Case Study - Career Crafter API

CareerCrafterApplication.java

```
package com.careercrafter;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
@SpringBootApplication
public class CareerCrafterApplication {
    public static void main(String[] args) {
        SpringApplication.run(CareerCrafterApplication.class, args);
    }
}
```

User.java

```
package com.careercrafter.model;
import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Builder;
import lombok.Data;
import lombok.NoArgsConstructor;
import org.springframework.security.core.GrantedAuthority;
import org.springframework.security.core.authority.SimpleGrantedAuthority;
import org.springframework.security.core.userdetails.UserDetails;
import java.time.LocalDateTime;
import java.util.Collection;
import java.util.List;

@Data
@Builder
@NoArgsConstructor
@AllArgsConstructor
@Entity
@Table(name = "users")
public class User implements UserDetails {

        private static final long serialVersionUID = 1L;
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
```

```
private Long id;

@Column(unique = true, nullable = false)
private String email;

@Column(nullable = false)
private String password;

@Column(name = "first_name", nullable = false)
private String firstName;

@Column(name = "last_name", nullable = false)
private String lastName;

private String phone;

@Enumerated(EnumType.STRING)
@Column(nullable = false)
private Role role;

@Column(name = "created_date", updatable = false)
private LocalDateTime createdDate;

@Column(name = "updated_date")
private LocalDateTime updatedDate;

@Column(name = "is_active")
private boolean isActive;

@Override
public Collection<? extends GrantedAuthority> getAuthorities() {
    return List.of(new SimpleGrantedAuthority(role.name()));
}

@Override
public String getUsername() {
    return email;
}

@Override
public boolean isAccountNonExpired() {
```

```
        return true;
    }

    @Override
    public boolean isAccountNonLocked() {
        return true;
    }

    @Override
    public boolean isCredentialsNonExpired() {
        return true;
    }

    @Override
    public boolean isEnabled() {
        return isActive;
    }

    @PrePersist
    protected void onCreate() {
        createdDate = LocalDateTime.now();
        updatedDate = LocalDateTime.now();
        isActive = true;
    }

    @PreUpdate
    protected void onUpdate() {
        updatedDate = LocalDateTime.now();
    }

    public enum Role {
        JOB_SEEKER, EMPLOYER
    }
}
```

Job.java

```
package com.careercrafter.model;

import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Builder;
```

```
import lombok.Data;
import lombok.NoArgsConstructor;

import java.math.BigDecimal;
import java.time.LocalDateTime;

@Data
@Builder
@NoArgsConstructor
@AllArgsConstructor
@Entity
@Table(name = "jobs")
public class Job {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @Column(nullable = false)
    private String title;

    @Column(nullable = false, columnDefinition = "TEXT")
    private String description;

    @Column(columnDefinition = "TEXT")
    private String requirements;

    private String location;

    @Column(name = "salary_min", precision = 10, scale = 2)
    private BigDecimal salaryMin;

    @Column(name = "salary_max", precision = 10, scale = 2)
    private BigDecimal salaryMax;

    @Column(name = "job_type")
    private String jobType;

    @Column(name = "experience_level")
    private String experienceLevel;

    @Column(name = "company_name")
```

```
    private String companyName;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "employer_id", nullable = false)
    private User employer;

    @Column(name = "created_date", updatable = false)
    private LocalDateTime createdDate;

    @Column(name = "updated_date")
    private LocalDateTime updatedDate;

    @Column(name = "is_active")
    private boolean isActive;

    @PrePersist
    protected void onCreate() {
        createdDate = LocalDateTime.now();
        updatedDate = LocalDateTime.now();
        isActive = true;
    }

    @PreUpdate
    protected void onUpdate() {
        updatedDate = LocalDateTime.now();
    }
}
```

Resume.java

```
package com.careercrafter.model;

import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Builder;
import lombok.Data;
import lombok.NoArgsConstructor;

import java.time.LocalDateTime;

@Data
@Builder
@NoArgsConstructor
```

```
@AllArgsConstructor
@Entity
@Table(name = "resumes")
public class Resume {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "user_id", nullable = false)
    private User user;

    @Column(name = "file_name", nullable = false)
    private String fileName;

    @Column(name = "file_path", nullable = false)
    private String filePath;

    @Column(name = "file_size")
    private Long fileSize;

    @Column(name = "content_type")
    private String contentType;

    @Column(columnDefinition = "TEXT")
    private String skills;

    @Column(columnDefinition = "TEXT")
    private String experience;

    @Column(columnDefinition = "TEXT")
    private String education;

    @Column(name = "created_date", updatable = false)
    private LocalDateTime createdDate;

    @Column(name = "updated_date")
    private LocalDateTime updatedDate;

    @Column(name = "is_active")
    private boolean isActive;
```

```
    @PrePersist
    protected void onCreate() {
        createdDate = LocalDateTime.now();
        updatedDate = LocalDateTime.now();
        isActive = true;
    }

    @PreUpdate
    protected void onUpdate() {
        updatedDate = LocalDateTime.now();
    }
}
```

