**Source code for OnlineQuizPortal:**

1. **AdminController.java:**

package com.sp.qz;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.DeleteMapping;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.PutMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

@RequestMapping("/admin")

public class AdminController {

@Autowired

private QuestionService service;

@Autowired

private QuizService quizService;

///create new record

@PostMapping("/")

public ResponseEntity<Questions> addUser(@RequestBody Questions u){

Questions user= service.addUser(u);

if(user!=null)

return new ResponseEntity<Questions>(user,HttpStatus.CREATED);

else

return new ResponseEntity<Questions>(user, HttpStatus.INTERNAL\_SERVER\_ERROR);

}

//List of Users Or RETRIVE DATA

@GetMapping("/")

public List<Questions> getAllUser(){

return service.getAllUser();

}

//get user by id

@GetMapping("/{id}")

public ResponseEntity<Questions> getUserById(@PathVariable int id){

Questions user= service.getUserById(id);

if(user!=null)

return new ResponseEntity<Questions>(user,HttpStatus.FOUND);

else

return new ResponseEntity<Questions>(user,HttpStatus.NOT\_FOUND);

}

//update user by id

@PutMapping("/{id}")

public ResponseEntity<Object> updateUser(@RequestBody Questions user,@PathVariable int id){

Questions data= service.updateUser(user, id);

if(data!=null)

return new ResponseEntity<Object>(data,HttpStatus.OK);

else

return new ResponseEntity<Object>("User is Not Available",HttpStatus.NOT\_FOUND);

}

///DELETE USER BY ID

@DeleteMapping("/{id}")

public ResponseEntity<Object> deleteUser(@PathVariable int id ){

if(service.deleteUser(id))

return new ResponseEntity<Object>("User Deleted", HttpStatus.OK);

else

return new ResponseEntity<Object>("No User Found",HttpStatus.NOT\_FOUND);

}

///create new record

@PostMapping("/quiz")

public ResponseEntity<QuizCompetition> addUserr(@RequestBody QuizCompetition u){

QuizCompetition user= quizService.addUser(u);

if(user!=null)

return new ResponseEntity<QuizCompetition>(user,HttpStatus.CREATED);

else

return new ResponseEntity<QuizCompetition>(user, HttpStatus.INTERNAL\_SERVER\_ERROR);

}

//List of Users Or RETRIVE DATA

@GetMapping("/quiz")

public List<QuizCompetition> getAllUserr(){

return quizService.getAllUser();

}

}

1. **Answer.java:**

package com.sp.qz;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

@Entity

public class Answer {

@Id

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private int PId;

private int QuizId;

private int QuestionId;

private String Ans;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public int getPId() {

return PId;

}

public void setPId(int pId) {

PId = pId;

}

public int getQuizId() {

return QuizId;

}

public void setQuizId(int quizId) {

QuizId = quizId;

}

public int getQuestionId() {

return QuestionId;

}

public void setQuestionId(int questionId) {

QuestionId = questionId;

}

public String getAns() {

return Ans;

}

public void setAns(String ans) {

Ans = ans;

}

public Answer(int id, int pId, int quizId, int questionId, String ans) {

this.id = id;

this.PId = pId;

this.QuizId = quizId;

this.QuestionId = questionId;

this.Ans = ans;

}

public Answer(int pId, int quizId, int questionId, String ans) {

this.PId = pId;

this.QuizId = quizId;

this.QuestionId = questionId;

this.Ans = ans;

}

public Answer() {

}

}

1. **AnswerRepository.java:**

package com.sp.qz;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

@Repository

public interface AnswerRepository extends JpaRepository<Answer,Integer>{

}

1. **AnswerService.java:**

package com.sp.qz;

import java.util.List;

import javax.persistence.EntityManager;

import javax.persistence.PersistenceContext;

import javax.persistence.Query;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

@Service

public class AnswerService {

@Autowired

private AnswerRepository repo;

public Answer addAnswer(Answer u) {

return repo.save(u);

}

@PersistenceContext

EntityManager em ;

public List<Object[]> getResult(int pId, int quizId)

{

Query query = em.createQuery("select a.QuestionId, q.Answer , a.Ans , case when (q.Answer = a.Ans) then 'Correct' else 'Wrong' end "

