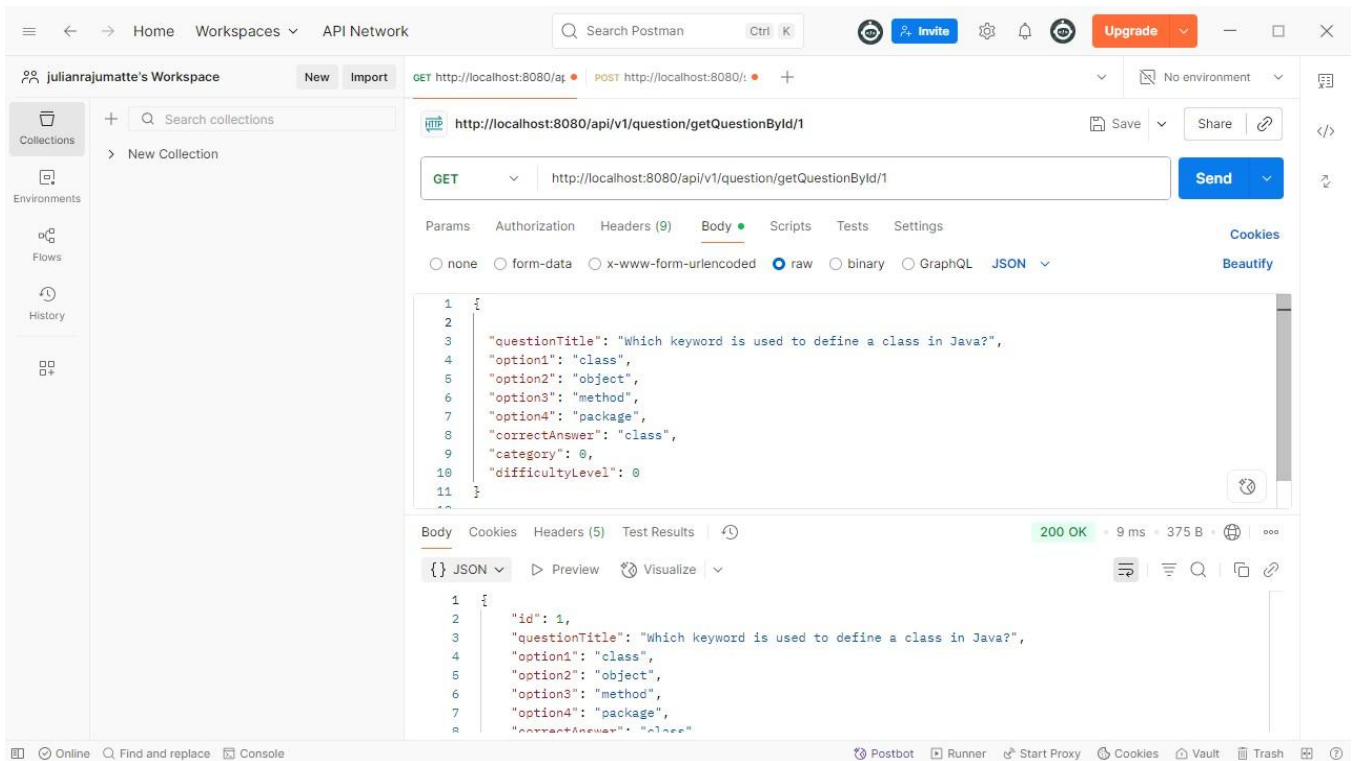


QuizApp-monorepo Testing Endpoints outputs :