JobApplication.java

```
package com.careercrafter.model;
import jakarta.persistence.*;
import lombok.AllArgsConstructor;
import lombok.Builder;
import lombok.Data;
import lombok.NoArgsConstructor;
import java.time.LocalDateTime;
@Data
@Builder
@NoArgsConstructor
@AllArgsConstructor
@Entity
@Table(name = "job_applications")
public class JobApplication {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Long id;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "job_id", nullable = false)
    private Job job;

    @ManyToOne(fetch = FetchType.LAZY)
    @JoinColumn(name = "job_seeker_id", nullable = false)
    private User jobSeeker;

    @Column(name = "cover_letter", columnDefinition = "TEXT")
```

```
    private String coverLetter;

    @Enumerated(EnumType.STRING)
    @Column(nullable = false)
    private Status status;

    @Column(name = "applied_date", updatable = false)
    private LocalDateTime appliedDate;

    @Column(name = "updated_date")
    private LocalDateTime updatedDate;

    @PrePersist
    protected void onCreate() {
        appliedDate = LocalDateTime.now();
        updatedDate = LocalDateTime.now();
        status = Status.APPLIED;
    }

    @PreUpdate
    protected void onUpdate() {
        updatedDate = LocalDateTime.now();
    }

    public enum Status {
        APPLIED, REVIEWED, SHORTLISTED, REJECTED, ACCEPTED
    }
}
```

JobApplicationRepository.java

```
package com.careercrafter.repository;
import com.careercrafter.model.Job;
import com.careercrafter.model.JobApplication;
import com.careercrafter.model.User;
import org.springframework.data.jpa.repository.JpaRepository;
import java.util.List;
import java.util.Optional;

public interface JobApplicationRepository extends JpaRepository<JobApplication, Long> {
    List<JobApplication> findByJobSeeker(User jobSeeker);
    List<JobApplication> findByJob(Job job);
    Optional<JobApplication> findByJobAndJobSeeker(Job job, User jobSeeker);
```

```
    boolean existsByJobAndJobSeeker(Job job, User jobSeeker);
}
```

JobRepository.java

```
package com.careercrafter.repository;
import com.careercrafter.model.Job;
import com.careercrafter.model.User;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;
import java.util.List;
public interface JobRepository extends JpaRepository<Job, Long> {
    List<Job> findByEmployer(User employer);
    List<Job> findByIsActiveTrue();

    @Query("SELECT j FROM Job j WHERE " +
           "(:title IS NULL OR j.title LIKE %:title%) AND " +
           "(:location IS NULL OR j.location LIKE %:location%) AND " +
           "(:jobType IS NULL OR j.jobType = :jobType) AND " +
           "(:experienceLevel IS NULL OR j.experienceLevel = :experienceLevel) AND " +
           "j.isActive = true")
    List<Job> searchJobs(
            @Param("title") String title,
            @Param("location") String location,
            @Param("jobType") String jobType,
            @Param("experienceLevel") String experienceLevel);
}
```

ResumeRepository.java

```
package com.careercrafter.repository;
import com.careercrafter.model.Resume;
import com.careercrafter.model.User;
import org.springframework.data.jpa.repository.JpaRepository;
import java.util.List;
import java.util.Optional;
public interface ResumeRepository extends JpaRepository<Resume, Long> {
    List<Resume> findByUser(User user);
    Optional<Resume> findByUserAndIsActiveTrue(User user);
}
```

UserRepository.java

```
package com.careercrafter.repository;
import com.careercrafter.model.User;
import org.springframework.data.jpa.repository.JpaRepository;
import java.util.Optional;
public interface UserRepository extends JpaRepository<User, Long> {
    Optional<User> findByEmail(String email);
    boolean existsByEmail(String email);
}
```

AuthenticationService.java

```
package com.careercrafter.service;

import com.careercrafter.dto.LoginRequest;
import com.careercrafter.dto.RegisterRequest;
import com.careercrafter.dto.UserResponse;

import java.util.Map;

public interface AuthenticationService {
    UserResponse register(RegisterRequest request);
    Map<String, String> login(LoginRequest request);
    void logout();
}
```

JobApplicationService.java

```
package com.careercrafter.service;

import com.careercrafter.dto.JobApplicationRequest;
import com.careercrafter.dto.JobResponse;

import java.util.List;

public interface JobApplicationService {
    JobResponse applyForJob(JobApplicationRequest jobApplicationRequest);
    List<JobResponse> getApplicationsByJobSeeker(Long jobSeekerId);
    List<JobResponse> getApplicationsByJob(Long jobId);
    void withdrawApplication(Long applicationId);
    void updateApplicationStatus(Long applicationId, String status);
}
```

JobService.java

```
package com.careercrafter.service;
```

```
import com.careercrafter.dto.JobRequest;
import com.careercrafter.dto.JobResponse;

import java.util.List;

public interface JobService {
    JobResponse postJob(JobRequest jobRequest);
    List<JobResponse> getAllJobs();
    List<JobResponse> getJobsByEmployer(Long employerId);
    JobResponse getJobById(Long jobId);
    JobResponse updateJob(Long jobId, JobRequest jobRequest);
    void deactivateJob(Long jobId);
    List<JobResponse> searchJobs(String title, String location, String jobType, String experienceLevel);
}
```

