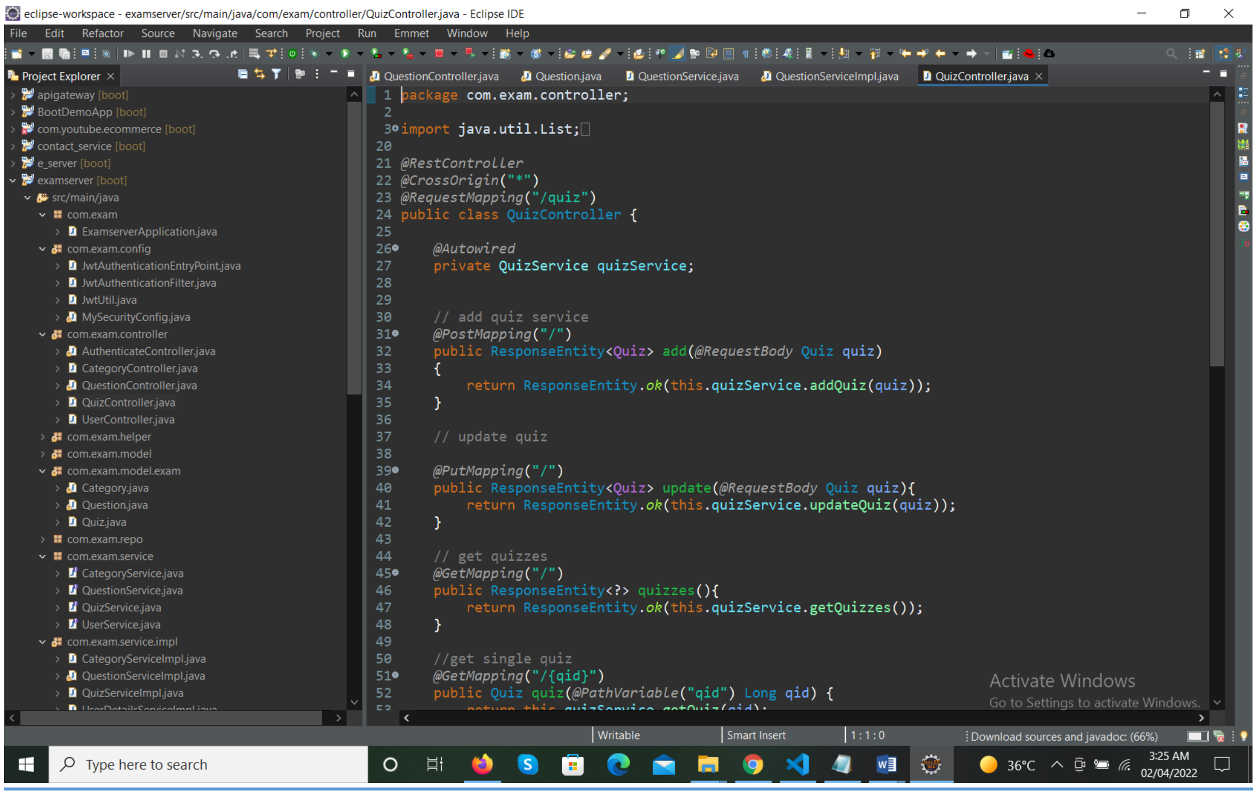
SourceCode: For online Test application

GitHub link:- https://github.com/rushighale/Online-Test-Application



```java
package com.exam.controller;

import java.util.List;

@RestController
@CrossOrigin("*")
@RequestMapping("/quiz")
public class QuizController {

    @Autowired
    private QuizService quizService;


    // add quiz service
    @PostMapping("/")
    public ResponseEntity<Quiz> add(@RequestBody Quiz quiz)
    {
        return ResponseEntity.ok(this.quizService.addQuiz(quiz));
    }

    // update quiz

    @PutMapping("/")
    public ResponseEntity<Quiz> update(@RequestBody Quiz quiz){
        return ResponseEntity.ok(this.quizService.updateQuiz(quiz));
    }

    // get quizzes
    @GetMapping("/")
    public ResponseEntity<?> quizzes(){
        return ResponseEntity.ok(this.quizService.getQuizzes());
    }

    //get single quiz
    @GetMapping("/{qid}")
    public Quiz quiz(@PathVariable("qid") Long qid) {
        return this.quizService.getQuiz(qid);
```