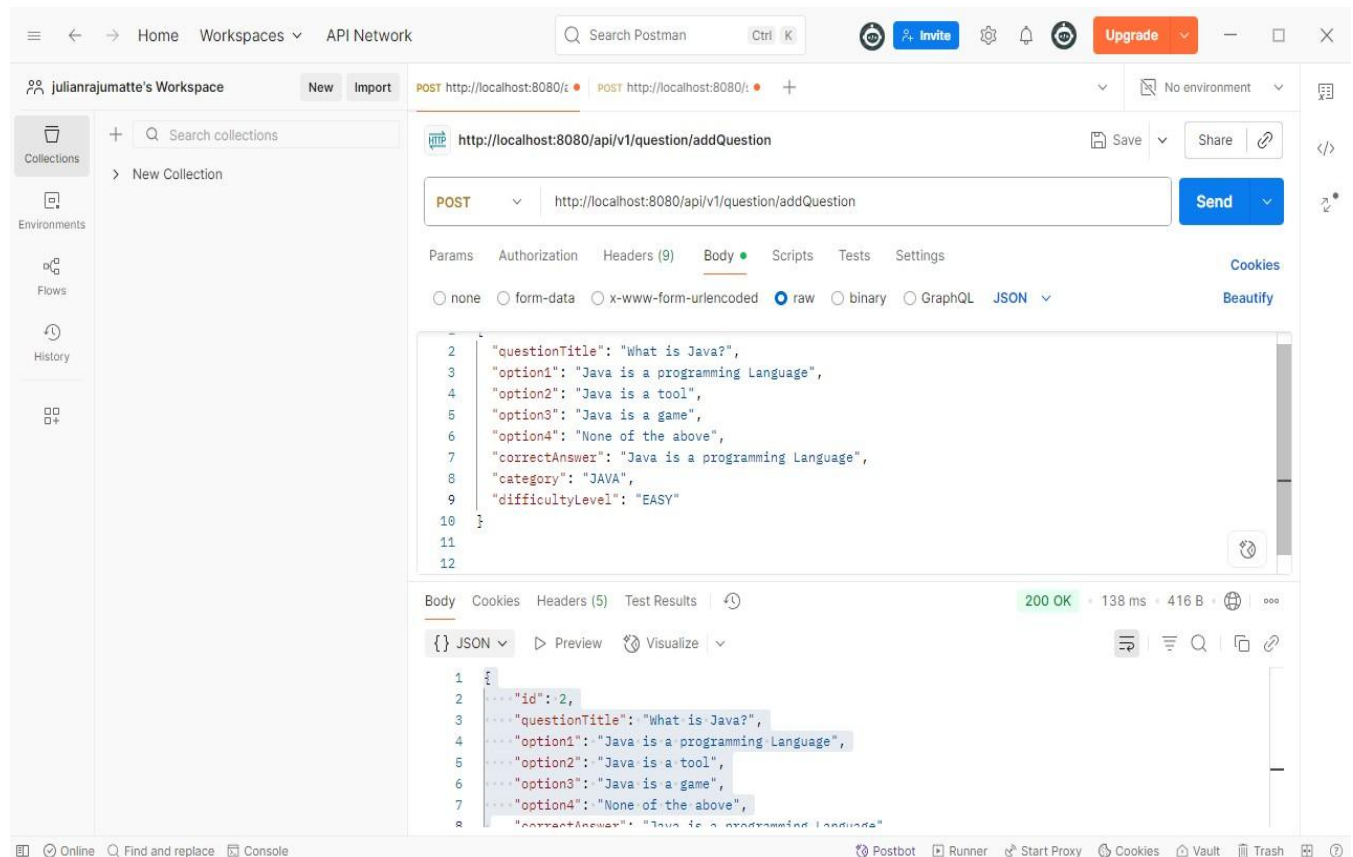
GET:



The screenshot shows the Postman interface for a GET request. The URL is `http://localhost:8080/api/v1/question/getQuestionById/1`. The response is a 200 OK status with a response time of 9 ms and a body size of 375 B. The response body is a JSON object:

```
1 {
2   "questionTitle": "Which keyword is used to define a class in Java?",
3   "option1": "class",
4   "option2": "object",
5   "option3": "method",
6   "option4": "package",
7   "correctAnswer": "class",
8   "category": 0,
9   "difficultyLevel": 0
10 }
```

