Advance Java Case Study – Account Management System

Advanced Java Case Study – Bank Account Management System

Table of Contents

Contents

Welcome! 3

1. **Introduction – Architecture Overview** 4
2. **Architectural Goals** 5
3. **Architecture Overview** 7

Case Study - Account Management System in a bank 9

General Description 9

User Stories 9

User Story 1.1: View Home Page 9

User Story 1.2: Apply for opening an Account 10

User Story 1.3: Check the status of Account application 11

User Story 1.4: View/Edit Customer own Profile 12

User Story 1.5: Add Customer Transaction 13

User Story 1.6: View Account History 15

Advanced Java Case Study – Bank Account Management System

Welcome!

Welcome to the Advance Java Case Study exercise for Account 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 Account 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.

Advanced Java Case Study – Bank Account Management System

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 Account 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.

Advanced Java Case Study – Bank Account Management System

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, which means 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.

Advanced Java Case Study – Bank Account Management System

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

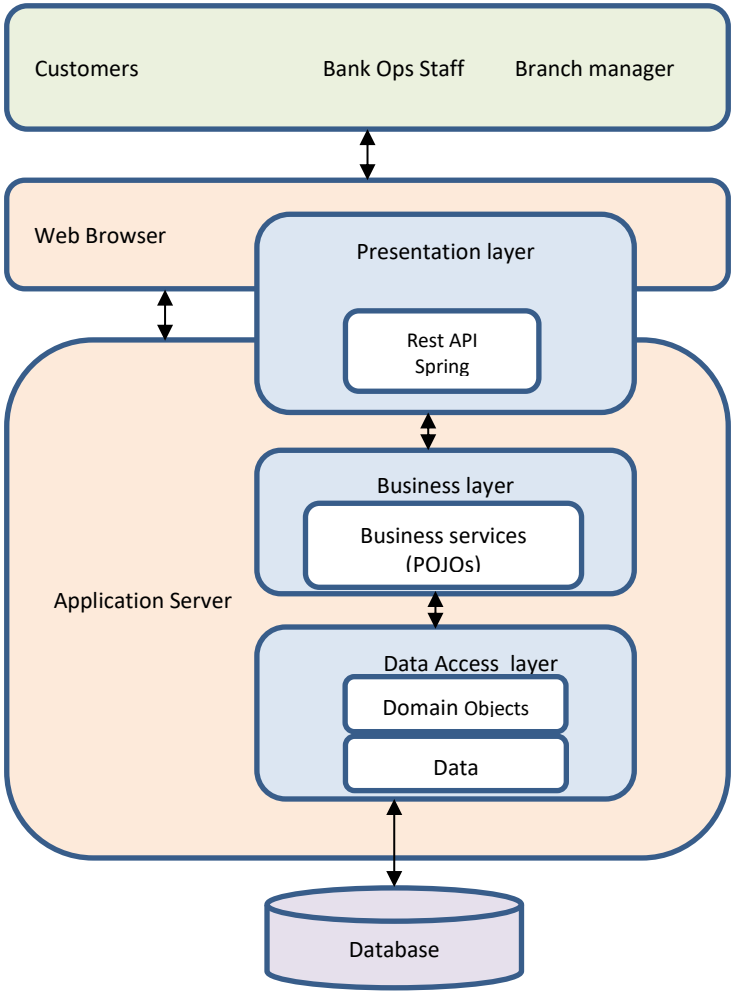
Application server	Apache Tomcat or any application server
Relational database server	MySQL

Advanced Java Case Study – Bank Account Management System

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 Account 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.

Advanced Java Case Study – Bank Account Management System

Case Study Exercise

Case Study - Account Management System in a bank

General Description

The Account Management System in a bank application is used by a Bank Manager to track Customers. Different types of accounts are provided and tracked on parameters, such as transactions per month per customer, Monthly Average Balance (MAB), total money deposited/withdrawn in a month, interest accrual per quarter. Based on this information, the manager wants to categorise the customers for business development.

User Stories

User Story 1.1: View Home Page

Description

As a branch Manager, I want to view the application home page when I log in so that I have easy access to the features of the application and can quickly see relevant information.

The manager should see the home page of the application after entering the web application's URL.

The home page should contain the following:

1. Name of application

Advanced Java Case Study – Bank Account Management System

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. Base application URL leads to home page.
2. Home page displays web application name.
3. Home page has visible areas for functions and default views.

User Story 1.2: Add Customer

Description

As a bank Manager, I want to add a customer so that I can manage customer information.

Functional Requirements

Customer profile fields should include the following:

Field	Type	Required?
First Name	String	Yes
Middle Name	String	No
Last Name	String	Yes
Birth Date	Date	Yes
Number of accounts	Numeric	Yes

In addition, the system should generate a BRANCH for each successfully added customer.

