Name : M Shahzeb

Roll Number: SP22-BSE-073

Submitted to: Dr Muktiar Zamin

JOB PORTAL SYSTEM REPORT

1. Project Overview

The Job Portal System is a web-based application designed to connect job seekers with potential employers. The objective of this project is to create a centralized platform where users can efficiently search for job opportunities, apply for jobs, manage their applications, and allow employers to post job listings and review applications.

2. Modules:

Search Job: This module enables job seekers to find job listings based on specific criteria, such as keywords, location, and job category.

Apply for Job: In this module, job seekers can submit their applications along with resumes and cover letters for job listings they are interested in.

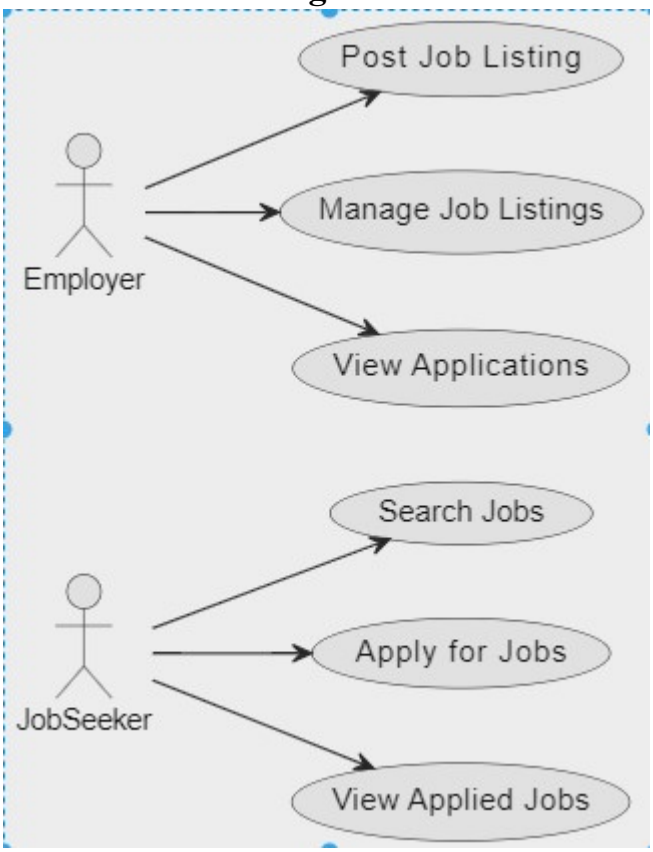
View Applied Jobs: This feature allows job seekers to track their submitted applications and view their current status (e.g., submitted, shortlisted).

Post Job Listing: This module provides employers with the functionality to create and publish job openings on the portal for job seekers to view.

Manage Job Listings: Employers can edit or remove their job postings through this module, allowing them to keep their listings up-to-date.

View Applications: This module enables employers to review applications submitted for their job postings and manage the application process.

3. Use Case Diagram:



Actors:

Job Seeker: Uses the portal to search for jobs, apply for positions, and view the status of their applications.

Employer: Responsible for posting job listings, managing listings, and reviewing applications.

Main Use Cases:

Search Jobs: Job seekers can search for available job listings using various criteria.

Apply for Jobs: Job seekers submit their applications for job positions.

View Applied Jobs: Job seekers can track the status of their submitted applications.

Post Job Listing: Employers can create and publish job openings on the portal.

Manage Job Listings: Employers can edit or remove job postings as needed.

View Applications: Employers can review applications received for their job listings.

Fully Dressed UseCase:

Use Case: Post Job Listings

Use Case ID: UC-001
Use Case Name: Post Job Listings
Primary Actor: Employer (or Admin)
Stakeholders and Interests:

- **Employer**: Wants to post job listings to attract qualified candidates.
- **Job Seekers**: Interested in viewing job listings for potential employment opportunities.
- **System Administrator**: Ensures the job listing conforms to platform guidelines and policies.