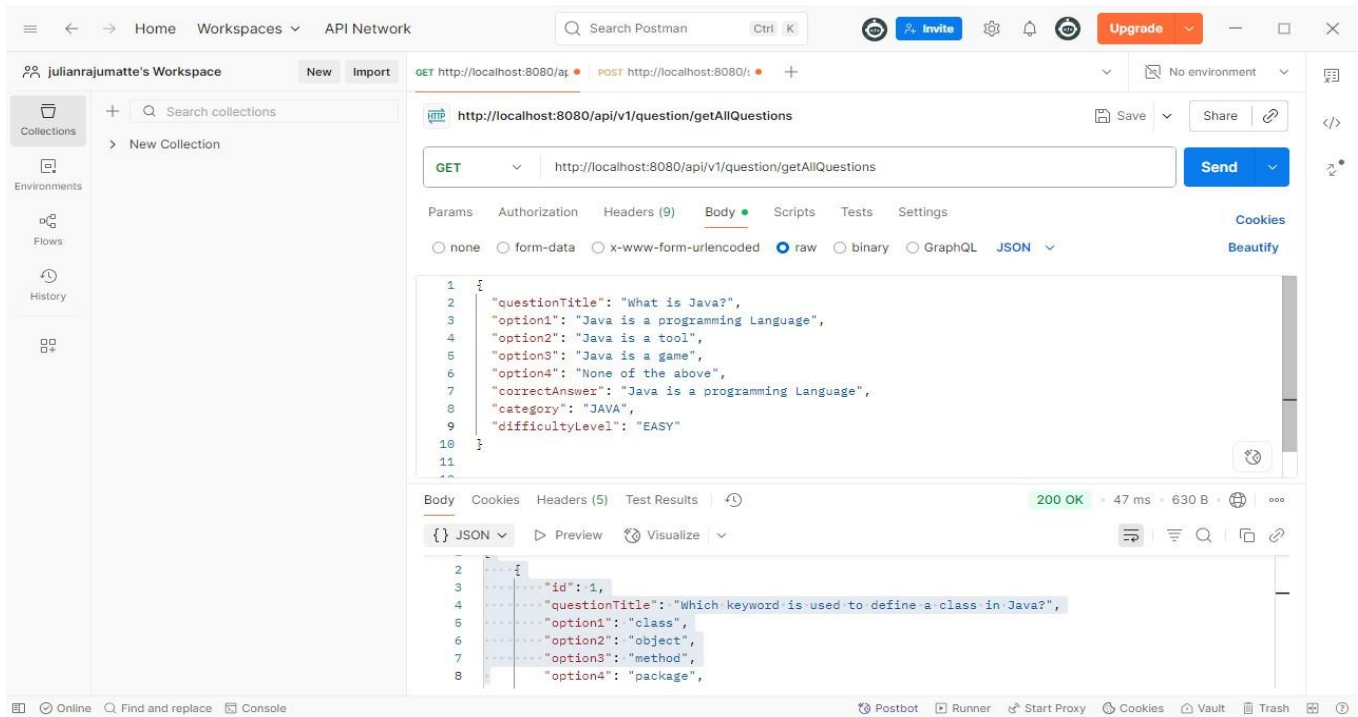
//Post



The screenshot shows the Postman interface for a POST request. The URL is `http://localhost:8080/api/v1/question/addQuestion`. The response is a 200 OK status with a response time of 138 ms and a body size of 416 B. The response body is a JSON object:

```
1 {
2   "id": 2,
3   "questionTitle": "What is Java?",
4   "option1": "Java is a programming Language",
5   "option2": "Java is a tool",
6   "option3": "Java is a game",
7   "option4": "None of the above",
8   "correctAnswer": "Java is a programming Language",
9   "category": "JAVA",
10  "difficultyLevel": "EASY"
11 }
```