1.)AuthenticationController.java

package com.exam.controller;

import java.security.Principal;

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

import org.springframework.http.ResponseEntity;

import org.springframework.security.authentication.AuthenticationManager;

```
import
org.springframework.security.authentication.BadCredentialsException;
import org.springframework.security.authentication.DisabledException;
import
org.springframework.security.authentication.UsernamePasswordAuthe
nticationToken;
import org.springframework.security.core.userdetails.UserDetails;
import
org.springframework.security.core.userdetails.UserDetailsService;
import
org.springframework.security.core.userdetails.UsernameNotFoundExce
ption;
import org.springframework.web.bind.annotation.CrossOrigin;
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.RestController;

import com.exam.config.JwtUtil;
import com.exam.helper.UserNotFoundException;
import com.exam.model.JwtRequest;
import com.exam.model.JwtResponse;
import com.exam.model.User;
```

```
@RestController
@CrossOrigin("*")
public class AuthenticateController {

        @Autowired
        private AuthenticationManager authenticationManager;
        @Autowired
        private UserDetailsService userDetailsService;
        @Autowired
        private JwtUtil jwtUtil;

        // generate token
        @PostMapping("/generate-token")
        public ResponseEntity<?> generateToken(@RequestBody
JwtRequest jwtRequest) throws Exception{
                try {



        authenticate(jwtRequest.getUsername(),jwtRequest.getPassword
());



                }catch(UserNotFoundException e) {
                    e.printStackTrace();

                    throw  new Exception("user not found ");

            }



            ////////////authenticate


            UserDetails
userDetails=this.userDetailsService.loadUserByUsername(jwtRequest.g
etUsername());

            String token =this.jwtUtil.generateToken(userDetails);

            return ResponseEntity.ok(new JwtResponse(token));

        }




        private void authenticate(String username,String password)
throws Exception  {


            try {




                    authenticationManager.authenticate(new
UsernamePasswordAuthenticationToken(username, password));
```

```
		}catch(DisabledException e) {
			throw new Exception("USER DISABLED 
"+e.getMessage());
		}catch(BadCredentialsException e) {
			throw new Exception("Invalid Credentials 
"+e.getMessage());
		}
	}

	@GetMapping("/current-user")
	public User getCurrentUser(Principal principal) {
		return (User) 
this.userDetailsService.loadUserByUsername(principal.getName());
	}
}
```

2.)CategoryController.java

```
package com.exam.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.CrossOrigin;
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;

import com.exam.model.exam.Category;
import com.exam.service.CategoryService;

@RestController
@RequestMapping("/category")
@CrossOrigin("*")
public class CategoryController {

    @Autowired
    private CategoryService categoryService;



    //add category
```

```
	@PostMapping("/")
	public ResponseEntity<?> addCategory(@RequestBody Category 
category){
		Category 
category1=this.categoryService.addCategory(category);

		return ResponseEntity.ok(category1);

	}

	// getcategory
	@GetMapping("/{categoryId}")
	public Category getCategory(@PathVariable("categoryId") Long 
categoryId)
	{
		return this.categoryService.getCategory(categoryId);
	}

	// get all categories
	@GetMapping("/")
	public ResponseEntity<?> getCategories(){
		return 
ResponseEntity.ok(this.categoryService.getCategories());
```

```
	}

	//update category
	@PutMapping("/")
	public Category updateCategory(@RequestBody Category 
category)
	{
		return this.categoryService.updateCategory(category);
	}

	// delete category
	@DeleteMapping("/{categoryId}")
	public void deleteCategory(@PathVariable("categoryId") Long 
categoryId) {
		this.categoryService.deleteCategory(categoryId);
	}



}
```

