

# Employers and Holiday Management System

Prepared by: Imrane Errafi

December 16, 2024

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Design Patterns Used</b>	<b>2</b>
2.1	Model-View-Controller (MVC) . . . . .	2
2.2	Data Access Object (DAO) . . . . .	2
<b>3</b>	<b>Folder Structure</b>	<b>2</b>
3.1	Model Folder . . . . .	2
3.1.1	EmployerModel . . . . .	2
3.1.2	HolidayModel . . . . .	3
3.2	DAO Folder . . . . .	3
3.3	View Folder . . . . .	3
3.4	Enums Folder . . . . .	3
3.5	Controller Folder . . . . .	3
3.6	Main Class . . . . .	3
<b>4</b>	<b>Conclusion</b>	<b>4</b>
<b>5</b>	<b>Figures</b>	<b>4</b>
5.1	Login Form . . . . .	4
5.2	Employer Frame . . . . .	4
5.3	Holiday Frame . . . . .	4
5.4	Create Login Frame . . . . .	4

# 1 Introduction

The "Employers and Holiday Management System" is a Java-based project designed to manage employer details and holiday allocations. The project adheres to the **Model-View-Controller (MVC)** and **Data Access Object (DAO)** design patterns. This report provides a comprehensive overview of the project structure, including its components, logic, and functionality.

## 2 Design Patterns Used

### 2.1 Model-View-Controller (MVC)

The MVC pattern divides the application into three interconnected components:

- **Model:** Represents the data and business logic.
- **View:** Handles the user interface.
- **Controller:** Acts as a mediator between the model and the view.

### 2.2 Data Access Object (DAO)

The DAO pattern is used to abstract and encapsulate all access to the data source. This project uses DAO to handle CRUD operations for employers and holidays.

## 3 Folder Structure

The project is organized into the following folders within the src directory:

### 3.1 Model Folder

This folder contains the business logic and data structures. It is further divided into:

#### 3.1.1 *EmployerModel*

- **Employer.java:** Defines the Employer class with attributes: id, firstName, lastName, email, phoneNumber, salary, Role, Poste. It includes getters for all attributes.
- **EmployerLogic.java:** Implements validation logic to ensure:
  - Email contains @.
  - First and last names are not null.
  - Phone number contains 10 digits.
  - Salary is not null.

### 3.1.2 *HolidayModel*

- **Holiday.java:** Defines the Holiday class with attributes: id, employerId, employerName, startDate, endDate, holidayType. Includes getters for all attributes.
- **HolidayLogic.java:** Implements logic to validate:
  - End date is after the start date.
  - Each employer has a maximum of 25 holiday days.

## 3.2 DAO Folder

The DAO folder contains classes and interfaces to manage database operations. It includes:

- **InterfaceDAO.java:** A generic interface defining CRUD operations.
- **DBConnection.java:** Provides a method to establish and return database connections.
- **EmployerDAO:** Contains EmployerDAO.java, which implements InterfaceDAO for CRUD operations on employer data.
- **HolidayDAO:** Contains HolidayDAO.java, which implements InterfaceDAO for CRUD operations on holiday data.

## 3.3 View Folder

The view folder contains components for the graphical user interface, including panels, buttons, input forms, and other frontend features.

## 3.4 Enums Folder

This folder contains all enums used in the project, such as Role, Poste, holidayType, which define fixed sets of constants.

## 3.5 Controller Folder

This folder handles interactions between the model and the view. It includes:

- **EmployerController:** Manages CRUD operations for employers and links the view and model.
- **HolidayController:** Manages CRUD operations for holidays and links the view and model.