Preconditions

1. Employer must be registered and logged into the system.
2. Employer has a verified company profile on the platform.

Postconditions

1. Job listing is successfully created and visible to job seekers.
2. Employer can manage (edit, deactivate, or delete) the job listing if needed.

Main Success Scenario (Basic Flow)

1. **Employer navigates to the "Post a Job" page**:
   The system displays a job posting form.
2. **Employer enters job details**:
   - Job Title
   - Job Description
   - Required Skills and Qualifications
   - Job Type (e.g., Full-time, Part-time, Contract)
   - Salary Range
   - Location
   - Application Deadline
3. **Employer reviews job listing details**:
   The system displays a preview of the job listing.
4. **Employer submits the job listing**:
   - The system validates the information (e.g., checks for mandatory fields, correct format).
5. **System confirms job listing creation**:
   - The job listing is saved in the database and marked as "Active."
   - The system sends a confirmation notification to the employer.
6. **System makes job listing available**:
   - The job listing is now visible to job seekers browsing available positions.

Extensions (Alternate Flows)

- **3a. Employer provides incomplete or invalid details**:
  - The system prompts the employer to correct the missing or invalid fields.
  - Employer corrects and resubmits the details.
- **5a. Job listing fails system validation**:
  - The system provides error messages detailing what needs to be corrected.
  - Employer makes the required changes and resubmits.

Special Requirements

- The job listing should have the option to auto-deactivate after the application deadline.

- Employers can select whether they want applications submitted directly on the platform or via an external link.

Assumptions

- Employers have the necessary permissions to post job listings.
- Job listings are only accessible to authenticated job seekers if the platform is private.

Open Issues

- Should employers be able to preview how the listing appears to job seekers before finalizing?
- Are there specific guidelines or moderation for content within job postings?

GRASP Principles and OOP principles Applied

1. Information Expert

- **Application:** The principle of Information Expert assigns responsibility to the class that has the necessary information to fulfill it. Here, the `JobListing` class holds all the job-specific data (like `title`, `description`, `skills`, etc.) and therefore is responsible for providing that information whenever required. Similarly, `SystemService` contains methods to display forms, validate listings, and make them visible, centralizing the responsibilities related to system services.

2. Creator

- **Application:** The `JobListingFactory` class follows the Creator principle by taking responsibility for creating `JobListing` objects. Whenever a new job listing is posted, `JobListingFactory` is responsible for creating an instance of `JobListing` with the provided details.

3. Controller

- **Application:** The `Employer` class acts as a controller for posting job listings. It handles user interactions, like navigating to the job posting form, and delegates tasks to other classes such as `JobListingService` for processing the listing.

4. High Cohesion

- **Application:** The project's classes exhibit high cohesion, meaning each class has a focused responsibility. For instance:
    - `Employer` focuses on actions an employer performs.
    - `JobListingService` manages the business logic related to posting job listings.
    - `SystemService` centralizes utility methods related to displaying, validating, and saving listings.
- Each class has a clear role, which makes the system easier to understand and maintain.

5. Low Coupling

- **Application:** The system minimizes dependencies between classes to achieve low coupling. For instance, `SystemService` provides a centralized interface for job listing actions, reducing direct dependencies among `Employer`, `JobListingService`, and `JobListingDAO`. Additionally, `SystemService` is a singleton, which prevents multiple instances and ensures controlled access across the system.

6. Singleton Pattern (an application of Pure Fabrication)

- **Application:** The `SystemService` class implements the Singleton pattern, ensuring only one instance of `SystemService` exists throughout the application. This design is a type of Pure Fabrication, creating an instance where necessary to maintain system functionality, control, and data integrity.

7. Polymorphism

- **Application:** The project uses polymorphism in the data access layer through the `JobListingDAO` interface. `JobListingDAOImpl` implements this interface, allowing for flexible, interchangeable implementations of data access logic without changing the core system.

