Source Code Of OnLineQuizPortal Rest Api

Applicationn Properties

```
server.port 8082
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3306/spring
spring.datasource.username=root

spring.datasource.password=root1
```

OnlineQuizPortalApplication.java

```
package com;


import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.boot.autoconfigure.domain.EntityScan;


@SpringBootApplication(scanBasePackages="com")

@EntityScan(basePackages="com.bean")

public class OnlineQuizPortalApplication {


        public static void main(String[] args) {

                SpringApplication.run(OnlineQuizPortalApplication.class, args);

                System.out.println("Server up...");

        }


}
```

bean

Answer key .java

```
package com.bean;
```

```
import jakarta.persistence.Entity;
import jakarta.persistence.Id;

@Entity
public class Answerkey {
        @Id
        private int SNo;
        private String answer;
        public int getSNo() {
                return SNo;
        }
        public void setSNo(int sNo) {
                SNo = sNo;
        }
        public String getAnswer() {
                return answer;
        }
        public void setAnswer(String answer) {
                this.answer = answer;
        }
        @Override
        public String toString() {
                return "Answerkey [SNo=" + SNo + ", answer=" + answer + "]";
        }


}
```

Participant.java

```
package com.bean;

import jakarta.persistence.Entity;
import jakarta.persistence.Id;

@Entity
public class Participant {
	@Id
	private int pid;
	private String fname;
	private String lname;
	private int age;
	private String email;
	private String password;
	public int getPid() {
		return pid;
	}
	public void setPid(int pid) {
		this.pid = pid;
	}
	public String getFname() {
		return fname;
	}
	public void setFname(String fname) {
		this.fname = fname;
	}
	public String getLname() {
		return lname;
```

```
}
public void setLname(String lname) {
        this.lname = lname;
}
public int getAge() {
        return age;
}
public void setAge(int age) {
        this.age = age;
}
public String getemail() {
        return email;
}
public void setemail(String email) {
        this.email = email;
}
public String getPassword() {
        return password;
}
public void setPassword(String password) {
        this.password = password;
}
public Participant(int pid, String fname, String lname, int age, String email, String password) {
        super();
        this.pid = pid;
        this.fname = fname;
        this.lname = lname;
        this.age = age;
        this.email = email;
```

```
		this.password = password;

	}

	public Participant() {

		super();

		// TODO Auto-generated constructor stub

	}

	@Override

	public String toString() {

		return "Participant [pid=" + pid + ", fname=" + fname + ", lname=" + lname + ", age=" +
age + ", email="

				+ email + ", password=" + password + "]";

	}



}
```

Qusetions.java

```
package com.bean;


import jakarta.persistence.Entity;

import jakarta.persistence.Id;




@Entity

public class Questions {

	@Id

	private int id;

	private String question;
private String option1;
private String option2;
private String option3;
private String option4;
public int getid() {
        return id;
}
public void setid(int id) {
        id = id;
}
public String getQuestion() {
        return question;
}
public void setQuestion(String question) {
        this.question = question;
}
public String getOption1() {
        return option1;
}
public void setOption1(String option1) {
        this.option1 = option1;
}
public String getOption2() {
        return option2;
}
public void setOption2(String option2) {
        this.option2 = option2;
}
public String getOption3() {
        return option3;

    }

    public void setOption3(String option3) {

        this.option3 = option3;

    }

    public String getOption4() {

        return option4;

    }

    public void setOption4(String option4) {

        this.option4 = option4;

    }

    @Override

    public String toString() {

        return "Questions [id=" + id + ", question=" + question + ", option1=" + option1 + ",
option2=" + option2

                + ", option3=" + option3 + ", option4=" + option4 + "]";

    }





}
```

