A picture containing dome, honeycomb, vector graphics, ball

Description automatically generated

**Java Case Study – Employment System**

Table of Contents

Contents

[Welcome! 3](#_Toc59109793)

[**1.** **Introduction** 4](#_Toc59109794)

[**2.** **Architectural Goals** 5](#_Toc59109795)

[**3.** **Architecture Overview** 7](#_Toc59109796)

[Case Study - Employment System 9](#_Toc59109797)

[General Description 9](#_Toc59109798)

[User Stories 9](#_Toc59109799)

[**User Story 1.1: View Home Page** 9](#_Toc59109800)

[**User Story 1.2: Add Employee** 10](#_Toc59109801)

[**User Story 1.3: Search Employees** 11](#_Toc59109802)

[**User Story 1.4: View/Edit Employee Profile** 12](#_Toc59109803)

[**User Story 1.5: Add Employee Compensation Details** 13](#_Toc59109804)

[**User Story 1.6: View Compensation History** 15](#_Toc59109805)

[**User Story 1.7: View Compensation Breakdown for a Month** 16](#_Toc59109806)

[**User Story 1.8: Edit Employee Compensation Details** 17](#_Toc59109807)

# Welcome!

Welcome to the Java Case Study exercise for Employment System.

This case study exercise includes eight 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 Employment System.

Architecture Overview

# **Introduction**

* 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. **Scope**

This Architecture Overview is documented for Employment System application.

* 1. **Intended Audience**

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

* 1. **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.

# **Architectural Goals**

* 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 Spring MVC in the presentation layer with another MVC like JSF.
    - 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.
  1. **Architectural drivers**

**2.2.1 Technology Platform**

|  |  |
| --- | --- |
| Development platform | A Java-JEE/stack built around:   * HTML/CSS * Java Enterprise Edition   + Java Server Pages   + JSTL   + JavaBeans * Spring Framework   + Spring MVC   + Spring Data JPA * Eclipse or IntelliJ * JUnit |
| Application server | Apache Tomcat or any application server |
| Relational database server | MS SQL or any RDBMS |

# **Architecture Overview**

* 1. **Application Layers**

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

Diagram

Description automatically generated

**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 Spring MVC.

**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 uses Spring Data to retrieve, save, and update data in the database.

Case Study Exercise

# Case Study - Employment System

## General Description

The Employment System application is used by an HR Manager to track employee compensation. Different types of compensation are tracked such as monthly salary, bonuses, commission, etc.

## User Stories

### **User Story 1.1: View Home Page**

#### Description

As an HR 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 user 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
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 Employee**

#### Description

As an HR Manager, I want to add an employee so that I can manage employee information.

#### Functional Requirements

Employee 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 |
| Position | String (free form) | Yes |
|  |  |  |

In addition, the system should generate a UID for each successfully added employee.

**UI**

1. Employee entry form should have its own screen.
2. User should receive feedback that an employee was successfully added.
3. User should receive feedback if an employee was not added for the following reasons:
   1. Employee already exists – User should be alerted
   2. Missing required fields – User should be alerted as to which fields are missing
   3. Invalid field values – User should be alerted as to why a field value is wrong
4. If employee was not added for some other reason, a generic error message should be displayed

**Validation**

1. An employee cannot be entered into the system twice. Employee identity is based on first name, middle name, last name, and birth date. Employees with all four identical fields are considered to be the same employee.
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 employee to the system as long as it passes validation.
2. All functional requirements are met.



### **User Story 1.3: Search Employees**

#### Description

As an HR Manager I want to search for employees in the system so that I can select employee to manage.

#### 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 employee 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 Employee, unless employee entities are already in the database.

#### Acceptance Criteria

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



### **User Story 1.4: View/Edit Employee Profile**

#### Description

As an HR Manager, I want to view and edit employee profiles so that I can change employee details when needed.

#### Functional Requirements

All employee fields except UID should be editable. Validation rules for employee fields from Story 1.2 must still be followed

UI

1. Employee to be viewed/edited must be selected from search results (see Story 1.3)
2. View/Edit Employee form must be in its own screen.
3. View/Edit form fields must be pre-populated with employee information.
4. All employee fields should be visible, including UID.
5. UID should only be displayed. It should not be editable.
6. User should receive feedback if employee was successfully edited.
7. User should receive feedback if an employee was not updated for the following reasons:
   1. Identical employee found – User should be alerted if editing an employee would make it identical to an existing employee
   2. Missing required fields – User should be alerted as to which fields are missing
   3. Invalid field values – User should be alerted as to why a field value is wrong
8. If employee was not edited for some other reason, a generic error message should be displayed.
9. If employee 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 an employee to view/edit from search results.
2. User can view/edit an employee.
3. All functional requirements are fulfilled.



