

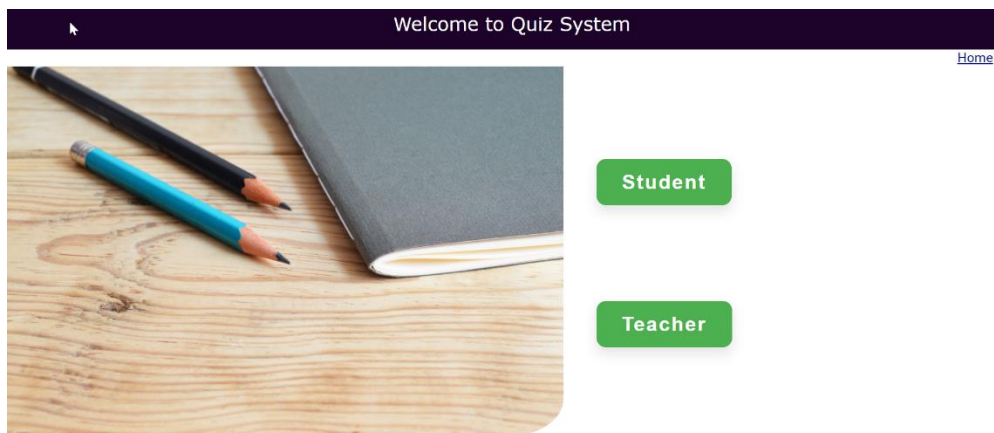
Advanced Java Report



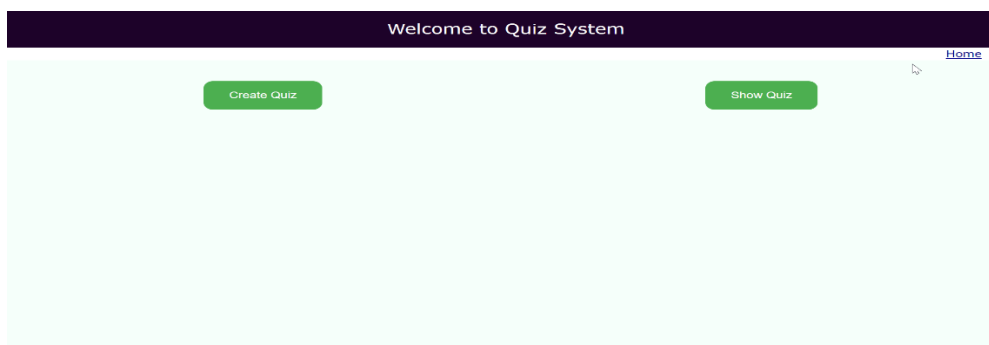
Submitted By: Naresh Kumar Jaiswal
Submitted To: Thomas Broussard Sir

Starting from Frontend,

User is welcome by the login page or Option page for Student or Teacher.



If a user chose to be Teacher, he will be redirected to another page.



He can choose create question/Quiz or show created quiz. Lets cho

Lets Choose Create quiz/question. Here Teacher can Add questions

A screenshot of a form titled "ENTER QUESTIONS". The form is dark-themed and includes the following elements: a "Select Quiz" dropdown menu with "SELECT" as the current value; input fields for "Question", "Topics", "Option 1", "Option 2", "Option 3", and "Option 4"; a "Select Correct Option" dropdown menu with "SELECT" as the current value; and two green buttons at the bottom labeled "Submit" and "Reset".

Welcome to Quiz System

[Home](#)

Select Quiz sap

Question 1.



what is sap?

1. CRM

2. ERP

3. Language

4. script



Question 2

Here Teacher can edit or delete the created question.

Now Student can start the quiz.

Welcome to Quiz System

[Home](#)

Student Name

Submit

Reset

Sample of Quiz Exam questions

Welcome to Quiz System

[Home](#)

Select Quiz sap

Question 1.

what is sap?

☐ 1. CRM

☐ 2. ERP

☐ 3. Language

☐ 4. script

After submitting student got the Result.

what is sap?

- ☐ 1. CRM
- ☒ 2. ERP
- ☐ 3. Language
- ☐ 4. script

Your answer is correct.

Question 2.

what is your name?

- ☒ 1. naresh
- ☐ 2. raj
- ☐ 3. rojiy
- ☐ 4. aa

Your answer is wrong.