## 3.6 Main Class

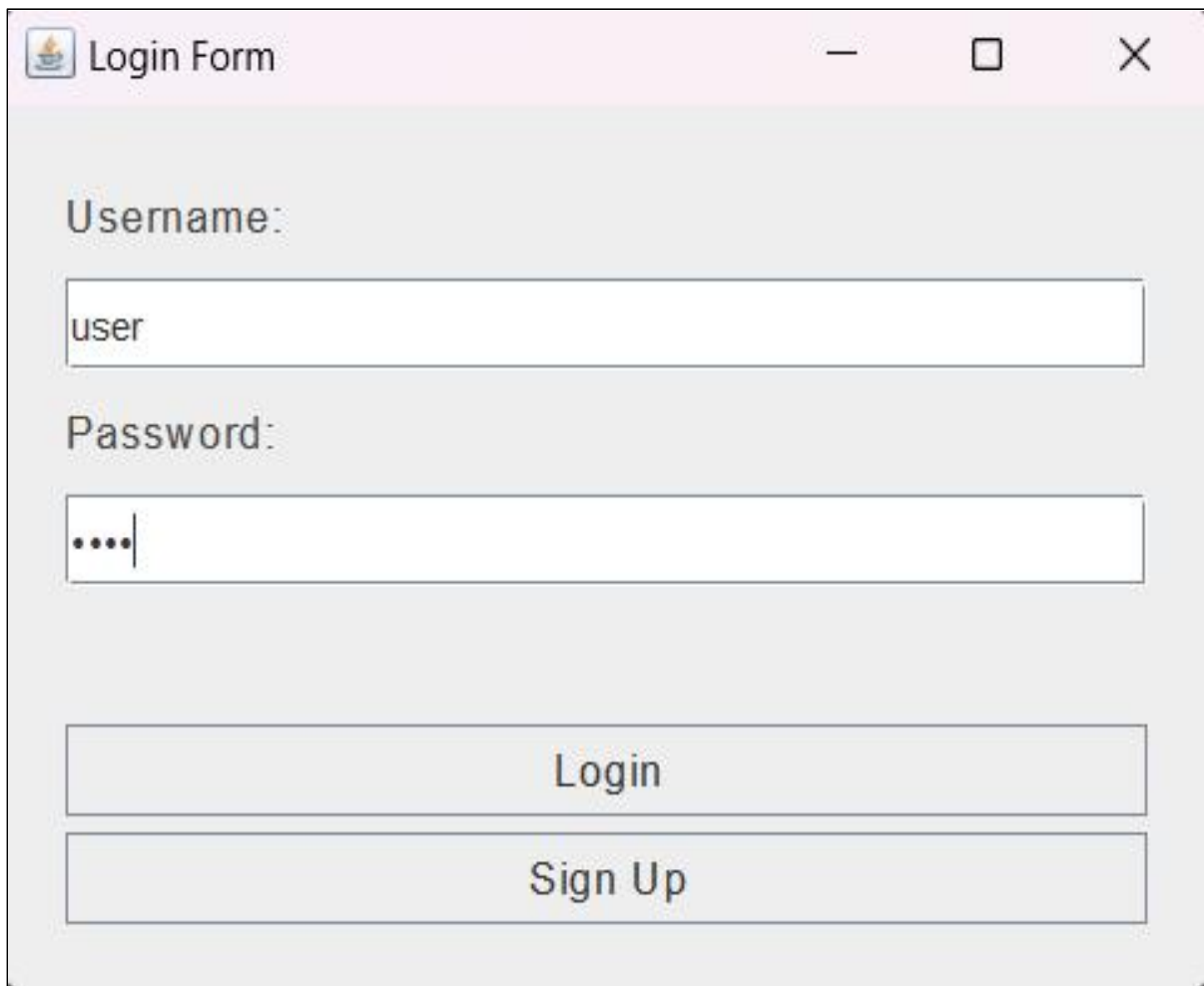
The entry point of the project is Main.java, which initializes the application and its components.

## 4 Conclusion

The "Employers and Holiday Management System" is a well-structured Java project adhering to industry-standard design patterns. By following the MVC and DAO patterns, it ensures maintainability, scalability, and a clear separation of concerns. This report outlines the project structure, providing a blueprint for understanding its implementation.

## 5 Figures

### 5.1 Login Form



The image shows a Java Swing window titled "Login Form". The window has a title bar with a standard icon, a minimize button, a maximize button, and a close button. The main content area is light gray and contains two labels: "Username:" and "Password:". Below "Username:" is a text input field containing the text "user". Below "Password:" is a password input field with four dots and a cursor. At the bottom of the form are two buttons: "Login" and "Sign Up".

Figure 1: Login Form

## 5.2 Employer Frame

The screenshot shows a web application window titled "Employer Platform". It has two tabs: "Employer" (selected) and "Holiday". The "Employer" tab contains a form with the following fields:

- First Name:
- Last Name:
- Email:
- Telephone Number:
- Salary:
- Role:
- Poste:

Below the form is a table with the following data:

Id	Nom	Prenom	Email	Salaire
1	Smith	John_Doe	johndoe@email.com	200.0
4	errafi	imrane	imrane@gmail.com	2000.0
5	user	user	user@gmail.com	20000.0

At the bottom of the interface are four buttons: "Add", "Remove", "Update", and "Create".