+ "from Answer a, Questions q where (a.PId="+pId+" and a.QuizId="+quizId+") and a.QuestionId = q.id");

return query.getResultList();

}

}

1. **CustomAuthenticatorProvider.java:**

package com.sp.qz;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.security.authentication.\*;

import org.springframework.security.core.\*;

import org.springframework.security.core.authority.SimpleGrantedAuthority;

import org.springframework.security.crypto.password.PasswordEncoder;

import org.springframework.stereotype.Component;

import java.util.ArrayList;

import java.util.List;

import java.util.Optional;

@Component

public class CustomAuthenticationProvider implements AuthenticationProvider {

@Autowired

private UserRepo repo;

@Autowired

private PasswordEncoder pEncoder;

@Override

public Authentication authenticate(Authentication authentication) throws AuthenticationException {

String username = authentication.getName();

String password = authentication.getCredentials().toString();

List<Users> users = repo.findByName(username);

if (users.size() > 0) {

if (pEncoder.matches(password, users.get(0).getPassword())) {

List<GrantedAuthority> authorities = new ArrayList<>();

authorities.add(new SimpleGrantedAuthority(users.get(0).getRole()));

return new UsernamePasswordAuthenticationToken(username, password, authorities);

}else {

throw new BadCredentialsException("Invalid password");

}

} else {

throw new BadCredentialsException("No user registered with this details");

}

}

@Override

public boolean supports(Class<?> authentication) {

return authentication.equals(UsernamePasswordAuthenticationToken.class);

}

}

1. **ExamRepository.java:**

package com.sp.qz;

import java.util.List;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Query;

import org.springframework.stereotype.Repository;

@Repository

public interface ExamRepository extends JpaRepository<Questions,Integer>{

@Query("select new com.sp.qz.Questions(q.id,q.question,q.OptionA,q.OptionB,q.OptionC,q.OptionD) from Questions q inner join QuizCompetition qz on q.id=qz.questionId where qz.quizId=:id")

public List<Questions> getQuizById(int id);

}

1. **ExamService.java;**

package com.sp.qz;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

@Service

public class ExamService {

@Autowired

private ExamRepository repo;

public List<Questions> getQuizById(int id) {

return repo.getQuizById(id);

}

}

1. **GetAllQuizController.java:**

package com.sp.qz;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PathVariable;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

@RequestMapping("/protected")

public class GetAllQuizesController {

@Autowired

private AnswerService service;

@Autowired

private ExamService qservice;

@GetMapping("/{id}")

public List<Questions> getUserById(@PathVariable int id){

return qservice.getQuizById(id);

}

@PostMapping("/answer")

public ResponseEntity<Answer> saveAnswer(@RequestBody Answer a)

{

Answer answer = service.addAnswer(a);

if(answer!=null)

return new ResponseEntity<Answer>(answer,HttpStatus.CREATED);

else

return new ResponseEntity<Answer>(answer, HttpStatus.INTERNAL\_SERVER\_ERROR);

}

@GetMapping("/result/{pId}/{quizId}")

public ResponseEntity<Object> getResult(@PathVariable int pId, @PathVariable int quizId) {

List<Object[]> list = service.getResult(pId,quizId);

if(list!=null)

return new ResponseEntity<Object>(list,HttpStatus.FOUND);

else

return new ResponseEntity<Object>(list,HttpStatus.NOT\_FOUND);

}

}

1. **MainController.java:**

package com.sp.qz;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

public class MainController {

@RequestMapping("/")

public String hello(){

return "Hello World";

}

}

1. **OnlineQuizPortalApplication.java:**

package com.sp.qz;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.context.annotation.ComponentScan;

@SpringBootApplication

@ComponentScan("com.sp.qz")

public class OnlineQuizPortalApplication {

public static void main(String[] args) {

SpringApplication.run(OnlineQuizPortalApplication.class, args);

}

}

1. **QuestionRepo.java:**

package com.sp.qz;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

@Repository

public interface QuestionRepo extends JpaRepository<Questions, Integer>{

}

1. **Questions.java:**

package com.sp.qz;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

@Entity

public class Questions {

@Id

@Column(name="id")