Quiz.java

```
package com.bean;


import jakarta.persistence.Entity;

import jakarta.persistence.Id;
```

```
@Entity
public class Quiz {
	@Id
	private int qid;
	private String category;
	private String topic;
	private String description;
	public int getQid() {
		return qid;
	}
	public void setQid(int qid) {
		this.qid = qid;
	}
	public String getCategory() {
		return category;
	}
	public void setCatogory(String catogory) {
		this.category = catogory;
	}
	public String getTopic() {
		return topic;
	}
	public void setTopic(String topic) {
		this.topic = topic;
	}
	public String getDescription() {
		return description;
	}
	public void setDescription(String description) {
		this.description = description;
	}
	@Override
	public String toString() {
		return "Quiz [qid=" + qid + ", category=" + category + ", topic=" + topic + ", description="
+ description
				+ "]";
	}




}
```

Responses.java

```
package com.bean;


import jakarta.persistence.Entity;
import jakarta.persistence.Id;


@Entity
public class Responses {
	@Id
	private int id;
	private String response;
	public int getId() {
		return id;
	}
	public void setId(int id) {
		this.id = id;
	}
	public String getResponse() {
		return response;
	}
	public void setResponse(String response) {
		this.response = response;
	}
	@Override
	public String toString() {
		return "Responses [id=" + id + ", response=" + response + "]";
	}


}
```