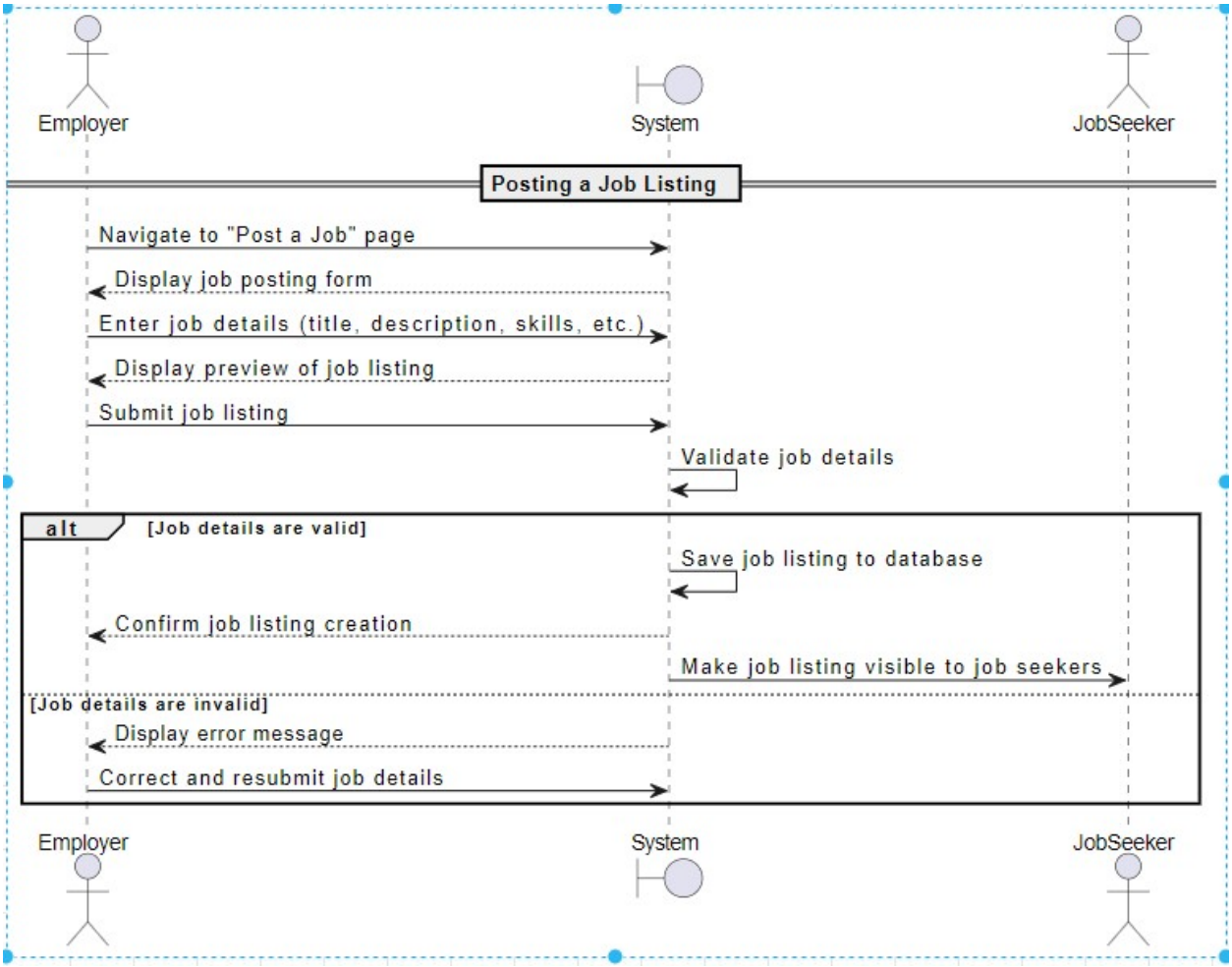
8. Indirection

- **Application:** Indirection is used to decouple the responsibilities among different classes. For instance, `JobListingService` intermediates between `Employer` and `SystemService`, and `SystemService` handles various operations without `Employer` directly accessing database functionality.