Figure 2: Employer Frame

### 5.3 Holiday Frame

Employer Platform

Employer Holiday

Start Date (dd/MM/yyyy): 16/12/2024

End Date (dd/MM/yyyy): 16/12/2024

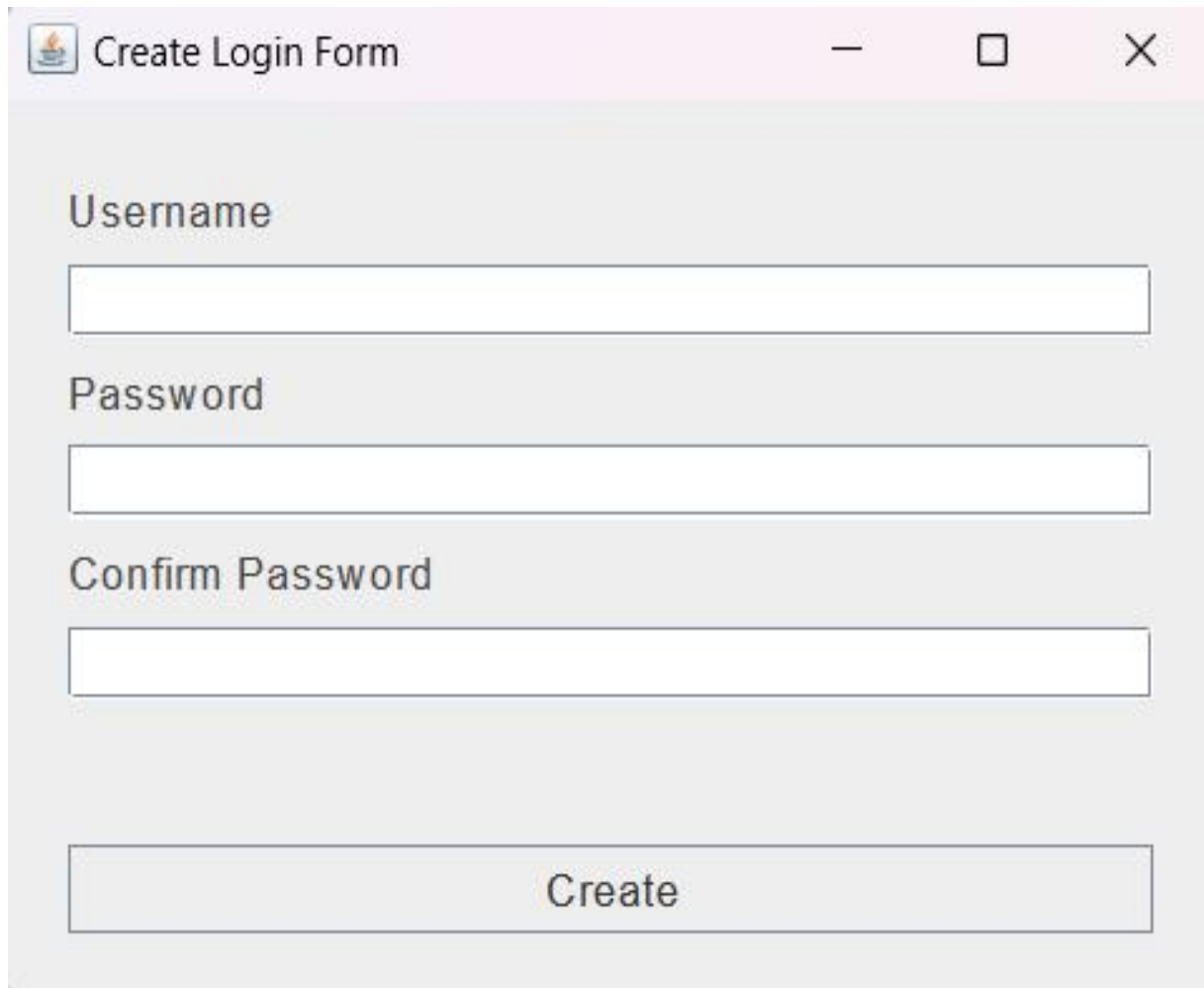
Holiday Type: PAYED\_HOLIDAY

Id	Nom	Start Date	End Date	Holiday Type
15	user user	2024-12-15	2024-12-20	PAYED_HOLIDAY

Add Remove Update

Figure 3: Holiday Frame

## 5.4 Create Login Frame



The image shows a Java Swing window titled "Create Login Form". The window has a light gray background and a title bar with standard Windows-style controls (minimize, maximize, close). Inside the window, there are three text input fields stacked vertically, each with a label to its left: "Username", "Password", and "Confirm Password". Below these fields is a single button labeled "Create".

Username

Password

Confirm Password

Create

Figure 4: Create Login Frame