ResumeService.java

```
package com.careercrafter.service;

import com.careercrafter.dto.ResumeRequest;
import com.careercrafter.dto.ResumeResponse;
import org.springframework.web.multipart.MultipartFile;

import java.util.List;

public interface ResumeService {
    ResumeResponse uploadResume(ResumeRequest resumeRequest, MultipartFile file);
    List<ResumeResponse> getResumesByUser(Long userId);
    ResumeResponse getActiveResumeByUser(Long userId);
    ResumeResponse updateResume(Long resumeId, ResumeRequest resumeRequest, MultipartFile file);
    void deactivateResume(Long resumeId);
}
```

UserService.java

```
package com.careercrafter.service;

import com.careercrafter.dto.RegisterRequest;
import com.careercrafter.dto.UserResponse;
import com.careercrafter.model.User;

import java.util.List;
```

```
public interface UserService {
    UserResponse registerUser(RegisterRequest registerRequest);
    List<UserResponse> getAllUsers();
    UserResponse getUserById(Long userId);
    void deactivateUser(Long userId);
    User getAuthenticatedUser();
}
```

GlobalExceptionHandler.java

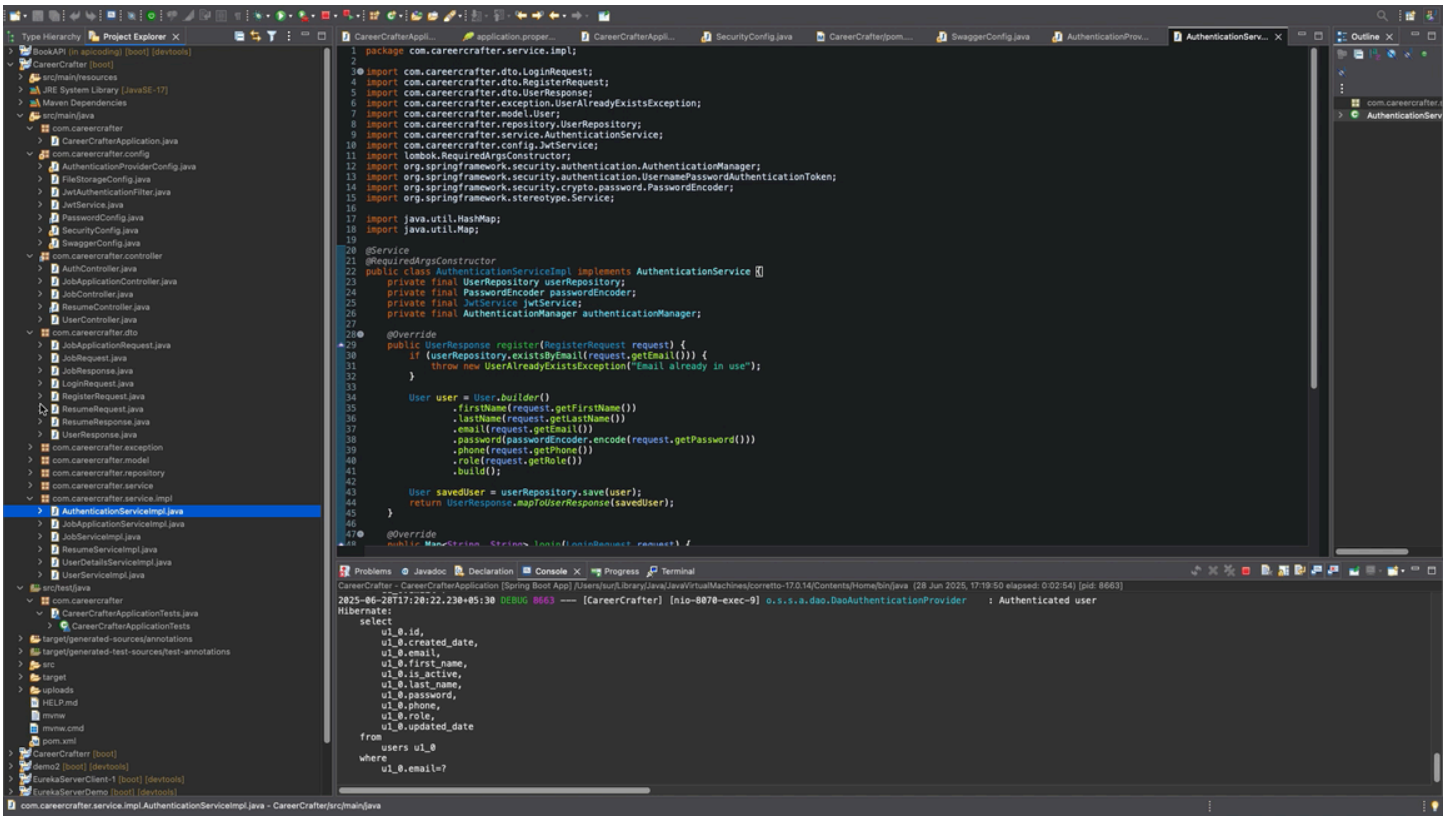
```
package com.careercrafter.exception;

import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.MethodArgumentNotValidException;
import org.springframework.web.bind.annotation.ExceptionHandler;
import org.springframework.web.bind.annotation.RestControllerAdvice;

import java.util.HashMap;
import java.util.Map;

@RestControllerAdvice
public class GlobalExceptionHandler {
    @ExceptionHandler(MethodArgumentNotValidException.class)
    public ResponseEntity<Map<String, String>>
 handleValidationExceptions(MethodArgumentNotValidException ex) {
        Map<String, String> errors = new HashMap<>();
        ex.getBindingResult().getFieldErrors().forEach(error -> {
            String fieldName = error.getField();
            String errorMessage = error.getDefaultMessage();
            errors.put(fieldName, errorMessage);
        });
        return ResponseEntity.badRequest().body(errors);
    }

    @ExceptionHandler(UserAlreadyExistsException.class)
    public ResponseEntity<String>
 handleUserAlreadyExistsException(UserAlreadyExistsException ex) {
        return ResponseEntity.status(HttpStatus.CONFLICT).body(ex.getMessage());
    }

    @ExceptionHandler(RuntimeException.class)
```