3.)QuestionController.java

```
package com.exam.controller;
```

```
import java.util.ArrayList;
import java.util.Collection;
import java.util.Collections;
import java.util.List;
import java.util.Map;
import java.util.Set;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.CrossOrigin;
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;

import com.exam.model.exam.Question;
import com.exam.model.exam.Quiz;
import com.exam.service.QuestionService;
import com.exam.service.QuizService;

@RestController
@CrossOrigin("*")
@RequestMapping("/question")
public class QuestionController {

        @Autowired
        private QuestionService service;
        @Autowired
        private QuizService quizService;

        // add question
        @PostMapping("/")
        public ResponseEntity<Question> add(@RequestBody Question
question)
        {
                return
ResponseEntity.ok(this.service.addQuestion(question));
        }

        // update the question
        @PutMapping("/")
```

```
	public ResponseEntity<Question> update (@RequestBody
Question question){
		return
ResponseEntity.ok(this.service.upateQuestion(question));
	}

	// get all question

	// get all question of any quiz
	@GetMapping("/quiz/{qid}")
	public ResponseEntity<?>
getQuestionsOfQuiz(@PathVariable("qid") Long qid){
//		Quiz quiz=new Quiz();
//		quiz.setqId(qid);
//		Set<Question> questionsOfQuiz=
this.service.getQuestionsOfQuiz(quiz);
//		return ResponseEntity.ok(questionsOfQuiz);

		Quiz quiz=this.quizService.getQuiz(qid);
		Set<Question> questions=quiz.getQuestions();
		List <Question> list=new ArrayList(questions);
		if(list.size() >
Integer.parseInt(quiz.getNumberOfQuestions())) {
```

```
                    list=list.subList(0,
Integer.parseInt(quiz.getNumberOfQuestions()+1));


            }
            list.forEach((q)->{
                    q.setAnswer("");
            });



            Collections.shuffle(list);
            return ResponseEntity.ok(list);
    }



    @GetMapping("/quiz/all/{qid}")
    public ResponseEntity<?>
getQuestionsOfQuizAdmin(@PathVariable("qid") Long qid){
            Quiz quiz=new Quiz();
            quiz.setqId(qid);
            Set<Question> questionsOfQuiz=
this.service.getQuestionsOfQuiz(quiz);
            return ResponseEntity.ok(questionsOfQuiz);
    }
```

```
	//get single question
	@GetMapping("/{quesId}")
	public Question get(@PathVariable("quesId") Long quesId) {
		return this.service.getQuestion(quesId);
	}



	//delete question
	@DeleteMapping("/{quesId}")
	public void delete(@PathVariable("quesId") Long quesId)
	{
		this.service.deleteQuestion(quesId);
	}


	// eval quiz
	@PostMapping("/eval-quiz")
	public ResponseEntity<?> evalQuiz(@RequestBody List<Question> 
questions){
		System.out.println(questions);
		    Double   marksGot = (double) 0 ;
```

```
                int     correctAnswers = 0;

                int attempted = 0;






        for(Question q:questions){

                // single questions

            Question question=  this.service.get(q.getQuesId());

            if(question.getAnswer().equals(q.getGivenAnswer())) {

                 // correct

                 correctAnswers++;


                double marksSingle =
Double.parseDouble(questions.get(0).getQuiz().getMaxMarks())/questi
ons.size();

                                marksGot += marksSingle;




            }


            if (q.getGivenAnswer()!=null ){
                    attempted++;
                    //
                }
            };

            Map<String, Object>
map=Map.of("marksGot",marksGot,"correctAnswer",correctAnswers,"
attempted",attempted);
            return ResponseEntity.ok(map);
    }



}
```

