Project Title: Resume Creator Application

Prepared by: Navaneethan Ravi

Date: 16-12-2024

Introduction

1. *Project Overview*

The Resume Creator application is designed to allow users to create, manage, and update their resumes. The system provides functionality for users to input their details (personal, professional, and educational information) and generate a resume.

2. *Technology Stack:*

The application is built using

- **Backend**: Spring Boot (Java)
- **Database**: PostgreSQL
- **Frontend**: HTML, CSS, Thymeleaf
- **Testing**: JUnit, Mockito, Spring Test

System Design

1. *Architecture:*

The system follows a **Model-View-Controller (MVC)** architecture:

- **Controller**: Manages HTTP requests and responses. Handles CRUD operations for resumes.
- **Service**: Contains the business logic for handling resumes, including save, update, fetch, and delete functionalities.
- **Repository**: Interfaces with the database (PostgreSQL) using Spring Data JPA.

2. *Database Design:*

The application uses a relational database to store user data and resume information. The main entity is CV, which contains fields like id, name, email, phone, linkedin, etc.

Features of the Application

1. *User Registration and Login:*

Users can sign up and log in with their credentials. The system uses Spring Security for authentication and password hashing with BCrypt.

2. *Resume Creation:*

Users can fill in personal information, work experience, skills, and education. The resume is stored in the database and can be updated or deleted.

3. *Resume Editing and Updating:*

Users can edit their existing resumes and update fields such as name, contact details, and experiences.

4. *Export Resume to PDF:*

Users can export their resume to a PDF format for printing or online submission.

Implementation Details

1. *Backend Implementation:*

- **Controllers**: The *CvController* class handles HTTP requests for creating, fetching, updating, and deleting resumes. The *CvController* interacts with the service layer to execute business logic.
- **Services**: The *CvService* class implements methods for saving, retrieving, updating, and deleting resumes from the database.
- **Repositories**: The *CvRepository* extends the *JpaRepository* interface, providing database operations like *findById()*, *save()*, and *deleteById().*

2. *Frontend Implementation:*

Thymeleaf templates are used to render the views (HTML pages) for resume creation and editing. CSS is used for styling, and JavaScript handles front-end logic.

3. *Database Integration:*

The database is integrated using Spring Data JPA and PostgreSQL. The CV entity is mapped to a database table, and CRUD operations are implemented using JPA repository methods.

Folder Structure

- Provide a breakdown of the main project folders:
    - **src/main/java**: Contains Java code (controllers, services, repositories, models)
    - **src/main/resources**: Contains configuration files like application.properties and HTML templates.
    - **src/test/java**: Contains test files (unit tests and integration tests).

API Endpoints

- Provide a list of all the important API endpoints exposed by the application with descriptions.
    - **POST /cv**: Create a new resume.
    - **GET /cv/{id}**: Retrieve a resume by ID.
    - **PUT /cv/{id}**: Update an existing resume.
    - **DELETE /cv/{id}**: Delete a resume by ID.
    - **GET /cv/{id}/view**: View a resume in a browser (for rendering templates).

❖ Create Resume JSON


```
{

  "name": "Navaneethan",
  "email": "navanee@example.com",
  "linkedin" :"https://linkedin.com/in/navaneeravi",
  "phone": "7530087211",
  "careerObjective": [
    {
      "description": "Detail-oriented Java Full-Stack Developer with a strong
foundation in Core Java Programming, seeking a challenging Java Developer role."
    }
  ],
  "project": [
    {
      "title": "Online Food Delivery System",
      "description": "The Online Food Delivery System is a web-based application
developed "
    }
  ],
  "courses": [
    "Java Full Stack"
  ],
  "academicAchievements": [
```

```
    "Best Team Lead"
  ],
  "extracurricularActivities": [
    "kho kho state player",
    "1500 running district player"
  ],
  "educationList": [
    {
      "university": "K.L.N College of Engineering",
      "degree": "B.E Computer Science and Engineering",
      "graduationYear": 2023
    }
  ],
  "skills": [
    {
      "name": "Java"
    },
    {
      "name": "Spring Boot"
    },
    {
      "name": "MySql"
    },
    {
      "name": "HTML"
    },
    {
      "name": "CSS"
    },
    {
      "name": "JavaScript"
    }
  ],
  "workExperiences": [
    {
      "description": "Palle Technology  - Java Trainee"
    }
  ]
}
```

Workflow

1. ***User Registration Flow:***

- Describe how users can register and authenticate.

2. ***Resume Creation Flow:***

- Explain the steps for users to create a new resume by filling out forms and submitting them.

3. ***Resume Download Flow:***

- Describe how users can download the resume as a PDF after it is generated.

Testing

Unit Tests:

Unit tests are written to verify the functionality of individual components of the application. The tests are located in the *com.cv.resume.creator* package under the *service*, *controller*, and *repository* classes.

1. ***CvControllerUnitTests:***

   - **Test Methods**:
     - ***testCreateResume():*** Ensures that a resume is created correctly.
     - ***testGetResume():*** Verifies that a resume is fetched by ID.
     - ***testDeleteResume():*** Verifies that a resume can be deleted.

2. ***CvServiceUnitTests:***

   - **Test Methods**:
     - ***testSaveResume():*** Tests the saving of a resume.
     - ***testGetResumeById():*** Tests retrieving a resume by ID.
     - ***testUpdateResumeNotFound():*** Verifies behavior when updating a non-existent resume.
     - ***testDeleteResumeById():*** Ensures that the resume is deleted.

Integration Tests:

Integration tests are written to test the interaction between the controller, service, and repository layers.

3. *CvCreatorIntegrationTests:*

- **Test Methods**:
  - ***createCv():*** Verifies the creation of a resume through the HTTP POST request.
  - ***getCv():*** Verifies the fetching of a resume through the HTTP GET request.
  - ***updateCv():*** Verifies the update functionality.
  - ***deleteCv():*** Verifies the delete functionality.
  - ***getCvView():*** Tests the rendering of the resume view page.

Mocking and Assertions:

Mockito is used to mock the service and repository layers during unit tests. The *@Mock* and *@InjectMocks* annotations are used to mock dependencies and inject them into the test class.

Conclusion

The Resume Creator application enables users to easily generate and manage their resumes. The application follows industry-standard practices like MVC architecture, security, and RESTful web services. It also includes a robust testing suite that ensures the application works as expected. Future improvements could include integrating with LinkedIn APIs or adding more advanced resume templates.

References

- Spring Boot Documentation
- PostgreSQL Documentation
- JUnit Documentation