Now Backend,

I have used JAVA , SPRING , JPA and H2 for database.

API, s used are:

1. Creating questions.

```
@PUT
@Path("/{id}/questions")
@Consumes(MediaType.APPLICATION_JSON)
public Response updateQuestionForQuiz(@PathParam("id") String id, QuestionDTO dto) {
    if (dto == null) {
        return Response.status(400, "no question was provided").entity("bad request").build();
    }

    try {
        service.UpdateMCQQuestion(dto);
    } catch (CreationFailedException cfe) {
        return Response.notModified().build();
    }

    return Response.created(URI.create(String.valueOf("quiz/"+id+"/question/"+ dto.getQuestionId()))).build()
}
```

2. Updating question.

```
@PUT
@Path("/{id}/questions")
@Consumes(MediaType.APPLICATION_JSON)
public Response updateQuestionForQuiz(@PathParam("id") String id, QuestionDTO dto) {
    if (dto == null) {
        return Response.status(400, "no question was provided").entity("bad request").build();
    }

    try {
        service.UpdateMCQQuestion(dto);
    } catch (CreationFailedException cfe) {
        return Response.notModified().build();
    }

    return Response.created(URI.create(String.valueOf("quiz/"+id+"/question/"+ dto.getQuestionId()))).build()
}
```

3. Deleting question

```
@DELETE
@Path("/question/{id}")
@Consumes(MediaType.APPLICATION_JSON)
public Response DeleteQuestionForQuiz(@PathParam("id") String id) {

    try {
        service.DeleteMCQQuestion(id);
    } catch (CreationFailedException cfe) {
        return Response.notModified().build();
    }
    return Response.ok("deleted").build();
}
```

4. Get Question List according to Quiz id

```
@GET
@Path("/AllQuestionsByQuizId/{quizId}")
@Produces(MediaType.APPLICATION_JSON)
public Response AllQuestionsByQuizId(@PathParam("quizId") String quizIdId) {

    List<QuestionDTO> dto = service.getAllByQuizId(quizIdId);
    return Response.ok().entity(dto).build();
}
```

5. Save Student Quiz answers

```
@POST
@Path("/{id}/answers")
@Consumes(MediaType.APPLICATION_JSON)
public Response AddStudentAnswers(@PathParam("id") String id, MCQAnswerDTO dto) {
    if (dto == null) {
        return Response.status(400, "no Answers was provided").entity("bad request").build();
    }

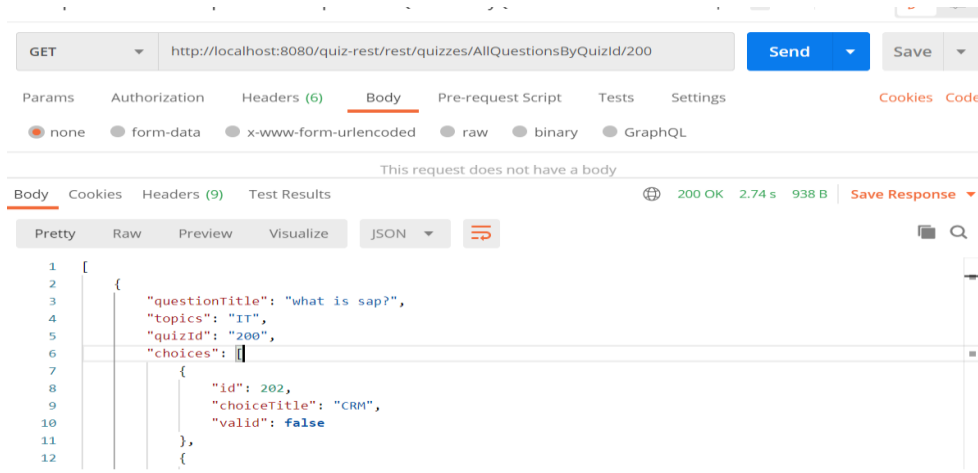
    try {
        service.createMCQAnswer(dto);
    } catch (CreationFailedException cfe) {}
    return Response.notModified().build();
}

return Response.created(URI.create(String.valueOf("quiz/"+id+"/Answer/"+ dto.getAnswerId()))).build();
}

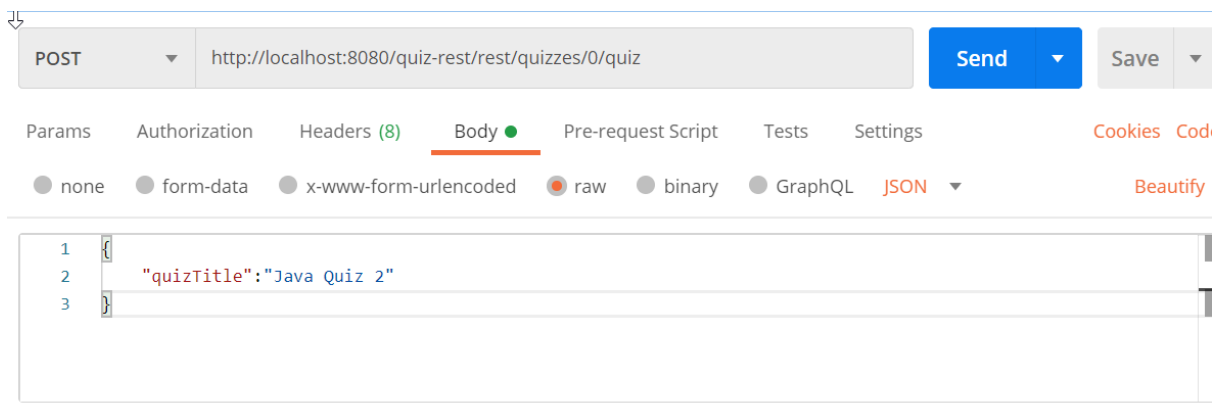
@GET
```