9. Factory Pattern (supporting Creator and Pure Fabrication)

- **Application:** The `JobListingFactory` follows the Factory Pattern to create instances of `JobListing`. This separates the creation logic from other classes, supporting reusability and centralized object creation.

SSD



SSD diagram code:

```
// --- Presentation Layer ---
class Employer {
    private String name;
    private JobListingService jobListingService;

    public Employer(String name, JobListingService jobListingService) {
        this.name = name;
        this.jobListingService = jobListingService;
    }
```

```
    public void postJobListing(String title, String description, String skills, String jobType, String location, double salary) {
        System.out.println(name + " is navigating to 'Post a Job' page.");

        JobListing jobListing = JobListingFactory.createJobListing(title, description, skills, jobType, location, salary);
        jobListingService.postJob(jobListing, this);
    }
}

// --- Business Logic Layer ---
class JobListing {
    private String title;
    private String description;
    private String skills;
    private String jobType;
    private String location;
    private double salary;

    public JobListing(String title, String description, String skills, String jobType, String location, double salary) {
        this.title = title;
        this.description = description;
        this.skills = skills;
        this.jobType = jobType;
        this.location = location;
        this.salary = salary;
```

```
    }

    public String getTitle() { return title; }
    public String getDescription() { return description; }
    public String getSkills() { return skills; }
    public String getJobType() { return jobType; }
    public String getLocation() { return location; }
    public double getSalary() { return salary; }
}

class JobListingService {
    private SystemService systemService;

    public JobListingService(SystemService systemService) {
        this.systemService = systemService;
    }

    public void postJob(JobListing jobListing, Employer employer) {
        systemService.displayJobPostingForm();
        systemService.previewJobListing(jobListing);

        if (systemService.validateJobListing(jobListing)) {
            systemService.saveJobListing(jobListing);
            systemService.confirmJobListingCreation(employer);
            systemService.makeJobListingVisibleToJobSeekers(jobListing);
        } else {
            systemService.displayErrorMessage();
```

