Java: Practice Session Report

Ali Assouli

December 12, 2024

Table Of Contents

1. Objective

The goal of this practice session was to design and implement a Java-based Employee Management System following the **MVC (Model-View-Controller)** architecture. This included integrating a PostgreSQL database for persistent storage.

2. Assignment Overview

The project involved creating several interconnected components:

- **Model:** Represents employee data with attributes such as *name*, *email*, *phone*, *salary*, *post*, and *role*.
- **DAO (Data Access Object):** Handles database operations such as insert, delete, and update.
- **Controller:** Implements logic to manage and process user input.
- **View:** Provides a graphical interface for user interaction.

3. Code Implementation

3.1. Main Class

```
import Controllers.EmployeeController;
import DAO.EmployeeDAOImpl;
import Views.EmployeeView;

public class Main {
  public static void main(String[] args) {
    // Initialize the database connection
    EmployeeDAOImpl dao = new EmployeeDAOImpl();

    // Render the View
    EmployeeView ev = new EmployeeView();

    // Add controller for the view
    EmployeeController ec = new EmployeeController();
  }
}
```

3.2. DAO (Data Access Object)

3.2.1. DAO Interface

```
package DAO;

interface EmployeeDAOI {
  // Credentials
  public String url = "jdbc:postgresql://localhost:5432/java_db";
  public String dbuser = "postgres";   // Database user
```

```
  public String dbpw = "pg1234";        // Database password

  // Abstract methods
  public boolean addEmployee(Employee em);
  public boolean deleteEmployee(int id);
  public boolean updateEmployee(int id, Employee em);
}
```

3.2.2. DAO Implementation

```
package DAO;

public class EmployeeDAOImpl implements EmployeeDAOI {
  // Constructor
  public EmployeeDAOImpl();

  // Methods to be override here
}
```

3.3. Model

```
package Models;

public class Employee {
  // Constructor
  public Employee(ResultSet rs);

  // Getters
  public int getId();
  public String getLname();
  public String getFname();
  public String getEmail();
  public double getSalary();
  public String getPhone();
  public String getPost();
  public String getRole();

  // Setters
  public void setId(int id);
  public void setLname(String lname);
  public void setFname(String fname);
  public void setEmail(String email);
  public void setSalary(double salary);
  public void setPhone(String phone);
  public void setPost(String post);
  public void setRole(String role);

  // Methods for Controller interactions
  public boolean addEmployee();
  public static boolean deleteEmployee(int id);
  public boolean updateEmployee(int id);

  public String toString();
}
```

3.4. Controller

```
package Controllers;

public class EmployeeController {
  // Constructor
  public EmployeeController();

  // Event listeners initialization methods
  private void initAddEvent();
  private void initDeleteEvent();
  private void initUpdateEvent();
  private void initShowEvent();

  // Useful View handling methods
  public static void populateTable();
  public static void emptyFields();
}
```

3.5. View

```
package Views;

public class EmployeeView extends JFrame {
  // Constructor
  public EmployeeView();
}
```

4. Results

The application was tested on Ubuntu using PostgreSQL as the database management system.

The implementation successfully demonstrated:

- Connecting to a PostgreSQL database.
- Performing CRUD (Create, Read, Update, Delete) operations on employee records.
- Displaying employee data in a dynamic GUI table.

5. Challenges and Solutions

- **Challenge:** Handling SQL exceptions during database operations.
- **Solution:** Used `try-catch` blocks with detailed error logging.
- **Challenge:** Keeping the GUI synchronized with database changes.
- **Solution:** Implemented methods to dynamically refresh the table view.

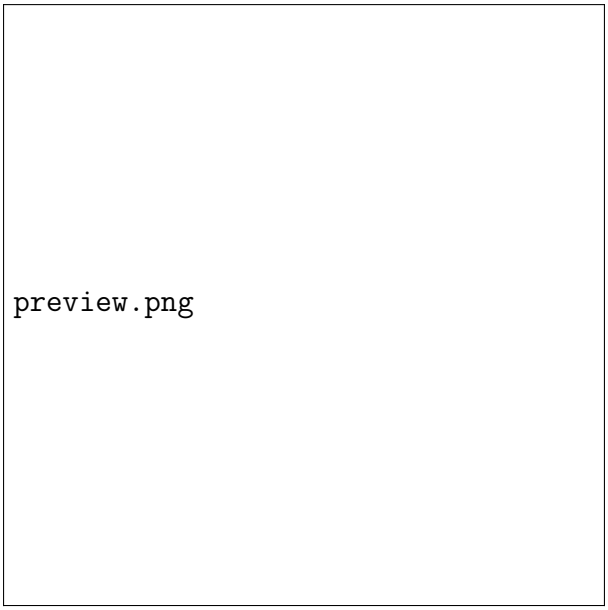

Figure 1: Application Preview

6. Conclusion

This practice session provided valuable hands-on experience in developing a complete Java application with database integration and GUI implementation. It reinforced key concepts of **MVC architecture**, object-oriented programming, and SQL database interaction.