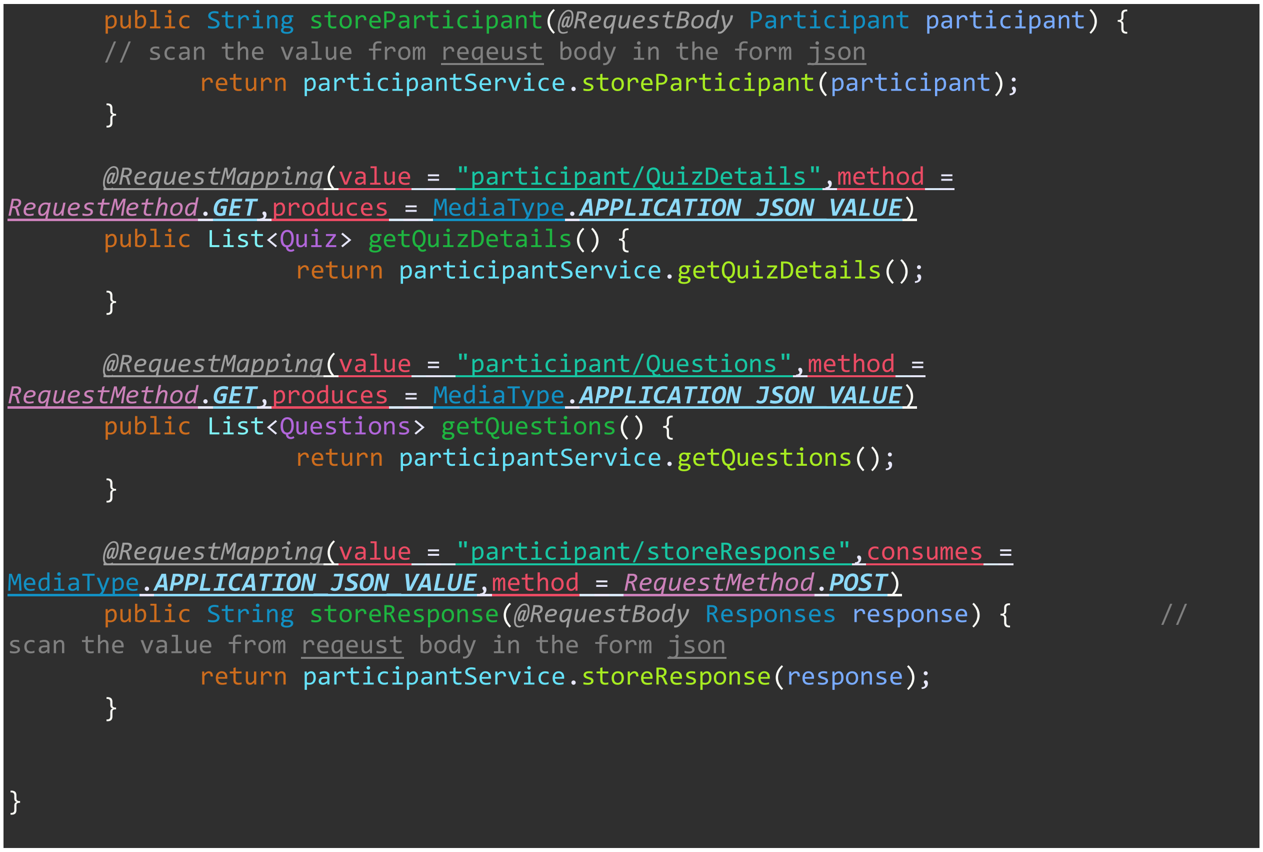
Controller

AdminController.java

```
package com.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.MediaType;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RestController;

import com.bean.Answerkey;
import com.bean.Questions;
import com.bean.Quiz;
import com.service.AdminService;

@RestController
public class AdminController {
	@Autowired
	AdminService adminService;
	@RequestMapping(value = "admin/storeQuiz",consumes =
MediaType.APPLICATION_JSON_VALUE,method = RequestMethod.POST)
	public String storeParticipant(@RequestBody Quiz quiz) {          // scan the
value from reqeust body in the form json
		return adminService.storeQuiz(quiz);
	}

	@RequestMapping(value = "admin/storeQuestions",consumes =
MediaType.APPLICATION_JSON_VALUE,method = RequestMethod.POST)
	public String storeQuestions(@RequestBody Questions questions) {         //
scan the value from reqeust body in the form json
		return adminService.storeQuestions(questions);
	}

	@RequestMapping(value = "admin/storeAnswer",consumes =
MediaType.APPLICATION_JSON_VALUE,method = RequestMethod.POST)
	public String storeParticipant(@RequestBody Answerkey ans) {             //
scan the value from reqeust body in the form json
		return adminService.storeAnswerkey(ans);
	}


}
```

Participant Controller.java

```
package com.controller;

import java.util.List;

import jakarta.servlet.http.HttpServletRequest;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.MediaType;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RequestMethod;
import org.springframework.web.bind.annotation.RestController;

import com.bean.Participant;
import com.bean.Questions;
import com.bean.Quiz;
import com.bean.Responses;
import com.service.ParticipantService;

@RestController
public class ParticipantController {

	@Autowired
	ParticipantService participantService;

	@RequestMapping(value = "participant/storeParticipant",consumes =
MediaType.APPLICATION_JSON_VALUE,method = RequestMethod.POST)
```



```
	public String storeParticipant(@RequestBody Participant participant) {
	// scan the value from reqeust body in the form json
		return participantService.storeParticipant(participant);
	}

	@RequestMapping(value = "participant/QuizDetails",method =
RequestMethod.GET,produces = MediaType.APPLICATION_JSON_VALUE)
	public List<Quiz> getQuizDetails() {
			return participantService.getQuizDetails();
	}

	@RequestMapping(value = "participant/Questions",method =
RequestMethod.GET,produces = MediaType.APPLICATION_JSON_VALUE)
	public List<Questions> getQuestions() {
			return participantService.getQuestions();
	}

	@RequestMapping(value = "participant/storeResponse",consumes =
MediaType.APPLICATION_JSON_VALUE,method = RequestMethod.POST)
	public String storeResponse(@RequestBody Responses response) {		//
scan the value from reqeust body in the form json
		return participantService.storeResponse(response);
	}


}
```

DAO

AdminDao

package com.dao;