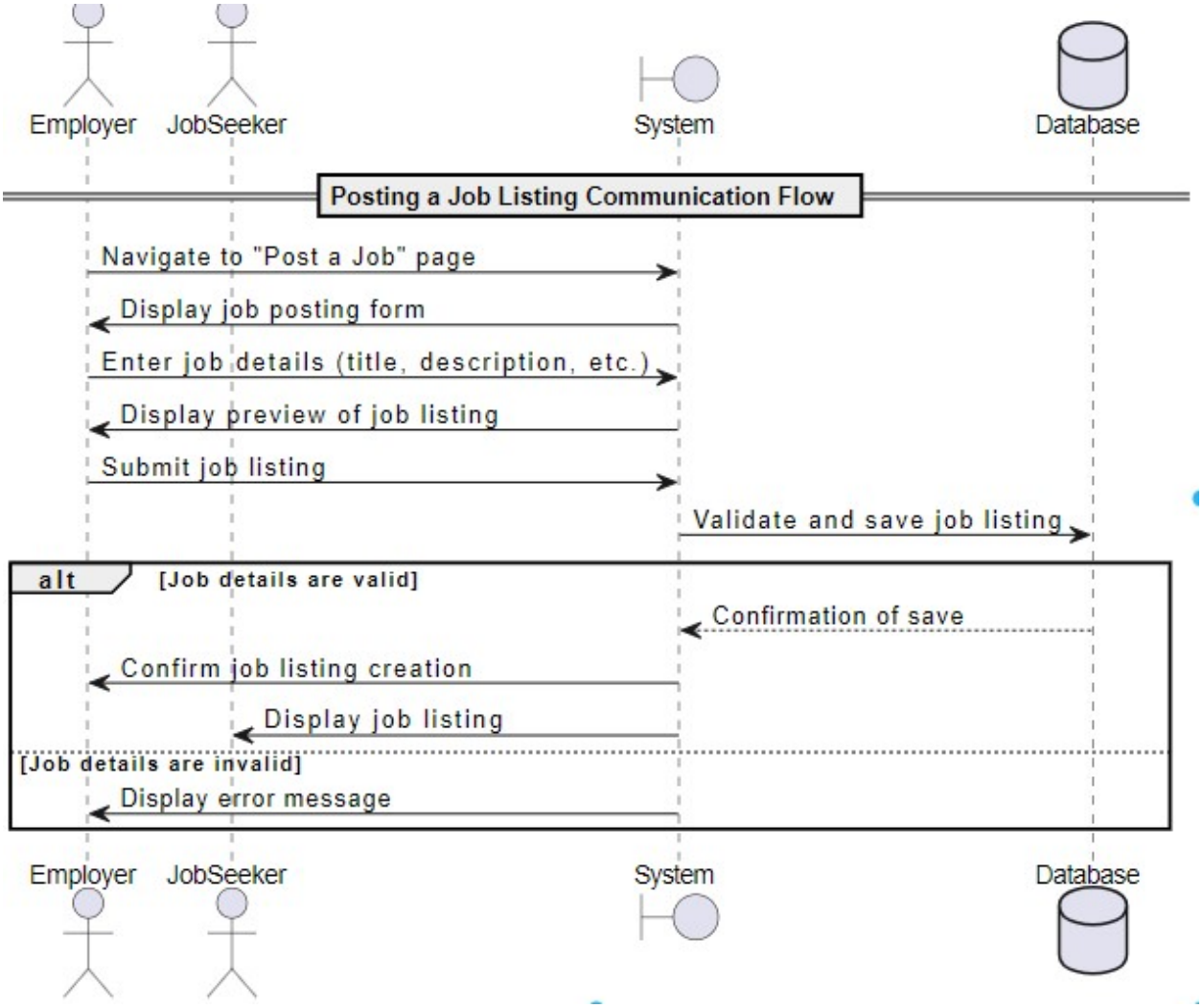
```
        }
    }
}

// --- Data Access Layer ---
interface JobListingDAO {
    void save(JobListing jobListing);
}

class JobListingDAOImpl implements JobListingDAO {
    @Override
    public void save(JobListing jobListing) {
        System.out.println("Saving job listing to database: " + jobListing.getTitle());
    }
}

// --- Utility Factory ---
class JobListingFactory {
    public static JobListing createJobListing(String title, String description, String skills, String
jobType, String location, double salary) {
        return new JobListing(title, description, skills, jobType, location, salary);
    }
}

// --- Singleton System Service ---
class SystemService {
    private static SystemService instance;
```

```
private JobListingDAO jobListingDAO;

private SystemService() {
    this.jobListingDAO = new JobListingDAOImpl();
}

public static synchronized SystemService getInstance() {
    if (instance == null) {
        instance = new SystemService();
    }
    return instance;
}

public void displayJobPostingForm() {
    System.out.println("Displaying job posting form.");
}

public void previewJobListing(JobListing jobListing) {
    System.out.println("Displaying job preview for '" + jobListing.getTitle() + "'.");
}

public boolean validateJobListing(JobListing jobListing) {
    return jobListing.getTitle() != null && !jobListing.getTitle().isEmpty()
            && jobListing.getDescription() != null && !jobListing.getDescription().isEmpty();
}

public void saveJobListing(JobListing jobListing) {
```

```
        jobListingDAO.save(jobListing);
    }

    public void confirmJobListingCreation(Employer employer) {
        System.out.println("Job listing creation confirmed for " + employer + ".");
    }

    public void makeJobListingVisibleToJobSeekers(JobListing jobListing) {
        System.out.println("Making job listing visible to job seekers.");
    }

    public void displayErrorMessage() {
        System.out.println("Job details are invalid. Displaying error message.");
    }
}

// --- Main Program ---
public class Main {
    public static void main(String[] args) {
        SystemService systemService = SystemService.getInstance();
        JobListingService jobListingService = new JobListingService(systemService);

        Employer employer = new Employer("John Doe", jobListingService);
        employer.postJobListing("Software Engineer", "Responsible for developing applications.",
"Java, Spring Boot", "Full-Time", "Remote", 85000.00);
    }
}
```

