


QuizService SpringBoot

👤 Created by	 Deepak Ks
🕒 Created time	@January 19, 2025 1:50 PM
☰ Project	SpringBoot
📄 Type	Documentation

Spring Boot Project Documentation

Configuration

application.properties

```
properties
CopyEdit
spring.application.name=Quiz-App
spring.datasource.url=jdbc:h2:tcp://localhost:9092/~ /test
spring.datasource.driverClassName=org.h2.Driver
spring.datasource.username=sa
spring.datasource.password=user
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.properties.hibernate.format_sql=true
```

Models

Question.java

```
java
CopyEdit
package com.telusko.Quiz_App.model;
```

```

import jakarta.persistence.*;
import lombok.Data;
import lombok.Getter;
import lombok.Setter;

@Entity
@Table(name = "question")
@Data
@Getter
@Setter
public class Question {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer id;
    private String category;

    @Column(name = "difficulty_level")
    private String difficultyLevel;
    private String option1;
    private String option2;
    private String option3;
    private String option4;

    @Column(name = "question_title")
    private String questionTitle;

    @Column(name = "right_answer")
    private String rightAnswer;
}

```

QuestionWrapper.java

```

java
CopyEdit
package com.telusko.Quiz_App.model;

```

```

import lombok.AllArgsConstructor;
import lombok.Data;

@Data
@AllArgsConstructor
public class QuestionWrapper {
    private Integer id;
    private String questionTitle;
    private String option1;
    private String option2;
    private String option3;
    private String option4;
}

```

Quiz.java

```

java
CopyEdit
package com.telusko.Quiz_App.model;

import jakarta.persistence.*;
import lombok.Data;

import java.util.List;

@Entity
@Data
public class Quiz {
    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer id;
    private String title;

    @ManyToMany

```

```
        private List<Question> questions;
    }
```

Response.java

```
java
CopyEdit
package com.telusko.Quiz_App.model;

import lombok.Data;
import lombok.RequiredArgsConstructor;

@Data
@RequiredArgsConstructor
public class Response {
    private Integer id;
    private String response;
}
```

Controllers

QuestionController.java

```
java
CopyEdit
package com.telusko.Quiz_App.controller;

import com.telusko.Quiz_App.model.Question;
import com.telusko.Quiz_App.service.QuestionService;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.*;
```

```

import java.util.List;

@RestController
@RequestMapping("/questions")
public class QuestionController {
    @Autowired
    private QuestionService questionService;

    @GetMapping("/GetAllQuestions")
    public ResponseEntity<List<Question>> getAll(){
        return questionService.getAllQuestions();
    }

    @GetMapping("/category/{category}")
    public ResponseEntity<List<Question>> getByCategory(@PathVariable String category){
        return questionService.findByCategory(category);
    }

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

    @PutMapping("/update/{id}")
    public ResponseEntity<Question> updateQuestion(@RequestBody Question updatedQuestion, @PathVariable Integer id){
        return questionService.updateQuestion(id, updatedQuestion);
    }

    @DeleteMapping("delete/{id}")
    public String deleteQuestion(@PathVariable Integer id){
        questionService.deleteQuestion(id);
        return "success";
    }
}

```

```
}  
}
```

QuizController.java

```
java  
CopyEdit  
package com.telusko.Quiz_App.controller;  
  
import com.telusko.Quiz_App.model.QuestionWrapper;  
import com.telusko.Quiz_App.model.Response;  
import com.telusko.Quiz_App.service.QuizService;  
import org.springframework.beans.factory.annotation.Autowired;  
import org.springframework.http.ResponseEntity;  
import org.springframework.web.bind.annotation.*;  
  
import java.util.List;  
  
@RestController  
@RequestMapping("/quiz")  
public class QuizController {  
    @Autowired  
    private QuizService quizService;  
  
    @PostMapping("/create")  
    public ResponseEntity<String> createQuiz(@RequestParam String category, @RequestParam int numQ, @RequestParam String title){  
        return quizService.createQuiz(category, numQ, title);  
    }  
  
    @GetMapping("/get/{id}")  
    public ResponseEntity<List<QuestionWrapper>> getQuizQuestions(@PathVariable Integer id){
```

```

        return quizService.getQuestionByQuizId(id);
    }

    @PostMapping("submit/{id}")
    public ResponseEntity<Integer> submitQuiz(@PathVariable Integer id, @RequestBody List<Response> responses){
        return quizService.calculateResult(id, responses);
    }
}