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private String question;

private String OptionA;

private String OptionB;

private String OptionC;

private String OptionD;

private String Answer;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getQuestion() {

return question;

}

public void setQuestion(String question) {

this.question = question;

}

public String getOptionA() {

return OptionA;

}

public void setOptionA(String optionA) {

OptionA = optionA;

}

public String getOptionB() {

return OptionB;

}

public void setOptionB(String optionB) {

OptionB = optionB;

}

public String getOptionC() {

return OptionC;

}

public void setOptionC(String optionC) {

OptionC = optionC;

}

public String getOptionD() {

return OptionD;

}

public void setOptionD(String optionD) {

OptionD = optionD;

}

public String getAns() {

return Answer;

}

public void setAnswer(String ans) {

Answer = ans;

}

public Questions(String question, String optionA, String optionB, String optionC, String optionD, String answer) {

this.question = question;

OptionA = optionA;

OptionB = optionB;

OptionC = optionC;

OptionD = optionD;

Answer = answer;

}

public Questions() {

}

public Questions(int id, String question, String optionA, String optionB, String optionC, String optionD) {

this.id = id;

this.question = question;

OptionA = optionA;

OptionB = optionB;

OptionC = optionC;

OptionD = optionD;

}

}

1. **QuestionService.java:**

package com.sp.qz;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

@Service

public class QuestionService {

@Autowired

private QuestionRepo repo;

//add method or CREATE RECORD

public Questions addUser(Questions u) {

return repo.save(u);

}

//List user Method

public List<Questions> getAllUser(){

return repo.findAll();

}

//get user by id

public Questions getUserById(int id) {

if(repo.findById(id).isPresent())

return repo.findById(id).get();

else

return null;

}

//update user by id

public Questions updateUser(Questions user, int id) {

if(repo.findById(id).isPresent())

{

Questions old= repo.findById(id).get();

old.setOptionA(user.getOptionA());

old.setOptionB(user.getOptionB());

old.setOptionC(user.getOptionC());

old.setOptionD(user.getOptionD());

old.setAnswer(user.getAns());

return repo.save(old);

}

else

{

return null;

}

}

///DELETE USER BY ID

public boolean deleteUser(int id) {

if(repo.findById(id).isPresent())

{

repo.deleteById(id);

return true;

}

else

return false;

}

}

1. **QuizCompetition.java:**

package com.sp.qz;

import javax.persistence.Column;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

@Entity

public class QuizCompetition {

@Id

@Column(name="id")

@GeneratedValue(strategy = GenerationType.IDENTITY)

private int id;

private int quizId;

private int questionId;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public int getQuizId() {

return quizId;

}

public void setQuizId(int quizId) {

this.quizId = quizId;

}

public int getQuestionId() {

return questionId;

}

public void setQuestionId(int questionId) {

this.questionId = questionId;

}

public QuizCompetition(int id, int quizId, int questionId) {

this.id = id;

this.quizId = quizId;

this.questionId = questionId;

}

public QuizCompetition() {

}

}

1. **QuizRepo.java:**

package com.sp.qz;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.stereotype.Repository;

@Repository

public interface QuizRepo extends JpaRepository<QuizCompetition, Integer> {

}

1. QuizService.java:

package com.sp.qz;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

@Service

public class QuizService {

@Autowired

private QuizRepo repo;

//add method or CREATE RECORD

public QuizCompetition addUser(QuizCompetition u) {

return repo.save(u);

}

//List user Method

public List<QuizCompetition> getAllUser(){

return repo.findAll();

}

}

1. **RegisterController.java:**

package com.sp.qz;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.http.HttpStatus;

import org.springframework.http.ResponseEntity;

import org.springframework.web.bind.annotation.GetMapping;

import org.springframework.web.bind.annotation.PostMapping;

import org.springframework.web.bind.annotation.RequestBody;

import org.springframework.web.bind.annotation.RequestMapping;

import org.springframework.web.bind.annotation.RestController;

@RestController

@RequestMapping("/register")

public class RegisterController {

@Autowired

private RegisterService service;

///create new record

@PostMapping("/")

public ResponseEntity<Users> addUser(@RequestBody Users u){

Users user= service.addUser(u);

if(user!=null)

return new ResponseEntity<Users>(user,HttpStatus.CREATED);

else

return new ResponseEntity<Users>(user, HttpStatus.INTERNAL\_SERVER\_ERROR);