Communication Diagram

s



COde for Communicaton Diagram

```
// Employer.java (Presentation Layer)
public class Employer {
    private String name;
    private JobListingService jobListingService;

    public Employer(String name, JobListingService jobListingService) {
```

```
        this.name = name;
        this.jobListingService = jobListingService;
    }

    public void postJobListing(String title, String description, String skills, String jobType, String location, double salary) {
        System.out.println(name + " is navigating to 'Post a Job' page.");
        JobListing jobListing = JobListingFactory.createJobListing(title, description, skills, jobType, location, salary);
        jobListingService.postJob(jobListing, this);
    }

    @Override
    public String toString() {
        return name;
    }
}

// JobListing.java (Business Logic Layer)
public class JobListing {
    private String title;
    private String description;
    private String skills;
    private String jobType;
    private String location;
```

```
    private double salary;

    public JobListing(String title, String description, String skills, String jobType,
String location, double salary) {
        this.title = title;
        this.description = description;
        this.skills = skills;
        this.jobType = jobType;
        this.location = location;
        this.salary = salary;
    }

    public String getTitle() { return title; }
    public String getDescription() { return description; }
    public String getSkills() { return skills; }
    public String getJobType() { return jobType; }
    public String getLocation() { return location; }
    public double getSalary() { return salary; }
}

// JobListingService.java (Business Logic Layer)
public class JobListingService {
    private SystemService systemService;

    public JobListingService(SystemService systemService) {
```