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

import org.springframework.stereotype.Repository;

import com.bean.Answerkey;

import com.bean.Questions;

import com.bean.Quiz;

import jakarta.persistence.EntityManager;

import jakarta.persistence.EntityManagerFactory;

```
import jakarta.persistence.EntityTransaction;


@Repository
public class AdminDAO {
    @Autowired
    EntityManagerFactory emf;


    public int storeQuiz(Quiz quiz) {
        try {
            EntityManager manager = emf.createEntityManager();
            EntityTransaction tran = manager.getTransaction();
            tran.begin();
                manager.persist(quiz);
    // session.save(emp)
            tran.commit();
            return 1;
        } catch (Exception e) {
        System.out.println(e);
        return 0;
        }
    }


    public int storeQuestions(Questions questions) {
        try {
            EntityManager manager = emf.createEntityManager();
            EntityTransaction tran = manager.getTransaction();
            tran.begin();
                manager.persist(questions);
    // session.save(emp)
```

```
            tran.commit();

            return 1;

        } catch (Exception e) {

        System.out.println(e);

        return 0;

        }

    }


    public int storeAnswerkey(Answerkey ans) {

        try {

            EntityManager manager = emf.createEntityManager();

            EntityTransaction tran = manager.getTransaction();

            tran.begin();

                manager.persist(ans);
    // session.save(emp)

            tran.commit();

            return 1;

        } catch (Exception e) {

        System.out.println(e);

        return 0;

        }

    }

}
```

ParticipantDao

```
package com.dao;


import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Repository;

import com.bean.Participant;
import com.bean.Questions;
import com.bean.Quiz;
import com.bean.Responses;

import jakarta.persistence.EntityManager;
import jakarta.persistence.EntityManagerFactory;
import jakarta.persistence.EntityTransaction;
import jakarta.persistence.Query;

@Repository
public class ParticipantDAO {

    @Autowired
    EntityManagerFactory emf;

    public int storeParticipant(Participant participant) {
        try {
            EntityManager manager = emf.createEntityManager();
            EntityTransaction tran = manager.getTransaction();
            tran.begin();
                manager.persist(participant);
    // session.save(emp)
            tran.commit();
            return 1;
        } catch (Exception e) {
```

```
    System.out.println(e);

    return 0;

    }

}


public List<Quiz> getQuizDetails() {

    EntityManager manger = emf.createEntityManager();

    Query qry = manger.createQuery("select q from Quiz q");                    // JPQL

    List<Quiz> listOfQuiz = qry.getResultList();

    return listOfQuiz;

}


public List<Questions> getQuestions() {

    EntityManager manger = emf.createEntityManager();

    Query qry = manger.createQuery("select q from Questions q");
// JPQL

    List<Questions> listOfQuestions = qry.getResultList();

    return listOfQuestions;

}


public int storeResponse(Responses response) {

    try {

        EntityManager manager = emf.createEntityManager();

        EntityTransaction tran = manager.getTransaction();

        tran.begin();

            manager.persist(response);
// session.save(emp)

        tran.commit();

        return 1;
        } catch (Exception e) {
        System.out.println(e);
        return 0;
        }
    }
}
```

AdminServices.java

```
package com.service;

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

import com.bean.Answerkey;
import com.bean.Questions;
import com.bean.Quiz;
import com.dao.AdminDAO;


@Service
public class AdminService {
    @Autowired
    AdminDAO AdminDao;

    public String storeQuiz(Quiz quiz) {
        if(AdminDao.storeQuiz(quiz)>0) {
            return "Quiz details stored";
        }else {
            return "Quiz details didn't store";
        }
    }

    public String storeQuestions(Questions questions) {
        if(AdminDao.storeQuestions(questions)>0) {
            return "Questions stored";
        }else {
            return "Questions didn't store";
        }
    }

    public String storeAnswerkey(Answerkey ans) {
        if(AdminDao.storeAnswerkey(ans)>0) {
            return "Answer stored";
        }else {
            return "Answer didn't store";
        }
    }
}
```