}

@GetMapping("/")

public List<Users> getAllUser(){

return service.getAllUser();

}

}

1. **RegisterRepo.java:**

package com.sp.qz;

import org.springframework.data.jpa.repository.JpaRepository;

public interface RegisterRepo extends JpaRepository<Users,Integer>{

}

1. **RegisterService.java:**

package com.sp.qz;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.stereotype.Service;

@Service

public class RegisterService {

@Autowired

private RegisterRepo repo;

//add method or CREATE RECORD

public Users addUser(Users u) {

return repo.save(u);

}

public List<Users> getAllUser() {

// TODO Auto-generated method stub

return repo.findAll();

}

}

1. **SpringSecurityCOnfig.java:**

package com.sp.qz;

import org.springframework.context.annotation.Bean;

import org.springframework.context.annotation.Configuration;

import org.springframework.security.config.annotation.authentication.builders.AuthenticationManagerBuilder;

import org.springframework.security.config.annotation.web.builders.HttpSecurity;

import org.springframework.security.config.annotation.web.configuration.WebSecurityConfigurerAdapter;

import org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;

import org.springframework.security.crypto.password.PasswordEncoder;

@Configuration

public class SpringSecurityConfig extends WebSecurityConfigurerAdapter{

@Override

protected void configure(HttpSecurity http) throws Exception {

http.httpBasic().and().authorizeRequests()

.antMatchers("/").permitAll()

.antMatchers("/protected").hasRole("USER")

.antMatchers("/admin").hasRole("ADMIN")

.antMatchers("/register").permitAll();

http.cors().and().csrf().disable();

}

@Bean

public PasswordEncoder passwordEncoder () {

return new BCryptPasswordEncoder();

}

}

1. **UserRepo.java:**

package com.sp.qz;

import java.util.List;

import org.springframework.data.jpa.repository.JpaRepository;

import org.springframework.data.jpa.repository.Query;

import org.springframework.security.core.userdetails.UserDetails;

import org.springframework.stereotype.Repository;

@Repository

public interface UserRepo extends JpaRepository<Users,Integer>{

List<Users> findByName(String name);

}

1. **Users.java:**

package com.sp.qz;

import javax.persistence.Entity;

import javax.persistence.GeneratedValue;

import javax.persistence.GenerationType;

import javax.persistence.Id;

@Entity

public class Users{

@Id

@GeneratedValue(strategy=GenerationType.IDENTITY)

private int id;

private String name;

private String password;

private String role;

public int getId() {

return id;

}

public void setId(int id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public String getPassword() {

return password;

}

public void setPassword(String password) {

this.password = password;

}

public String getRole() {

return role;

}

public void setRole(String role) {

this.role = role;

}

}

1. **UserService.java:**

package com.sp.qz;

import java.util.ArrayList;

import java.util.Collection;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;

import org.springframework.security.core.GrantedAuthority;

import org.springframework.security.core.authority.SimpleGrantedAuthority;

import org.springframework.security.core.userdetails.UserDetails;

import org.springframework.security.core.userdetails.UserDetailsService;

import org.springframework.security.core.userdetails.UsernameNotFoundException;

import org.springframework.stereotype.Service;

public class UserService implements UserDetails {

private static final long serialVersionUID = 1L;

private final Users users;

public UserService (Users users) {

this.users = users;

}

@Override

public Collection<? extends GrantedAuthority> getAuthorities() {

List<GrantedAuthority> authorities = new ArrayList<>();

authorities.add(new SimpleGrantedAuthority(users.getRole()));

return authorities;

}

@Override

public String getPassword() {

return users.getPassword();

}

@Override

public String getUsername() {

return users.getName();

}

@Override

public boolean isAccountNonExpired() {

// TODO Auto-generated method stub

return true;

}

@Override

public boolean isAccountNonLocked() {

// TODO Auto-generated method stub

return true;

}

@Override

public boolean isCredentialsNonExpired() {

// TODO Auto-generated method stub

return true;

}

@Override

public boolean isEnabled() {

// TODO Auto-generated method stub

return true;

}

}