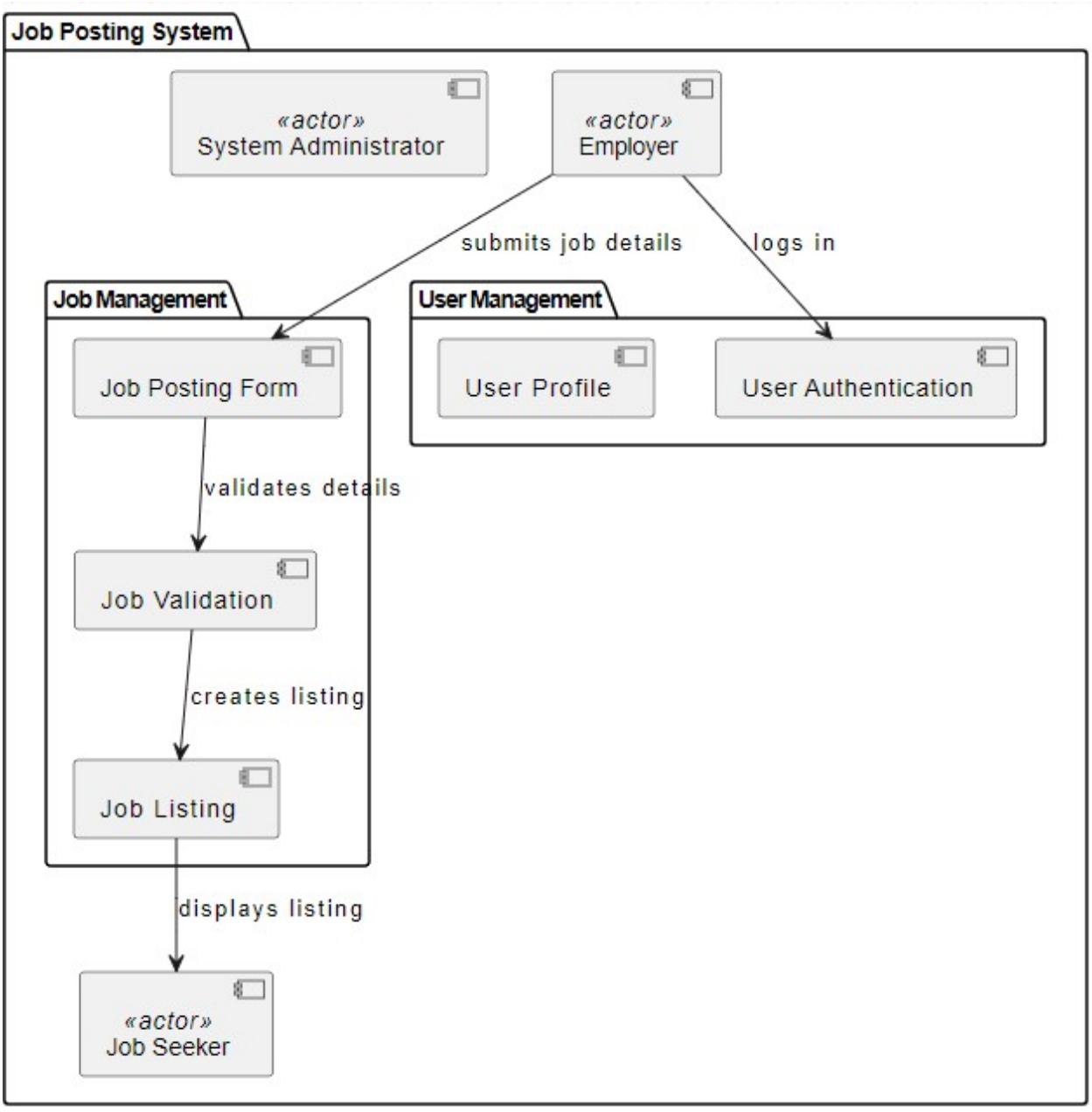
```
        this.systemService = systemService;
    }

    public void postJob(JobListing jobListing, Employer employer) {
        systemService.displayJobPostingForm();
        systemService.previewJobListing(jobListing);

        if (systemService.validateJobListing(jobListing)) {
            systemService.saveJobListing(jobListing);
            systemService.confirmJobListingCreation(employer);
            systemService.makeJobListingVisibleToJobSeekers(jobListing);
        } else {
            systemService.displayErrorMessage();
        }
    }
}

// JobListingDAO.java (Data Access Layer Interface)
public interface JobListingDAO {
    void save(JobListing jobListing);
}

// JobListingDAOImpl.java (Data Access Layer Implementation)
public class JobListingDAOImpl implements JobListingDAO {
```

```
    @Override
    public void save(JobListing jobListing) {
        System.out.println("Saving job listing to database: " + jobListing.getTitle());
    }
}

// JobListingFactory.java (Utility Factory)
public class JobListingFactory {
    public static JobListing createJobListing(String title, String description, String
skills, String jobType, String location, double salary) {
        return new JobListing(title, description, skills, jobType, location, salary);
    }
}

// SystemService.java (Singleton System Service)
public class SystemService {
    private static SystemService instance;
    private JobListingDAO jobListingDAO;

    private SystemService() {
        this.jobListingDAO = new JobListingDAOImpl();
    }

    public static synchronized SystemService getInstance() {
        if (instance == null) {
```

```
            instance = new SystemService();
        }
        return instance;
    }

    public void displayJobPostingForm() {
        System.out.println("Displaying job posting form.");
    }

    public void previewJobListing(JobListing jobListing) {
        System.out.println("Displaying job preview for '" + jobListing.getTitle() + "'.");
    }

    public boolean validateJobListing(JobListing jobListing) {
        return jobListing.getTitle() != null && !jobListing.getTitle().isEmpty()
                && jobListing.getDescription() != null
    && !jobListing.getDescription().isEmpty();
    }

    public void saveJobListing(JobListing jobListing) {
        jobListingDAO.save(jobListing);
    }

    public void confirmJobListingCreation(Employer employer) {
        System.out.println("Job listing creation confirmed for " + employer + ".");
```

```
    }

    public void makeJobListingVisibleToJobSeekers(JobListing jobListing) {
        System.out.println("Making job listing visible to job seekers.");
    }

    public void displayErrorMessage() {
        System.out.println("Job details are invalid. Displaying error message.");
    }
}

// Main.java (Main Program)
public class Main {
    public static void main(String[] args) {
        SystemService systemService = SystemService.getInstance();
        JobListingService jobListingService = new JobListingService(systemService);

        Employer employer = new Employer("John Doe", jobListingService);
        employer.postJobListing("Software Engineer", "Responsible for developing
applications.", "Java, Spring Boot", "Full-Time", "Remote", 85000.00);
    }
}
```