ParticipantServices.java

```
package com.service;

import java.util.List;

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

import com.bean.Participant;
import com.bean.Questions;
import com.bean.Quiz;
import com.bean.Responses;
import com.dao.ParticipantDAO;

@Service
public class ParticipantService {
    @Autowired
    ParticipantDAO participantDao;

    public String storeParticipant(Participant participant) {
        if(participantDao.storeParticipant(participant)>0) {
            return "Participant details stored";
        }else {
            return "Participant details didn't store";
        }
    }

    public List<Quiz> getQuizDetails() {
        return participantDao.getQuizDetails();
    }

    public List<Questions> getQuestions() {
        return participantDao.getQuestions();
    }

    public String storeResponse(Responses response) {
        if(participantDao.storeResponse(response)>0) {
            return "Response stored";
```

```
                }else {
                        return "Response didn't store";
                }
        }
}
```

ONLINE Quiz Application Test

```
package com;

import org.junit.jupiter.api.Test;
import org.springframework.boot.test.context.SpringBootTest;

@SpringBootTest
class OnlineQuizPortalApplicationTests {

        @Test
        void contextLoads() {
        }

}
```

POM.XML

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
	xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
https://maven.apache.org/xsd/maven-4.0.0.xsd">
	<modelVersion>4.0.0</modelVersion>
	<parent>
		<groupId>org.springframework.boot</groupId>
		<artifactId>spring-boot-starter-parent</artifactId>
		<version>3.1.0</version>
		<relativePath/> <!-- lookup parent from repository -->
	</parent>
	<groupId>com.example</groupId>
	<artifactId>OnlineQuizPortal</artifactId>
```

```xml
    <version>0.0.1-SNAPSHOT</version>
    <name>OnlineQuizPortal</name>
    <description>Demo project for Spring Boot</description>
    <properties>
        <java.version>11</java.version>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-data-jpa</artifactId>
        </dependency>
        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-web</artifactId>
        </dependency>

        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-devtools</artifactId>
            <scope>runtime</scope>
            <optional>true</optional>
        </dependency>

        <dependency>
             <groupId>com.mysql</groupId>
             <artifactId>mysql-connector-j</artifactId>
             <version>8.0.33</version>
        </dependency>

        <dependency>
            <groupId>org.springframework.boot</groupId>
            <artifactId>spring-boot-starter-test</artifactId>
            <scope>test</scope>
        </dependency>
    </dependencies>

    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
        </plugins>
    </build>

</project>
```

Step By Step Process

Admin User Scenario:

a. Admin creates a set of questions along with their answers.

b. Admin logs in by calling the adminLogin function with the admin username and password.

c. If the login is successful, an access token is generated and returned.

d. Admin updates their profile details by calling the updateAdminProfile function with the new profile details and the access token.

e. Admin changes their password by calling the changeAdminPassword function with the new password and the access token.

f. Admin adds questions by calling the addQuestions function with the new questions and the access token.

g. Admin creates a quiz by calling the createQuiz function with the quiz name, ID, selected question IDs, and the access token.

h. Admin can get the list of all quizzes by calling the getAllQuizzes function.

i. Admin can get the users who participated in a specific quiz along with their scores and standings by calling the getQuizUsers function with the quiz ID and the access token.

j. Admin can get statistics on total quizzes, questions, and users by calling the getQuizStatistics function.

1. **Participants Scenario:**
2. **a. Users register an account by calling the userRegister function with their account details, which returns an access token.**
3. **b. Users log in by calling the userLogin function with the access token.**
4. **c. Users can browse various quizzes created by the admin by calling the getQuizzes function.**
5. **d. Users attempt a quiz by calling the attemptQuiz function with the quiz ID, their answers, and the access token.**
6. **e. Users can check if their provided answers are correct by calling the getQuizResults function with the quiz ID.**
7. **f. Users can compare their standings with other users by calling the getQuizStandings function with the quiz ID.**