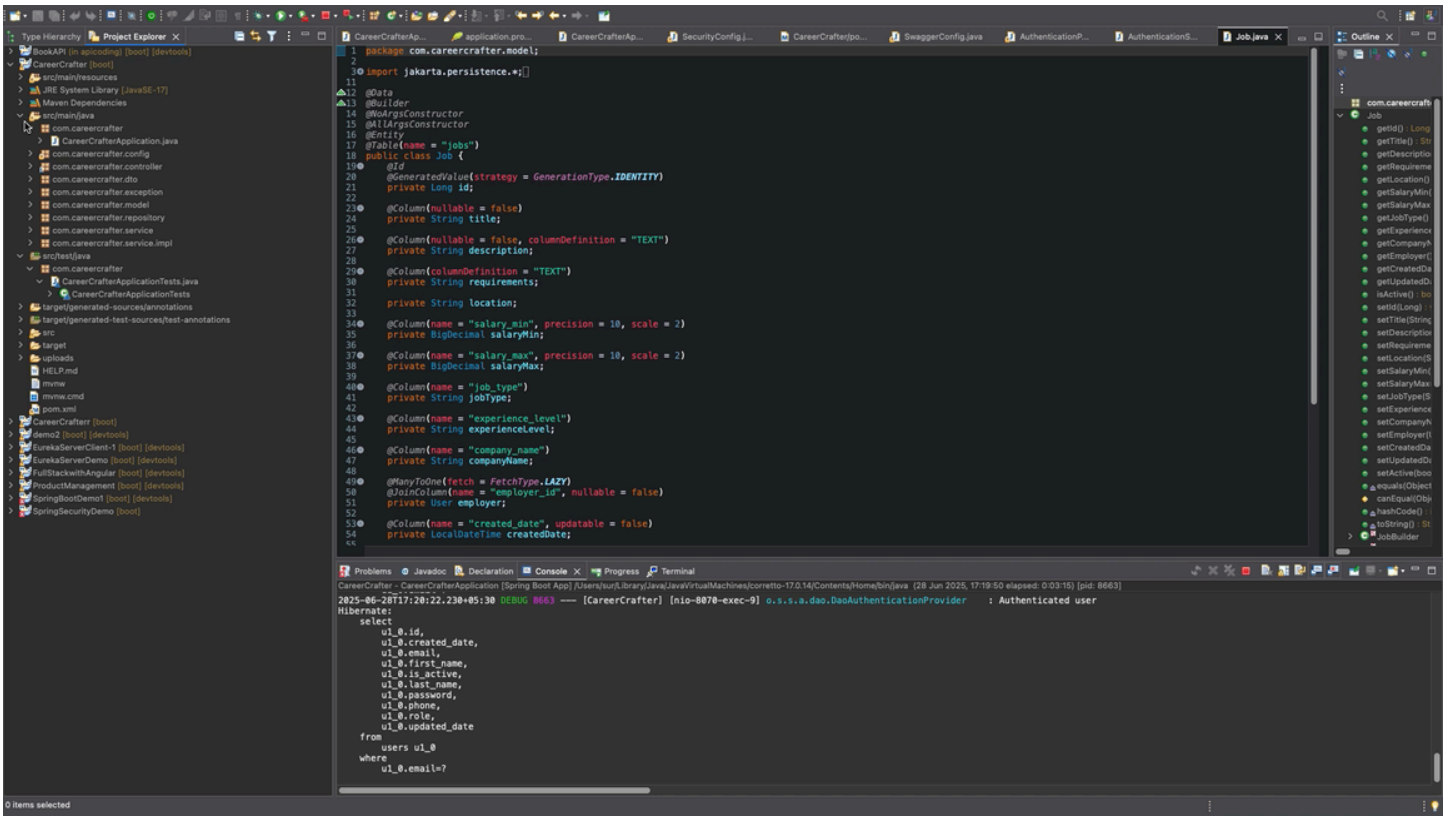
```
    public ResponseEntity<String> handleRuntimeException(RuntimeException ex) {
        return ResponseEntity.status(HttpStatus.BAD_REQUEST).body(ex.getMessage());
    }
}
```