Now PostMan Api Request and Response.

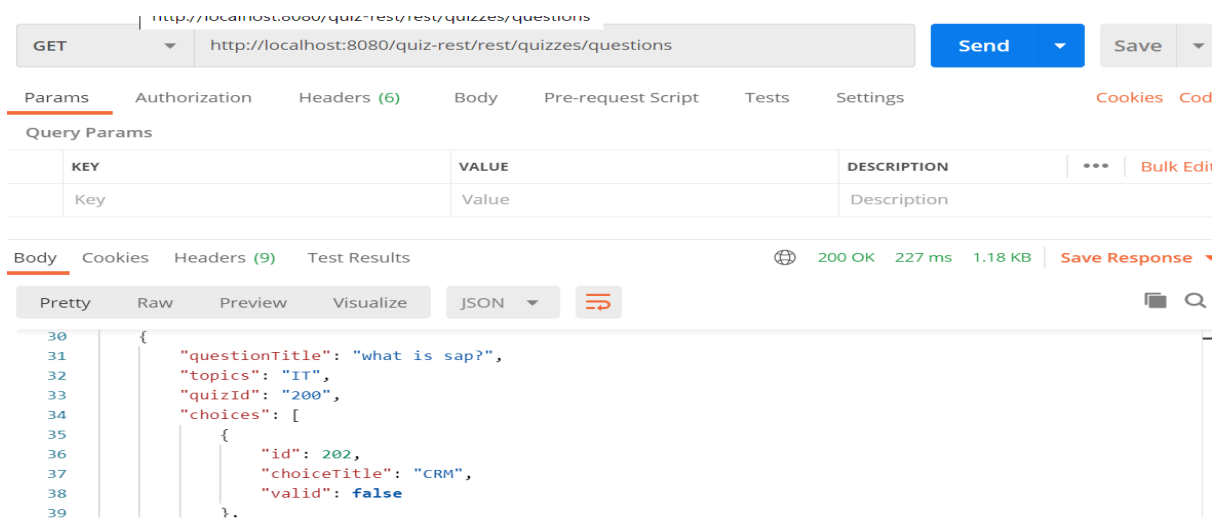
1. Get all Questions By QuizId



2. Create Quiz.



3. Get All Questions



4. Create Questions.

The screenshot shows the REST Client interface for a POST request. The method is **POST** and the URL is `http://localhost:8080/quiz-rest/rest/quizzes/0/questions`. The **Body** tab is selected, showing a JSON payload:

```
1 {
2   "questionTitle": "my new question 101",
3   "quizId": "1",
4   "topics": "abb",
5   "choices": [
6     {
7       "choiceTitle": "my choice1",
8       "valid": false
9     },
10    {
11      "choiceTitle": "my choice2",
12      "valid": false
13    }
14  ]
15 }
```

Below the request, the **Response** section is visible, featuring a rocket ship icon.

