a. Not adequate balance – User should be alerted
    b. Missing required fields – User should be alerted as to which fields are missing
    c. Invalid field values – User should be alerted as to why a field value is wrong
6. In case of failure, a generic error message should be displayed

Validation

3. A transaction can be done in multiples, within the daily transaction limits of number/amount.
4. Each transaction will have a transaction number.

Out-of-scope

1. N/A

Dependencies

None

Acceptance Criteria

1. User can carry out transactions as it passes validation.
2. All functional requirements are met.

User Story 1.4: Transfer money/Payments

Description

As a user, I want to transfer money to other accounts after an authentication process.

Functional Requirements

All transfer fields should be editable. Validation rules for customer fields from other user stories must still be followed

UI

1. Transfer to be viewed/edited before confirmation.
2. All transaction records to be stored, with a transaction id.
3. User should receive feedback if transfer was successful.
4. User should receive feedback if a transaction failed for the following reasons:
    a. User should be alerted if transfer is to a blocked account.
    b. Not adequate balance

    c. Missing required fields – User should be alerted as to which fields are missing
    d. Invalid field values – User should be alerted as to why a field value is wrong

Out-of-scope

1. N/A

Dependencies

Other stories

Acceptance Criteria

1. All functional requirements are fulfilled.

User Story 1.5: Pin change Details

Description

As a user, I want to change my pin details , subject to the authentication in the account.

Functional Requirements

Customer name to be displayed after the customer is recognized. At the end of transaction. A transaction summary is displayed on screen, irrespective of success or failure of operation.

UI

1. User should receive feedback that a transaction was success or failure
2. User should receive feedback if a transaction failed for any of the the following reasons:
    a. No authentication – User should be alerted
    b. Missing required fields – User should be alerted as to which fields are missing
    c. Invalid field values – User should be alerted as to why a field value is wrong
3. In case of failure, a generic error message should be displayed

Validation

4. Each transaction will have a transaction number.

Out-of-scope

2. N/A

Dependencies

None

Acceptance Criteria

1. All functional requirements are met.

User Story 1.6: View Account balance and transaction History

Description

As a user, I want to view the transaction history for a duration, so that a summary report is available.

Only total statistics per account will be displayed for chosen customer.

Functional Requirements

UI Flow

1. User has card based authentication
2. User enters gives from and to dates
3. New screen shows total statistics per request
4. User can choose to return to Home Page or Step 4

UI

1. As per the user story

Validation

1. When entering date range, user cannot enter an end date that occurs before start date

Advanced Java Case Study – Bank ATM Management System

Out-of-scope

Breakdown of account wise details