Package diagram



Code For Package Diagram

// Package: UserManagement

package UserManagement;

public class UserProfile {

```
private String username;
private String email;

public UserProfile(String username, String email) {
   this.username = username;
   this.email = email;
}

public String getUsername() {
   return username;
}

public String getEmail() {
   return email;
}

public void setUsername(String username) {
   this.username = username;
}

public void setEmail(String email) {
   this.email = email;
}
```

```
    public void displayProfile() {
        System.out.println("Username: " + username);
        System.out.println("Email: " + email);
    }
}

public class UserAuthentication {
    public boolean login(String username, String password) {
        // Simple placeholder logic for authentication
        if (username.equals("admin") && password.equals("password")) {
            System.out.println("Login successful!");
            return true;
        }
        System.out.println("Invalid username or password.");
        return false;
    }

    public void logout() {
        System.out.println("User logged out.");
    }
}

// Package: JobManagement
package JobManagement;
```

```
public class JobPostingForm {
    private String jobTitle;
    private String jobDescription;

    public JobPostingForm(String jobTitle, String jobDescription) {
        this.jobTitle = jobTitle;
        this.jobDescription = jobDescription;
    }

    public String getJobTitle() {
        return jobTitle;
    }

    public String getJobDescription() {
        return jobDescription;
    }
}

public class JobValidation {
    public boolean validateJobDetails(JobPostingForm job) {
        if (job.getJobTitle() == null || job.getJobTitle().isEmpty()) {
            System.out.println("Job title is required.");
            return false;
```

```
        }
        if (job.getJobDescription() == null || job.getJobDescription().isEmpty()) {
            System.out.println("Job description is required.");
            return false;
        }
        System.out.println("Job details validated.");
        return true;
    }
}

public class JobListing {
    private List<JobPostingForm> jobList = new ArrayList<>();

    public void addJob(JobPostingForm job) {
        jobList.add(job);
        System.out.println("Job added to listing: " + job.getJobTitle());
    }

    public void displayJobs() {
        System.out.println("Job Listings:");
        for (JobPostingForm job : jobList) {
            System.out.println("- " + job.getJobTitle() + ": " + job.getJobDescription());
        }
    }
```

```
}

// Package: Actors
package Actors;

import UserManagement.UserProfile;
import UserManagement.UserAuthentication;
import JobManagement.JobPostingForm;
import JobManagement.JobValidation;
import JobManagement.JobListing;

public class SystemAdministrator {
   public void manageJobPosting(JobPostingForm job, JobValidation validator,
JobListing listing) {
      if (validator.validateJobDetails(job)) {
         listing.addJob(job);
      }
   }
}

public class Employer {
   private UserProfile profile;

   public Employer(UserProfile profile) {
      this.profile = profile;
```

```
    }

    public void submitJobDetails(JobPostingForm job, SystemAdministrator admin,
JobValidation validator, JobListing listing) {
        admin.manageJobPosting(job, validator, listing);
    }

    public void login(UserAuthentication auth, String username, String password) {
        auth.login(username, password);
    }
}

public class JobSeeker {
    public void viewJobListing(JobListing listing) {
        listing.displayJobs();
    }
}
```