5. Delete Questions.

The screenshot shows the REST Client interface for a DELETE request. The method is **DELETE** and the URL is `http://localhost:8080/quiz-rest/rest/quizzes/question/200`. The **Body** tab is selected, and a message states: "This request does not have a body".

Below the request, the **Response** section is visible, showing a status of **200 OK** with a response time of **1101 ms** and a size of **372 B**. The response body is displayed in the **Text** view:

```
1 deleted
```

6. Update Questions.

The screenshot shows the REST Client interface for a PUT request. The method is **PUT** and the URL is `http://localhost:8080/quiz-rest/rest/quizzes/15/questions`. The **Body** tab is selected, showing a JSON payload:

```
1 {
2   "questionId": 103,
3   "questionTitle": "my new question 999",
4   "choices": [
5     {
6       "id": 105,
7       "choiceTitle": "my choice18"
8     }
9   ]
10 }
11
```

Below the request, the **Response** section is visible, featuring a rocket ship icon.

7. Get Question By ID

GET

http://localhost:8080/quiz-rest/rest/quizzes/questions/201

Send

Save

ParamsAuthorizationHeaders (6)BodyPre-request ScriptTestsSettingsCookiesCode

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
-----	-------	-------------	-----	-----------

BodyCookiesHeaders (9)Test Results200 OK198 ms650 BSave Response

PrettyRawPreviewVisualizeJSON

```
1 {
2   "questionTitle": "what is sap?",
3   "topics": "IT",
4   "quizId": "200",
5   "choices": [
6     {
7       "id": 202,
8       "choiceTitle": "CRM",
9       "valid": false
10    },
11    {
12      "id": 203,
```

8. Add Student

POST

http://localhost:8080/quiz-rest/rest/students/0/student

Send

Save

ParamsAuthorizationHeaders (8)BodyPre-request ScriptTestsSettingsCookiesCode

noneform-datax-www-form-urlencodedrawbinaryGraphQLJSONBeautify

1 {
2 "studentName": "Naresh kumar"
3 }

Response



9. Get Student.

↓

GET http://localhost:8080/quiz-rest/rest/students/students/23 Send Save

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies Code

Query Params

KEY	VALUE	DESCRIPTION	...	Bulk Edit
Key	Value	Description		

Body Cookies Headers (9) Test Results 200 OK 600 ms 1.65 KB Save Response

Pretty Raw Preview Visualize JSON 🔍

```
1 {  
2   "id": 23,  
3   "studentName": "Naresh"  
4 },  
5 {  
6   "id": 24,  
7   "studentName": "Naresh"8 },  
9 ]
```

10. Save Student Exam Answers.

POST http://localhost:8080/quiz-rest/rest/studentExam/0/answers Send Save

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies Code

☒ none ☐ form-data ☐ x-www-form-urlencoded ☒ raw ☐ binary ☐ GraphQL JSON Beautify

```
1 {  
2   "questionId": "2",  
3   "studenId": "101",  
4   "answer": "1",  
5   "quizId": 1  
6 }
```

11. Get AllQuiz

GET

http://localhost:8080/quiz-rest/rest/quizzes/quizAll

Send

Save

Params

Authorization

Headers (6)

Body

Pre-request Script

Tests

Settings

Cookies

Code

Query Params

	KEY	VALUE	DESCRIPTION	...	Bulk Edit
--	-----	-------	-------------	-----	-----------

Body

Cookies

Headers (9)

Test Results

200 OK

110 ms

611 B

Save Response

Pretty

Raw

Preview

Visualize

JSON

```
1 [
2   {
3     "id": 102,
4     "quizTitle": "aaa"
5   },
6   {
7     "id": 126,
8     "quizTitle": "abc"
9   },
10  {
11    "id": 127,
12    "quizTitle": "new q1"
```

12. Add Quiz.

POST

http://localhost:8080/quiz-rest/rest/quizzes/0/quiz

Send

Save

Params

Authorization

Headers (8)

Body

Pre-request Script

Tests

Settings

Cookies

Code

none

form-data

x-www-form-urlencoded

raw

binary

GraphQL

JSON

Beautify

```
1 {
2   "quizTitle": "Java Quiz 2"
3 }
```

THANK YOU