```

Services

QuizService.java

```

java
CopyEdit
package com.telusko.Quiz_App.service;

import com.telusko.Quiz_App.Repository.QuestionRepository;
import com.telusko.Quiz_App.Repository.QuizRepository;
import com.telusko.Quiz_App.model.Question;
import com.telusko.Quiz_App.model.QuestionWrapper;
import com.telusko.Quiz_App.model.Quiz;
import com.telusko.Quiz_App.model.Response;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Service;

import java.util.ArrayList;
import java.util.List;
import java.util.Optional;

@Service

```

```

public class QuizService {
    @Autowired
    private QuizRepository quizRepository;

    @Autowired
    private QuestionRepository questionRepository;

    public ResponseEntity<String> createQuiz(String category,
int numQ, String title) {
        List<Question> questions = questionRepository.getRand
omQuestions(category, numQ);
        Quiz quiz = new Quiz();
        quiz.setTitle(title);
        quiz.setQuestions(questions);
        quizRepository.save(quiz);
        return new ResponseEntity<>("success", HttpStatus.CRE
ATED);
    }

    public ResponseEntity<List<QuestionWrapper>> getQuestionB
yQuizId(Integer id) {
        Optional<Quiz> quiz = quizRepository.findById(id);
        List<Question> questionsfromDB = quiz.get().getQuesti
ons();
        List<QuestionWrapper> questionForUser = new ArrayList
<>();
        for(Question q: questionsfromDB){
            QuestionWrapper qw = new QuestionWrapper(q.getId
(), q.getQuestionTitle(), q.getOption1(), q.getOption2(), q.g
etOption3(), q.getOption4());
            questionForUser.add(qw);
        }
        return new ResponseEntity<>(questionForUser, HttpStat
us.OK);
    }
}

```



```

    public ResponseEntity<Integer> calculateResult(Integer id, List<Response> responses) {
        Quiz quiz = quizRepository.findById(id).get();
        List<Question> questions = quiz.getQuestions();
        int right = 0;
        int i = 0;
        for(Response response: responses){
            if(response.getResponse().equals(questions.get(i).getRightAnswer())){
                right++;
            }
            i++;
        }
        return new ResponseEntity<>(right, HttpStatus.OK);
    }
}

```

QuestionService.java

```

java
CopyEdit
package com.telusko.Quiz_App.service;

import com.telusko.Quiz_App.Repository.QuestionRepository;
import com.telusko.Quiz_App.model.Question;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.stereotype.Service;

import java.util.ArrayList;
import java.util.List;

@Service

```

```

public class QuestionService {
    @Autowired
    private QuestionRepository questionRepository;

    public ResponseEntity<List<Question>> getAllQuestions() {
        return new ResponseEntity<>(questionRepository.findAll(), HttpStatus.OK);
    }

    public ResponseEntity<List<Question>> findByCategory(String category) {
        return new ResponseEntity<>(questionRepository.findByCategory(category), HttpStatus.FOUND);
    }

    public ResponseEntity<String> addQuestion(Question question) {
        questionRepository.save(question);
        return new ResponseEntity<>("success", HttpStatus.CREATED);
    }

    public ResponseEntity<Question> updateQuestion(Integer id, Question updatedQuestion) {
        return questionRepository.findById(id)
            .map(question -> {
                question.setCategory(updatedQuestion.getCategory());
                question.setDifficultyLevel(updatedQuestion.getDifficultyLevel());
                question.setOption1(updatedQuestion.getOption1());
                question.setOption2(updatedQuestion.getOption2());
                question.setOption3(updatedQuestion.getOption3());
            });
    }
}

```

```

        question.setOption4(updatedQuestion.getOption4());
        question.setQuestionTitle(updatedQuestion.getQuestionTitle());
        question.setRightAnswer(updatedQuestion.getRightAnswer());
        return ResponseEntity.ok(questionRepository.save(question));
    }
    .orElse(ResponseEntity.status(HttpStatus.NOT_FOUND).body(null));
}

    public void deleteQuestion(Integer id) {
        questionRepository.deleteById(id);
    }
}

```

Repositories

QuizRepository.java

```

java
CopyEdit
package com.telusko.Quiz_App.Repository;

import com.telusko.Quiz_App.model.Quiz;
import org.springframework.data.jpa.repository.JpaRepository;

public interface QuizRepository extends JpaRepository<Quiz, Integer> {
}

```

QuestionRepository.java

```

java
CopyEdit
package com.telusko.Quiz_App.Repository;

import com.telusko.Quiz_App.model.Question;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;

import java.util.List;

public interface QuestionRepository extends JpaRepository<Question, Integer> {
    List<Question> findByCategory(String category);

    @Query(value = "SELECT * FROM question q WHERE q.category =:category ORDER BY RANDOM() LIMIT :numQ", nativeQuery = true)
    List<Question> getRandomQuestions(String category, int numQ);
}

```