<u>4.) QuizController.java</u>

```
package com.exam.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.CrossOrigin;
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;

import com.exam.model.exam.Category;
import com.exam.model.exam.Quiz;
import com.exam.service.QuizService;

@RestController
@CrossOrigin("*")
@RequestMapping("/quiz")
public class QuizController {

	@Autowired
	private QuizService quizService;



	// add quiz service
```

```
@PostMapping("/")
public ResponseEntity<Quiz> add(@RequestBody Quiz quiz)
{
	return ResponseEntity.ok(this.quizService.addQuiz(quiz));
}

// update quiz

@PutMapping("/")
public ResponseEntity<Quiz> update(@RequestBody Quiz quiz){
	return ResponseEntity.ok(this.quizService.updateQuiz(quiz));
}

// get quizzes
@GetMapping("/")
public ResponseEntity<?> quizzes(){
	return ResponseEntity.ok(this.quizService.getQuizzes());
}

//get single quiz
@GetMapping("/{qid}")
public Quiz quiz(@PathVariable("qid") Long qid) {
```

```
            return this.quizService.getQuiz(qid);

        }


        // delete the quiz


        @DeleteMapping("/{qid}")


        public void delete(@PathVariable("qid") Long qid) {

            this.quizService.deleteQuiz(qid);

        }




        //

        @GetMapping("/category/{cid}")

        public List<Quiz> getQuizzesOfCategory(@PathVariable("cid")
    Long cid){

            Category  category= new Category();

            category.setCid(cid);

            return this.quizService.getQuizzesOfCategory(category);

        }
        // get active quizzes


        @GetMapping("/active")

        public List<Quiz> getActiveQuizzes(){

        return this.quizService.getActiveQuizzes();


        }


// get active quizzes of category

        @GetMapping("category/active/{cid}")

public List<Quiz> getActiveQuizzesOfCategory(@PathVariable("cid")
Long cid){

        Category category=new Category();

                category.setCid(cid);

                return
this.quizService.getActiveQuizzesofCategory(category);


}
```

```
}
```

5.) UserController.java :

```
package com.exam.controller;

import java.util.HashSet;
import java.util.Set;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import
org.springframework.security.crypto.bcrypt.BCryptPasswordEncoder;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.ExceptionHandler;
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;
import com.exam.helper.UserFoundException;
import com.exam.model.Role;
import com.exam.model.User;
import com.exam.model.UserRole;
import com.exam.service.UserService;

@RestController
@RequestMapping("/user")
@CrossOrigin("*")
public class UserController {



        @Autowired
        private UserService userService;
        @Autowired
        private BCryptPasswordEncoder bCryptPasswordEncoder;

        // creating user                    // we can return direct  responsentity
oruser
        @PostMapping("/")
        public User createUser(@RequestBody User user) throws
Exception {
```

```
                user.setProfile("defualt.png");
                // encoding pssword with bcryptpasswordencoder

        user.setPassword(this.bCryptPasswordEncoder.encode(user.getPa
ssword()));



                Set<UserRole> roles=new HashSet<>();


                Role role=new Role();
                role.setRoleId(45L);
                role.setRoleName("NORMAL");                // we applied
restriction jo bhi ayega vo normal user add hoga



                UserRole userRole=new UserRole();
                userRole.setUser(user);
                userRole.setRole(role);



                roles.add(userRole);
```

```
            return this.userService.createUser(user, roles);

        }


        // get the user
        @GetMapping("/{username}")
        public User getUser(@PathVariable("username") String
username) {


            return this.userService.getUser(username);
        }


        //delete user by id
        @DeleteMapping("/{userId}")
        public void deleteUser(@PathVariable("userId") Long userId) {
            this.userService.deleteUser(userId);


        }
        // update the user





        @ExceptionHandler(UserFoundException.class)
        public ResponseEntity<?> exceptionHandler(UserFoundException
ex){
                return ResponseEntity.ok(ex.getMessage());
        }




}
```