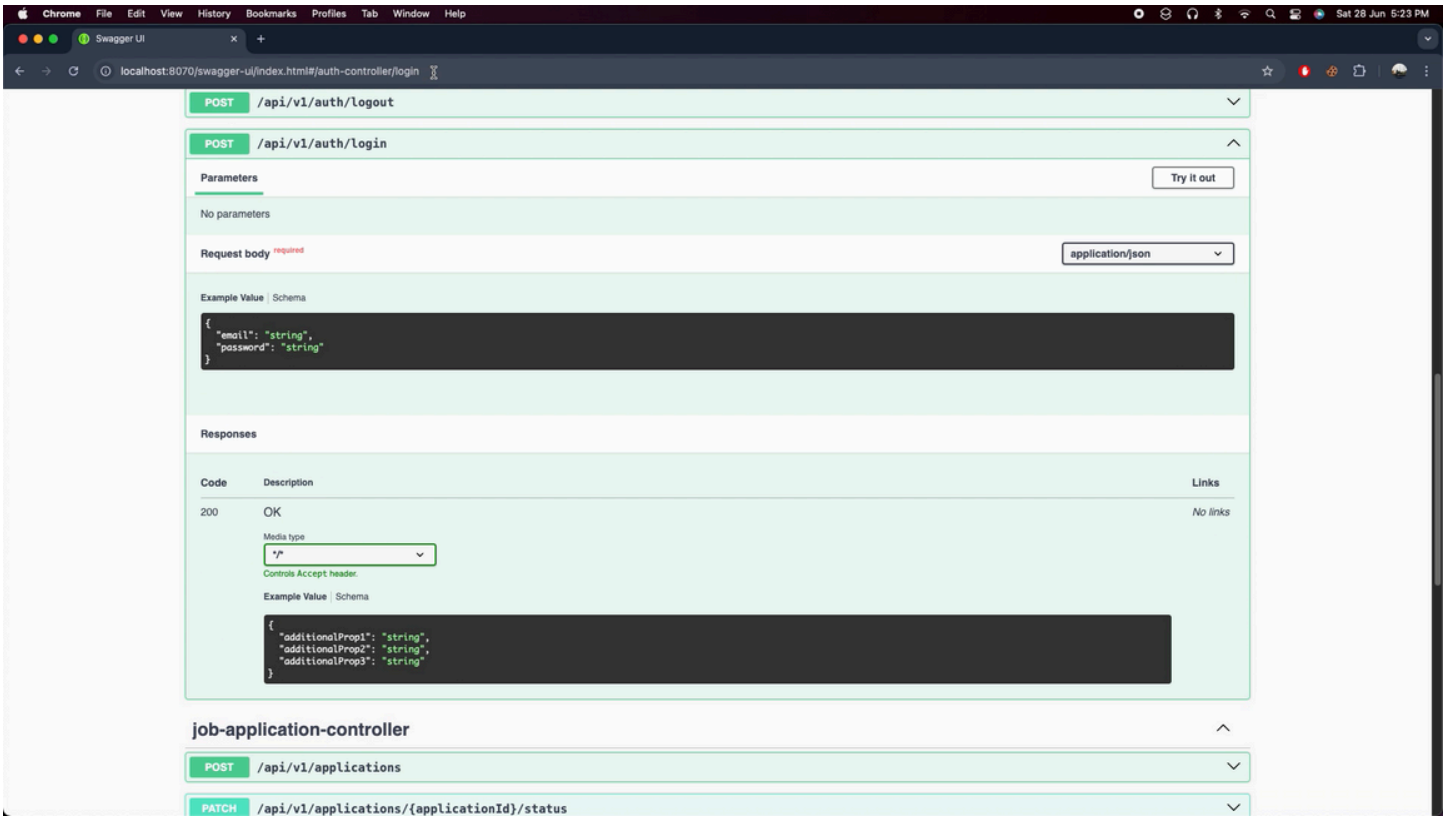
UserAlreadyExistsException.java

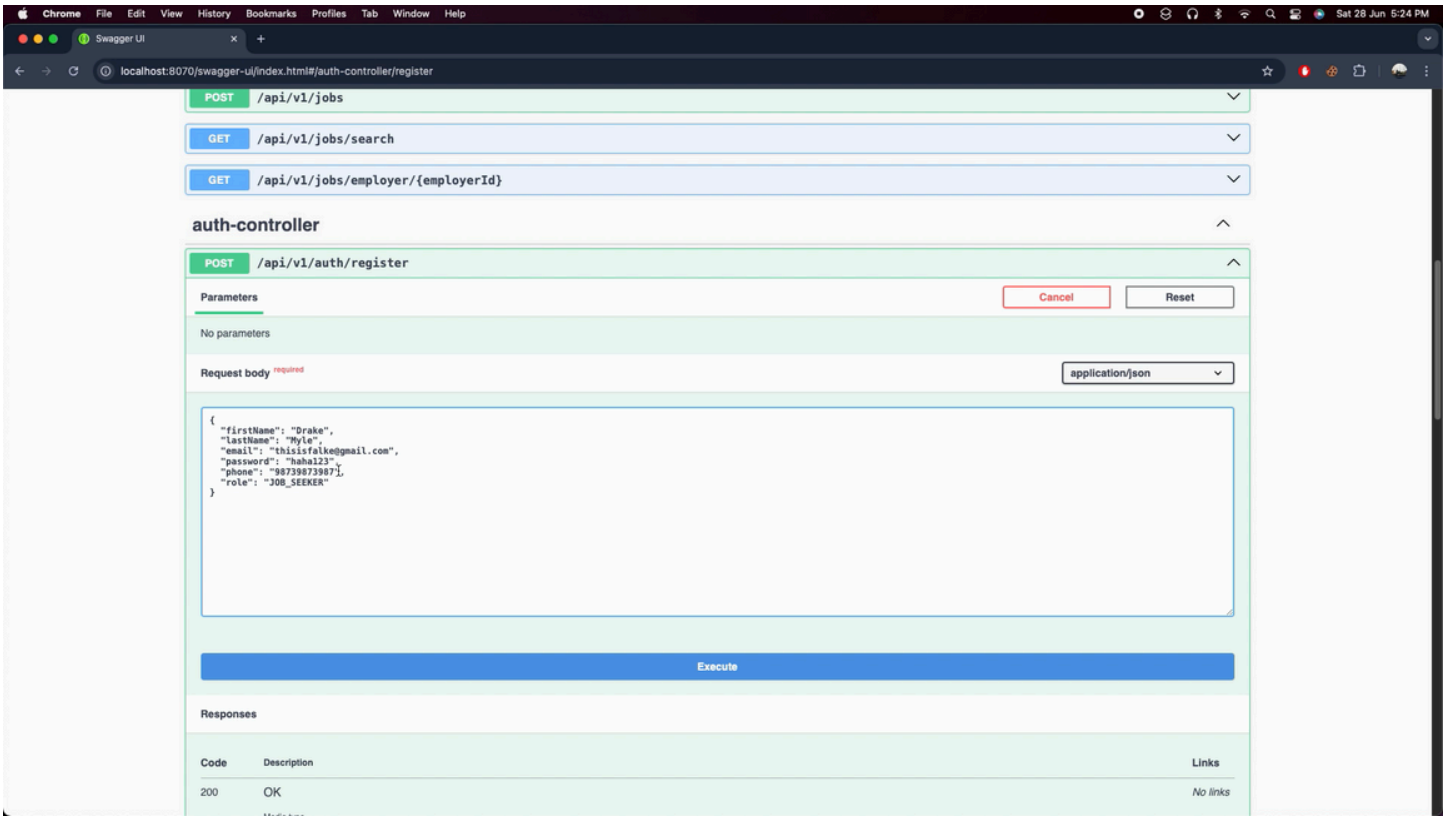
```
package com.careercrafter.exception;

public class UserAlreadyExistsException extends RuntimeException {