Advanced Java Case Study – Bank Account Management System

UI

1. Customer entry form (KYC) should have its own screen.
2. User should receive feedback that a customer was successfully added.
3. User should receive feedback if a customer was not added for the following reasons:
    a. Customer already exists – 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
4. If customer was not added for some other reason, a generic error message should be displayed

Validation

1. A customer cannot be entered into the system twice. Customer identity is based on first name, middle name, last name, and birth date. Customers with all four identical fields are considered to be the same customer.
2. Birth date should not be later than current date.

Out-of-scope

1. Legal age validation

Dependencies

None

Acceptance Criteria

1. User can add customer to the system as long as it passes validation.
2. All functional requirements are met.

User Story 1.3: Search Customer

Description

As a Branch Manager I want to search for customers in the system so that I can select customer to manage.

Advanced Java Case Study – Bank Account Management System

Functional Requirements

Search by first name, last name, position or any combination of the three. Search result should match all supplied search criteria (AND).
UI

1. One field for each search criteria
2. One Search button
3. One Clear button to clear search criteria
4. Executing a search should not clear search criteria
5. Search results should display customer UID plus all fields specified in Story 1.2
6. If no matching results are found, “0 results found” should be displayed

Out-of-scope

N/A

Dependencies

Story is dependent on Story 1.2: Add Customer, unless customer entities are already in the database.

Acceptance Criteria

1. User should be able to search for customers
2. All other functional requirements are fulfilled

User Story 1.4: View/Edit Customer Profile

Description

As a Branch Manager, I want to view and edit customer profiles so that I can change customer details when needed.

Functional Requirements

All customer fields except BRANCH should be editable. Validation rules for customer fields from Story 1.2 must still be followed
UI

Advanced Java Case Study – Bank Account Management System

1. Customer to be viewed/edited must be selected from search results (see Story 1.3)
2. View/Edit Customer form must be in its own screen.
3. View/Edit form fields must be pre-populated with customer information.
4. All customer fields should be visible, including CID.
5. CID should only be displayed. It should not be editable.
6. User should receive feedback if customer was successfully edited.
7. User should receive feedback if a customer was not updated for the following reasons:
    a. Identical customer found – User should be alerted if editing a customer would make it identical to an existing customer
    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
8. If customer was not edited for some other reason, a generic error message should be displayed.
9. If customer was not edited, form fields should not be reset.

Out-of-scope

1. Legal age validation

Dependencies

Story 1.3

Acceptance Criteria

1. User can select a customer to view/edit from search results.
2. User can view/edit a customer.
3. All functional requirements are fulfilled.

User Story 1.5: Add Customer Interest Details

Description

As a Branch Manager, I want to add interest compounded yearly for a customer so that I can track a customer's details in the system.

It is possible for a customer to have multiple accounts and each account will have separate interest calculation. For example, a savings

Advanced Java Case Study – Bank Account Management System

account will have a different rate from a current account. A Term Deposit will have a duration of months/days as the term of the deposit. The date of maturity should be calculated and displayed with the maturity amount. A penalty will be imposed for premature termination. These should be recorded in the system. Customer details include the account type (savings, current, Cumulative, Term, Loan etc.), statistics , and description should be available.

Functional Requirements

Assumptions

1. It will be up to the branch Manager to define the list of customers for categorizing them .

Fields

Field	Type	Required?
Account	String	*See below*
Type	Numeric	*See below*
Description	String	*See below*
Date	Date	Can use Java Date class

Types of information

1. Customer details – This is basic info. Only one entry per customer per account can be added. It is possible for the value to be zero) or upto 11. Description is optional.
2. Transaction details – A customer can deposit/withdraw not more than 6 times in a month. Extra transactions are charged at Rs 5 per transactions. It is possible for the value to be zero or more. Description is optional.
3. Any account that has not been operated for last 6 months.
4. A customer can have multiple accounts. Description is required.
5. Any Other info – This covers anything not covered above. More than one entry is allowed. Values can be zero also. Description is required.

Validation

1. See section above for validation rules for the different accounts/customers.

Advanced Java Case Study – Bank Account Management System

Out-of-scope

1. Transaction amount
2. Calculating total interest earned in a financial year

Dependencies

Story 1.3

Acceptance Criteria

1. User can add score details as per functional requirements

User Story 1.6: View Interest History

Description

> *As a Branch Manager, I want to view the interest history of a customer so that I can generate a report on how much an customer earns for every account.*

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

Functional Requirements

UI Flow

1. User searches for customer
2. User enters account id
3. New screen shows total statistics per month for each year
4. User can choose to return to Home Page or Step 4

UI

1. Total per account should be shown. No breakdown as this is done in another story.
2. If no details for a account can be found, that account is not displayed. However, in the case where details have zero value, that account should still be displayed.

Validation

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

Advanced Java Case Study – Bank Account Management System

Out-of-scope

Breakdown of account wise details