

Job board

Name: Racovita Alexandru

Group: 30236

Table of Contents

[Deliverable 1 2](#_Toc2055786399)

[Project Specification 3](#_Toc825512728)

[Functional Requirements 3](#_Toc362225657)

[Use Case Model 1 3](#_Toc324720183)

[Use Cases Identification 3](#_Toc159950011)

[UML Use Case Diagrams 3](#_Toc1353450407)

[Supplementary Specification 3](#_Toc501087280)

[Non-functional Requirements 3](#_Toc1600632703)

[Design Constraints 3](#_Toc729746755)

[Glossary 3](#_Toc178374120)

[Deliverable 2 3](#_Toc1123232848)

[Domain Model 3](#_Toc605147785)

[Architectural Design 3](#_Toc1115440048)

[Conceptual Architecture 3](#_Toc2005513277)

[Package Design 3](#_Toc596515525)

[Component and Deployment Diagram 4](#_Toc1771682286)

[Deliverable 3 4](#_Toc174956409)

[Design Model 4](#_Toc2136237097)

[Dynamic Behavior 4](#_Toc853475638)

[Class Diagram 4](#_Toc2045288274)

[Data Model 4](#_Toc1322833382)

[System Testing 4](#_Toc975007147)

[Future Improvements 4](#_Toc902595900)

[Conclusion 4](#_Toc1374315000)

[Bibliography 4](#_Toc2082771450)

# Deliverable 1

## Project Specification

This project aims to develop a Job Board web application that allows users to search for jobs, apply for jobs, and manage their job applications. Companies can post job listings, manage their listings, and review applicants. The application also includes features for filtering jobs by categories and searching jobs by title.

## Functional Requirements

1. User registration and authentication

2. User profile management

3. Job search functionality

4. Job listings management by companies

5. Job application management by users

6. Job categories management

7. Application tracking for companies

## Use Case Model 1

### Use Cases Identification

Use Case 1: Register a new user  
  
Level: User Goal  
Primary Actor: Unregistered User  
Main success scenario:  
  
1. Unregistered User visits the registration page.  
2. Unregistered User fills out the registration form with the required information.  
3. Unregistered User submits the form.  
4. System validates the information and creates a new user account.  
5. System sends a confirmation email to the user.

Use Case 2: Authenticate a user

Level: User Goal  
Primary Actor: Registered User  
Main success scenario:  
  
1. Registered User visits the login page.  
2. Registered User enters their username and password.  
3. System validates the credentials.  
4. System logs the user in and redirects them to their dashboard.

Use Case 3: Post a job listing  
  
Level: User Goal  
Primary Actor: Company  
Main success scenario:  
  
1. Company visits the job posting page.  
2. Company fills out the job posting form with the required information.  
3. Company submits the form.  
4. System validates the information and creates a new job listing.  
5. System displays the new job listing on the platform.

Use Case 4: Update a job listing  
  
Level: User Goal  
Primary Actor: Company  
Main success scenario:  
  
1. Company visits the job listing management page.  
2. Company selects the job listing to be updated.  
3. Company modifies the job listing information as required.  
4. Company submits the updated information.  
5. System validates the information and updates the job listing.

Use Case 5: Delete a job listing  
  
Level: User Goal  
Primary Actor: Company  
Main success scenario:  
  
1. Company visits the job listing management page.  
2. Company selects the job listing to be deleted.  
3. Company confirms the deletion.  
4. System deletes the job listing from the platform.

Use Case 6: Browse job listings  
  
Level: User Goal  
Primary Actor: Job Seeker  
Main success scenario:  
  
1. Job Seeker visits the job listings page.  
2. Job Seeker browses through the available job listings.  
3. Job Seeker filters the job listings by category or searches by job title.  
4. System displays the filtered job listings.

Use Case 7: Manage job applications  
  
Level: User Goal  
Primary Actor: Job Seeker  
Main success scenario:  
  
1. Job Seeker visits their dashboard.  
2. Job Seeker views the list of their job applications.  
3. Job Seeker updates or deletes an existing application as needed.

### UML Use Case Diagrams

Diagram

Description automatically generated

## Supplementary Specification

### Non-functional Requirements

1. Usability: The application should have an intuitive user interface, making it easy for users to navigate and find the information they need.
2. Scalability: The system should be able to handle a growing number of users and job listings without significant performance degradation.
3. Security: User data must be protected, and sensitive information like passwords should be securely stored using encryption and hashing techniques.
4. Responsiveness: The application should provide quick response times and load times, ensuring a smooth user experience.

### Design Constraints

1. The project must be developed using Java and Spring Boot as the back-end framework.
2. Front-end development should use a modern JavaScript framework (e.g., React, Angular, or Vue.js).
3. The database should be designed using a relational database management system (e.g., PostgreSQL, MySQL, or SQL Server).
4. The application should follow a RESTful API design.
5. Code should adhere to best practices, including proper documentation and testing.

## Glossary

# Job Seeker: A user who is looking for a job and applies for job listings on the platform.

# Company: An organization that posts job listings on the platform and reviews applicants.

# Job Listing: A job advertisement posted by a company, including information such as job title, description, and requirements.

# Application: A record of a Job Seeker applying for a specific job, including information such as the user, job, and application date.

# Category: A grouping of jobs based on a specific industry or field, used to help users filter their search results.

# Authentication - The process of verifying a user's identity by checking their entered credentials (username and password) against the stored user information.

# Deliverable 2

## Domain Model

[Define the domain model and create the conceptual class diagrams]

## Architectural Design

### Conceptual Architecture

[Define the system’s conceptual architecture; use an architectural style and pattern - highlight its use and motivate your choice.]

### Package Design

[Create a package diagram]

### Component and Deployment Diagram

[Create the component and deployment diagrams.]

# Deliverable 3

## Design Model

### Dynamic Behavior

[Create the interaction diagrams (1 sequence, 1 communication diagrams) for 2 relevant scenarios]

### Class Diagram

[Create the UML class diagram; apply GoF patterns and motivate your choice]

## Data Model

[Create the data model for the system.]

# System Testing

[Describe the testing methides and some test cases.]

# Future Improvements

[Present some features that apply to the application scope.]

# Conclusion

# Bibliography