    private static final long serialVersionUID = 1L;
    public UserAlreadyExistsException(String message) {
        super(message);
    }
}
```

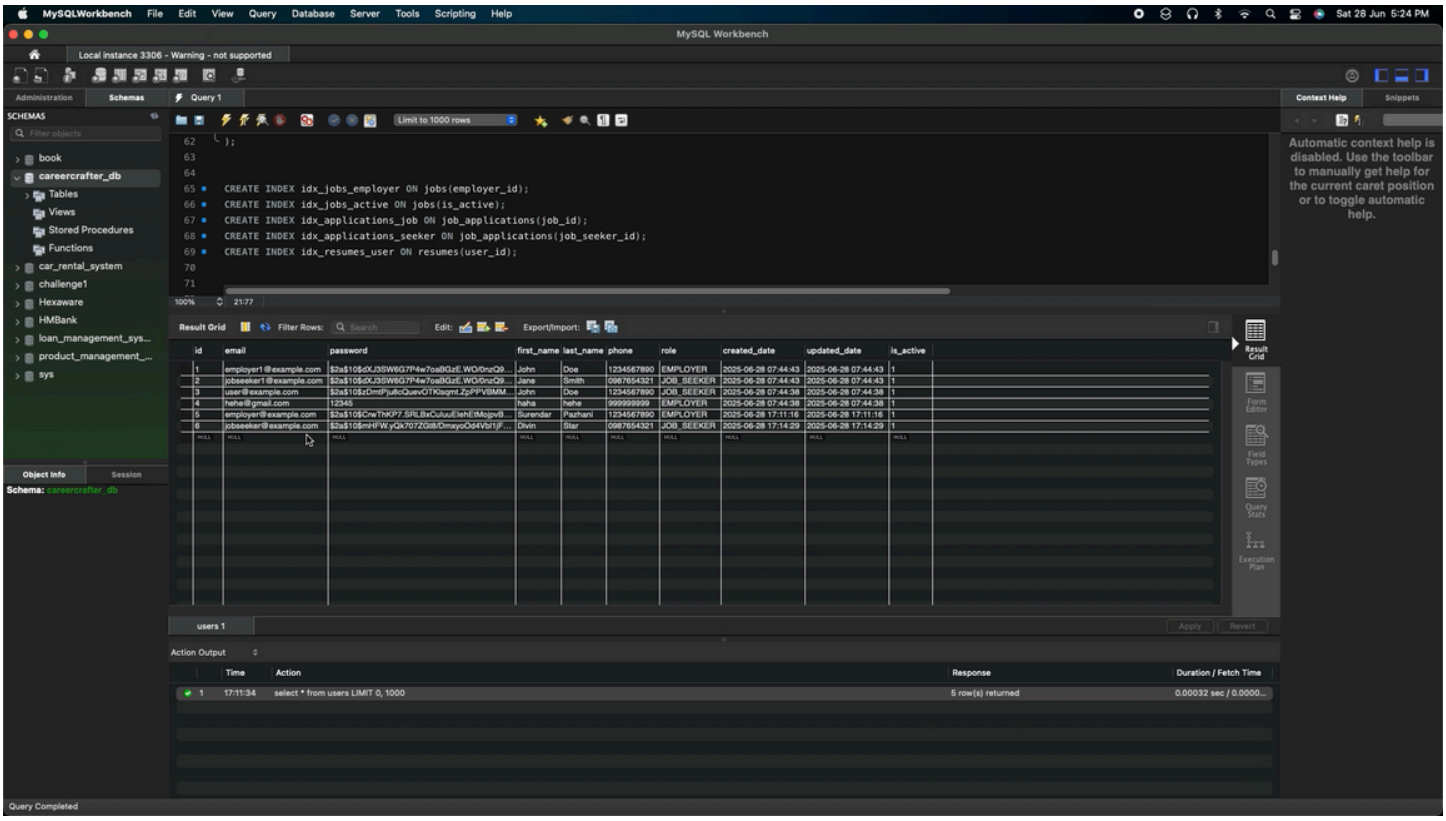
Screenshots:

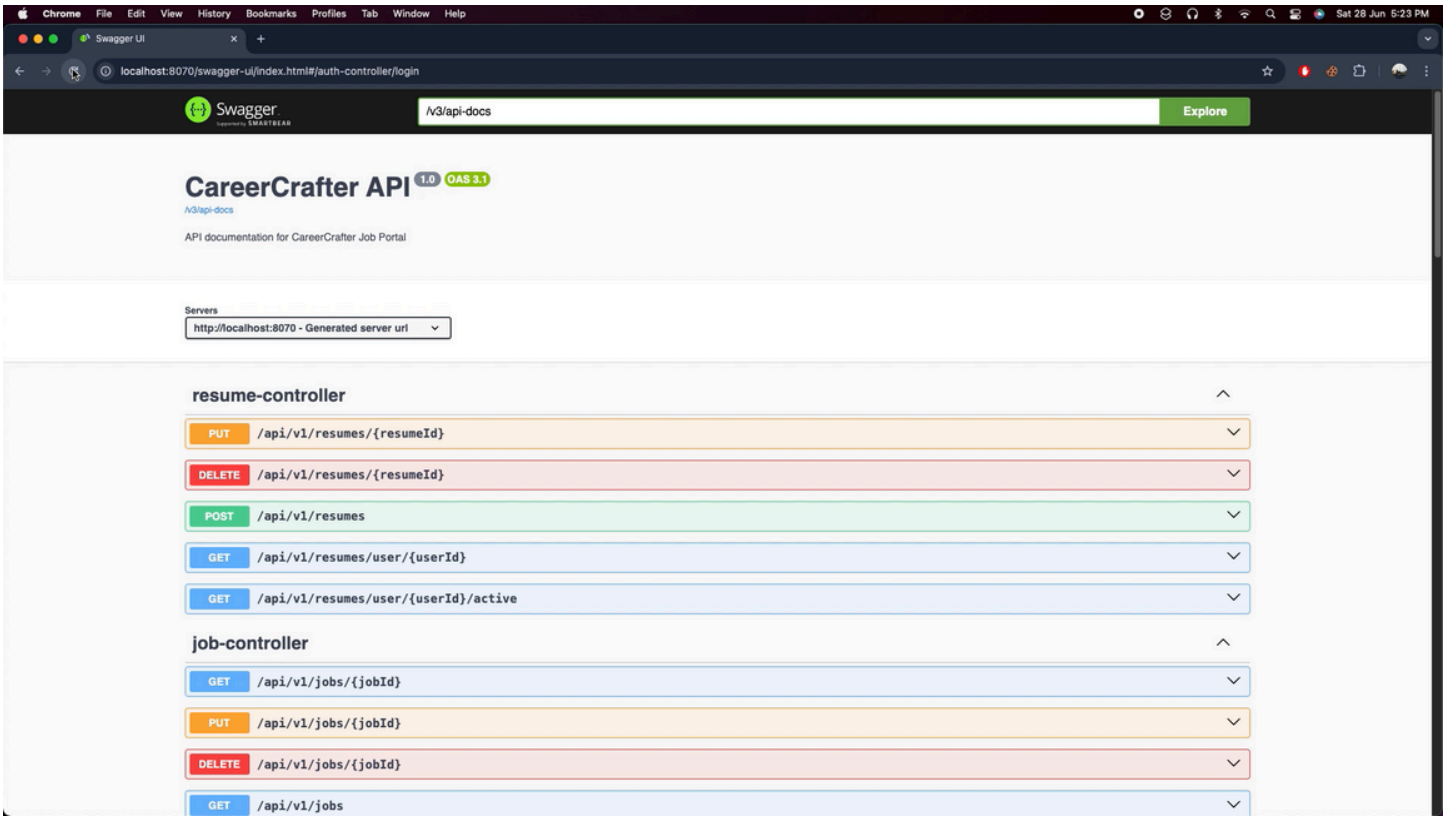



Job.java
package com.careercrafter.model;
import jakarta.persistence.*;
@Data
@Builder
@NoArgsConstructor
@AllArgsConstructor
@Entity
@Table(name = "jobs")
public class Job {
@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)
private Long id;
@Column(nullable = false)
private String title;
@Column(nullable = false, columnDefinition = "TEXT")
private String description;
@Column(columnDefinition = "TEXT")
private String requirements;
private String location;
@Column(name = "salary_min", precision = 10, scale = 2)
private BigDecimal salaryMin;
@Column(name = "salary_max", precision = 10, scale = 2)
private BigDecimal salaryMax;
@Column(name = "job_type")
private String jobType;
@Column(name = "experience_level")
private String experienceLevel;
@Column(name = "company_name")
private String companyName;
@ManyToOne(fetch = FetchType.LAZY)
@JoinColumn(name = "employer_id", nullable = false)
private User employer;
@Column(name = "created_date", updatable = false)
private LocalDateTime createdDate;
2025-06-28T17:20:22.230+05:30 DEBUG 8663 --- [CareerCrafter] [nio-8070-exec-9] o.s.s.a.dao.DaoAuthenticationProvider : Authenticated user
Hibernate:
select
u1_0.id,
u1_0.created_date,
u1_0.email,
u1_0.first_name,
u1_0.is_active,
u1_0.last_name,
u1_0.password,
u1_0.phone,
u1_0.role,
u1_0.updated_date
from
users u1_0
where
u1_0.email=?


Swagger UI
localhost:8070/swagger-ui/index.html#/auth-controller/login
POST /api/v1/auth/logout
POST /api/v1/auth/login
Parameters
Try it out
No parameters
Request body required
application/json
Example Value | Schema
{
"email": "string",
"password": "string"
}
Responses
Code
Description
Links
200
OK
No links
Media type
/
Controls Accept header.
Example Value | Schema
{
"additionalProp1": "string",
"additionalProp2": "string",
"additionalProp3": "string"
}
job-application-controller
POST /api/v1/applications
PATCH /api/v1/applications/{applicationId}/status


Swagger UI
localhost:8070/swagger-ui/index.html#/auth-controller/register
POST /api/v1/jobs
GET /api/v1/jobs/search
GET /api/v1/jobs/employer/{employerId}
auth-controller
POST /api/v1/auth/register
Parameters
Cancel
Reset
No parameters
Request body required
application/json
{
"firstName": "Drake",
"lastName": "Myle",
"email": "thisisfalke@gmail.com",