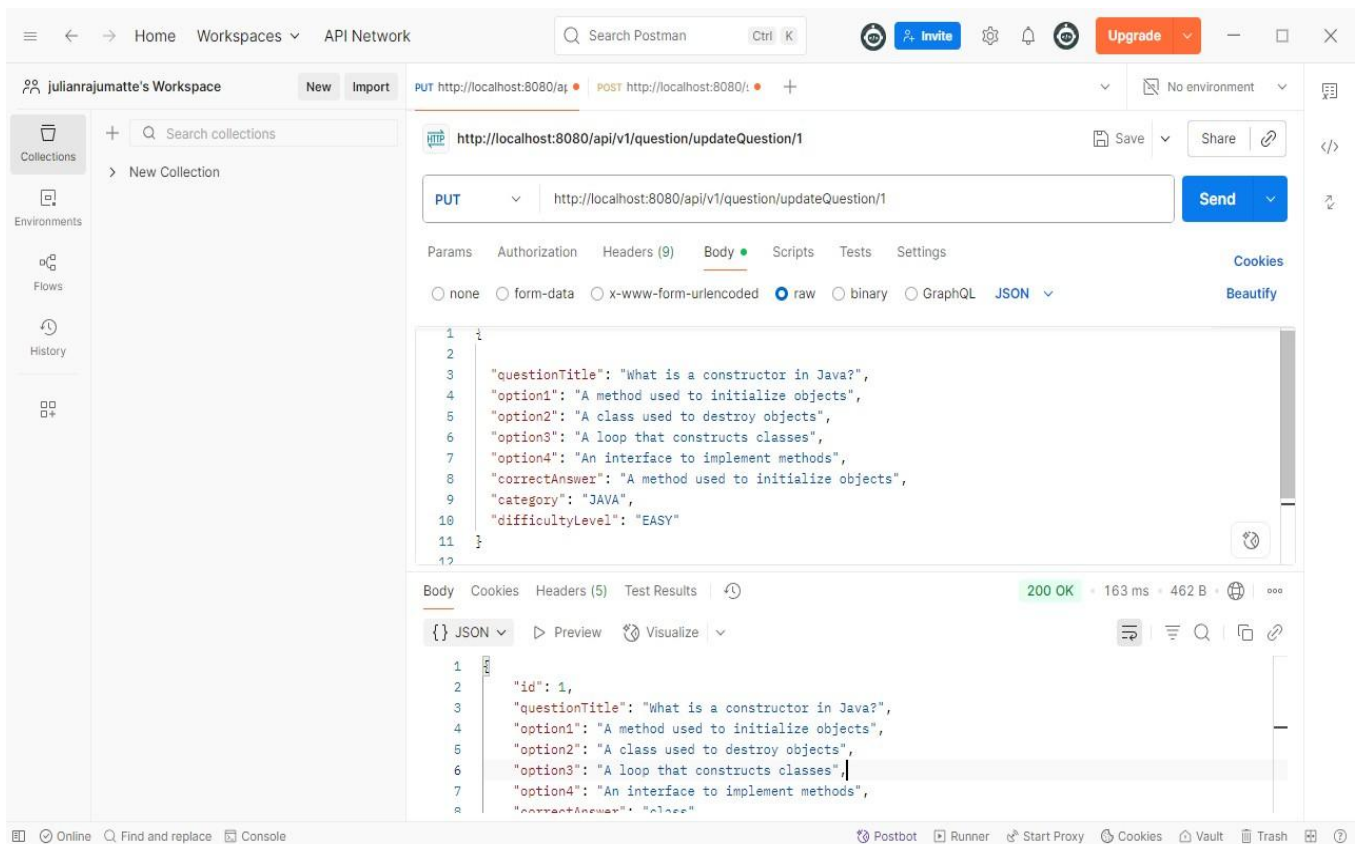
GetAllQuestions:



The screenshot shows a Postman interface with a GET request to `http://localhost:8080/api/v1/question/getAllQuestions`. The response is a JSON array of question objects. The first object in the array is:

```
{
  "questionTitle": "What is Java?",
  "option1": "Java is a programming Language",
  "option2": "Java is a tool",
  "option3": "Java is a game",
  "option4": "None of the above",
  "correctAnswer": "Java is a programming Language",
  "category": "JAVA",
  "difficultyLevel": "EASY"
}
```