### **User Story 1.5: Add Employee Compensation Details**

#### Description

As an HR Manager, I want to add monthly compensation details for an employee so that I can track an employee’s work and salary in the system.

It is possible for an employee to receive multiple types of compensation in a month. For example, salespeople receive a monthly salary and can receive commission on sales and bonuses. These should be recorded in the system. Compensation details include Compensation type (salary, bonus, commission, and so on), amount, and description.

#### Functional Requirements

Assumptions

1. All amounts are in a single home currency where the application will be used.
2. It will be up to the HR Manager to define monthly compensation (before or after taxes, SSS, and so on)

Fields

|  |  |  |
| --- | --- | --- |
| Field | Type | Required? |
| Type | String | *See below* |
| Amount | Numeric | *See below* |
| Description | String | *See below* |
| Date | Date | Only year and month is important. Assume end-of-month pay period. |

Types of Compensation

1. Salary – This is basic monthly salary. Only one salary entry per employee per month can be added. It is possible for the value to be zero or negative due to deductions. Description is optional.
2. Bonus – Any bonus received beyond contractual salary. An employee can have multiple bonuses in a month. Amount must be greater than zero. Description is required.
3. Commission – Any compensation based on a percentage from closing sales. An employee can have multiple commission from multiple sales. Amount should be actual amount received, not percentage. Amount should be greater than zero. Description is required. Entry can be based on individual closed sales, or aggregated sales.
4. Allowance – This covers per-diem, over-time food or transportation allowance, etc. An employee can have more than one type of allowance. Amount should be greater than zero. Amount is assumed to have been converted to home currency. Description is required.
5. Adjustment – This covers anything that is not covered above. More than one entry is allowed. Amount can be any value except zero. Description is required.

Validation

1. See section above for validation rules for the different compensation types.

#### Out-of-scope

1. Compensation calculation based on employee position
2. Currency conversion for per-diem
3. Calculations of deductions (taxes, SSS, health insurance, and so on)
4. Editing of compensation details is another story and is out of scope for this story

#### Dependencies

Story 1.3

#### Acceptance Criteria

1. User can add compensation details as per functional requirements



### **User Story 1.6: View Compensation History**

#### Description

As an HR Manager, I want to view the monthly compensation history of an employee so that I can generate a report on how much an employee earns every month.

Only total compensation per month will be displayed for chosen employee.

#### Functional Requirements

UI Flow

1. User searches for employee
2. User selects employee
3. User enters start month/year and end month/year
4. New screen shows total compensation per month for each year
5. User can choose to return to Home Page or Step 4

UI

1. Total per month should be shown. No breakdown as this is done in another story.
2. If specified date range covers more than one year, show monthly compensation for each month in each year within specified date range. For example, do not add totals for January from all years together.
3. If no compensation details for a month can be found, that month is not displayed. However, in the case where compensation details total zero, that month should still be displayed.

Validation

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

#### Out-of-scope

Breakdown of monthly compensation is done in Story 1.7.

#### Dependencies

Story 1.3

#### Acceptance Criteria

1. Compensation report for an employee is displayed



### **User Story 1.7: View Compensation Breakdown for a Month**

#### Description

As an HR Manager I want to view monthly breakdown of employment details for an employee so that I can see how much an employee is making from different types of income.

#### Functional Requirements

UI Flow

1. User brings up compensation history as per Story 1.6.
2. User selects displayed month from report.
3. New screen displays compensation details for that month.
4. User can go to home page or select new month to display.

UI

1. Only one month can be selected at a time for breakdown report
2. Total for the month should still be displayed in breakdown screen
3. All fields for compensation details should be displayed. Description can be truncated if too long.

#### Out-of-scope

None

#### Dependencies

Story 1.6

#### Acceptance Criteria

1. User can bring up compensation breakdown of an employee for a specific month.



### **User Story 1.8: Edit Employee Compensation Details**

#### Description

As an HR Manager, I want to edit monthly employment details for an employee so that I can make changes to an employee’s work and salary record.

#### Functional Requirements

UI Flow

1. User brings up compensation breakdown as per Story 1.7
2. User selects compensation to edit
3. User edits compensation details
4. Compensation breakdown screen is displayed with updated details and totals.

UI

1. User can only edit amount and description.
2. Validation rules for adding compensation in Story 1.5 are still enforced.

#### Out-of-scope

1. Deletion of compensation
2. Editing of compensation type and date.

#### Dependencies

Story 1.7

#### Acceptance Criteria

1. User is able to edit compensation details.