"password": "haha123",
"phone": "98739873987",
"role": "JOB_SEEKER"
}
Execute
Responses
Code
Description
Links
200
OK
No links


MySQL Workbench
Local instance 3306 - Warning - not supported
SCHEMAS
book
careercrafter_db
Tables
Views
Stored Procedures
Functions
car_rental_system
challenge1
Hexaware
HMBank
loan_management_sys...
product_management_...
sys
CREATE INDEX idx_jobs_employer ON jobs(employer_id);
CREATE INDEX idx_jobs_active ON jobs(is_active);
CREATE INDEX idx_applications_job ON job_applications(job_id);
CREATE INDEX idx_applications_seeker ON job_applications(job_seeker_id);
CREATE INDEX idx_resumes_user ON resumes(user_id);
Result Grid
id email password first_name last_name phone role created_date updated_date is_active
1 employer1@example.com $2a$10$dXJ3SW6G7P4w7oaBGzE.WO/0nzQ9... John Doe 1234567890 EMPLOYER 2025-06-28 07:44:43 2025-06-28 07:44:43 1
2 jobseeker1@example.com $2a$10$dXJ3SW6G7P4w7oaBGzE.WO/0nzQ9... Jane Smith 0987654321 JOB_SEEKER 2025-06-28 07:44:43 2025-06-28 07:44:43 1
3 user@example.com $2a$10$zDmdPju8cQuevOTKlsqmtLZpPPVBMM... John Doe 1234567890 JOB_SEEKER 2025-06-28 07:44:38 2025-06-28 07:44:38 1
4 hehe@gmail.com 12345 hahs hehe 999999999 EMPLOYER 2025-06-28 07:44:38 2025-06-28 07:44:38 1
5 employer@example.com $2a$10$CrwThKP7.SRLBxCuluuEIehEtMcjpv8... Surendar Pazhani 1234567890 EMPLOYER 2025-06-28 17:11:16 2025-06-28 17:11:16 1
6 jobseeker@example.com $2a$10$mHFW.yQk707ZGt8/DmxyoOd4Vbt1jF... Divin Star 0987654321 JOB_SEEKER 2025-06-28 17:14:29 2025-06-28 17:14:29 1
users 1
Action Output
Time Action Response Duration / Fetch Time
1 17:11:34 select * from users LIMIT 0, 1000 5 row(s) returned 0.00032 sec / 0.0000...
Query Completed


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Sat 28 Jun 5:23 PM
Swagger UI
localhost:8070/swagger-ui/index.html#/auth-controller/login
Swagger
/v3/api-docs
Explore
CareerCrafter API
1.0
OAS 3.1
/v3/api-docs
API documentation for CareerCrafter Job Portal
Servers
http://localhost:8070 - Generated server url
resume-controller
PUT /api/v1/resumes/{resumeId}
DELETE /api/v1/resumes/{resumeId}
POST /api/v1/resumes
GET /api/v1/resumes/user/{userId}
GET /api/v1/resumes/user/{userId}/active
job-controller
GET /api/v1/jobs/{jobId}
PUT /api/v1/jobs/{jobId}
DELETE /api/v1/jobs/{jobId}
GET /api/v1/jobs