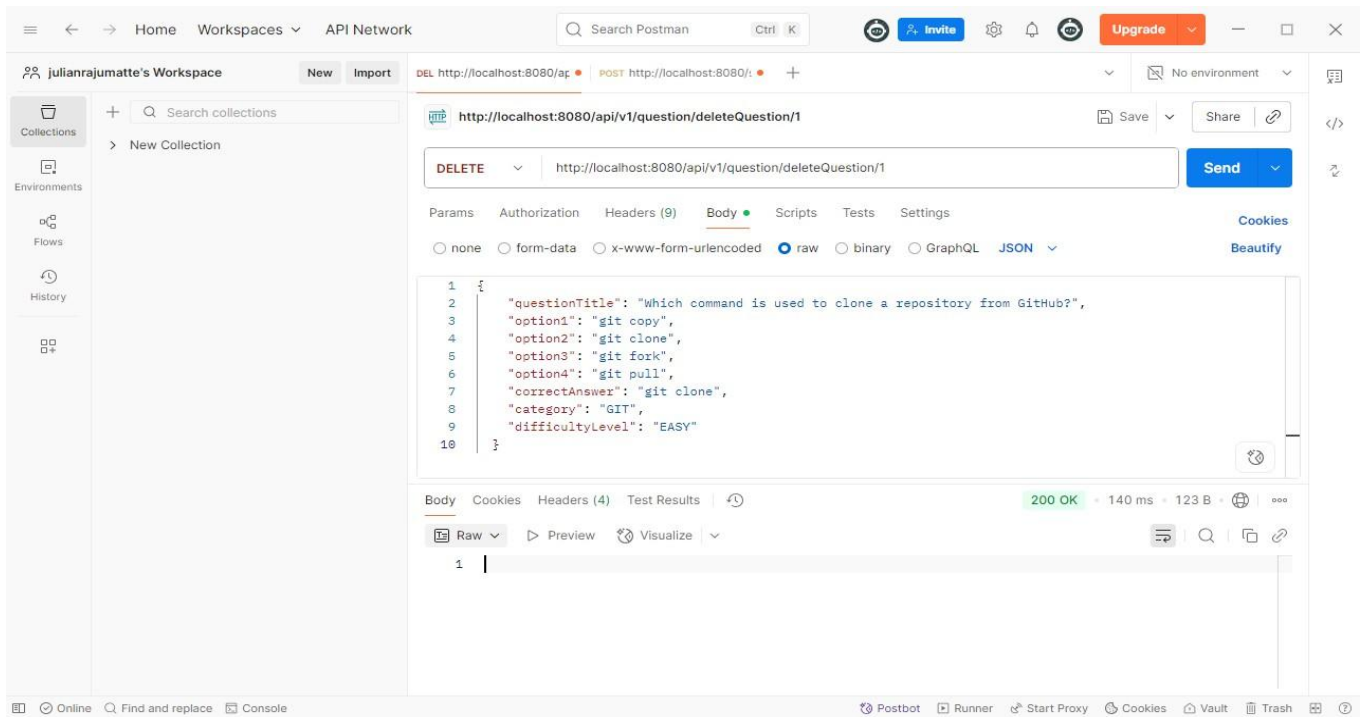
//UpdateQuestion:



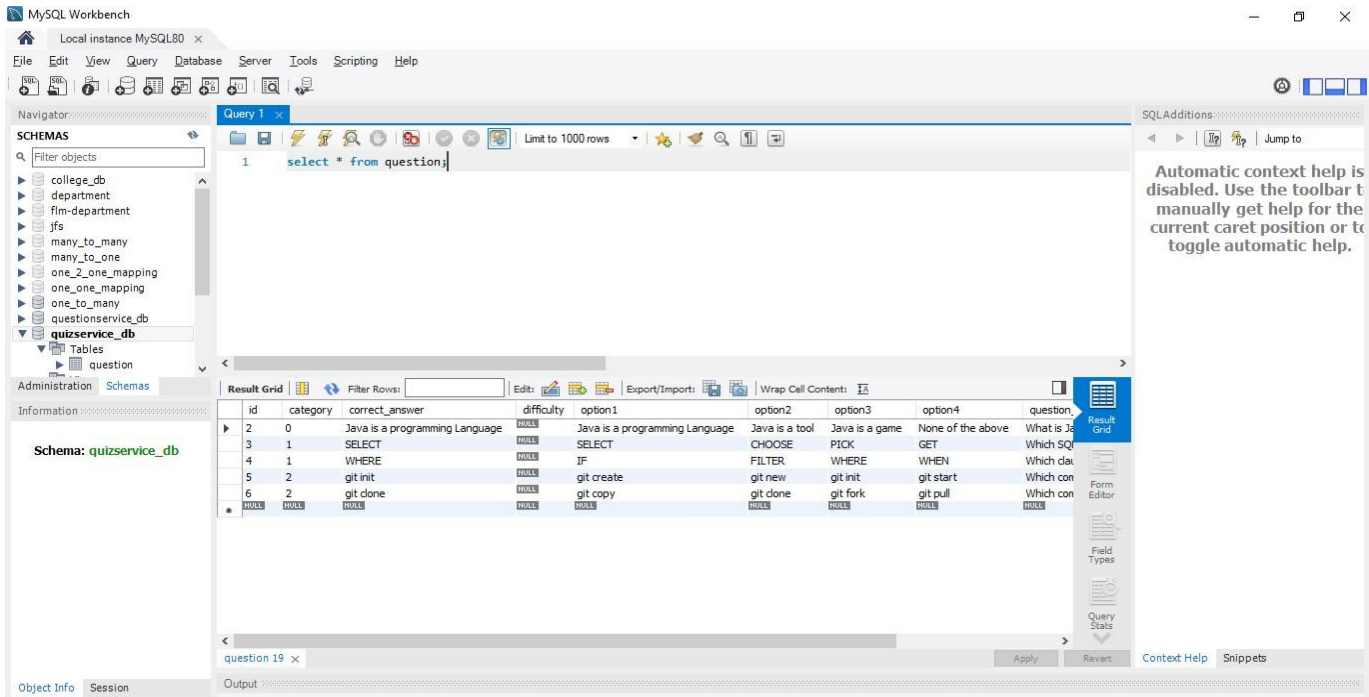
The screenshot shows a Postman interface with a PUT request to `http://localhost:8080/api/v1/question/updateQuestion/1`. The response is a JSON object representing the updated question:

```
{
  "id": 1,
  "questionTitle": "What is a constructor in Java?",
  "option1": "A method used to initialize objects",
  "option2": "A class used to destroy objects",
  "option3": "A loop that constructs classes",
  "option4": "An interface to implement methods",
  "correctAnswer": "A method used to initialize objects",
  "category": "JAVA",
  "difficultyLevel": "EASY"
}
```

//Delete Question:



MySQL Database:



// **QuestionNotFoundException** (Custom Exception)

The screenshot shows a Postman interface with a PATCH request to `http://localhost:8080/api/v1/question/updateQuestionById/1`. The request body is a JSON object: `{ "questionTitle": "Updated Title", "level": "MEDIUM" }`. The response is a 404 Not Found error with a message: `{ "message": "Question with that ID is not present: 1", "status": "QuestionNotFoundException" }`.

```
1 {
2   "questionTitle": "Updated Title",
3   "level": "MEDIUM"
4 }
```

```
1 {
2   "message": "Question with that ID is not present: 1",
3   "status": "QuestionNotFoundException"
4 }
```

// **Create Quiz**

The screenshot shows a Postman interface with a POST request to `http://localhost:8080/api/v1/quiz/create?category=JAVA&level=EASY&title=Java Quiz`. The request parameters are: `category=JAVA`, `level=EASY`, and `title=Java Quiz`. The response is a 200 OK status with a JSON object: `{ "id": 1, "questions": [], "quizTitle": "Java Quiz" }`.

Key	Value	Description
<input checked="" type="checkbox"/> category	JAVA	
<input checked="" type="checkbox"/> level	EASY	
<input checked="" type="checkbox"/> title	Java Quiz	

```
1 {
2   "id": 1,
3   "questions": [],
4   "quizTitle": "Java Quiz"
5 }
```

//create Quiz

The screenshot shows the Postman interface with a POST request to `http://localhost:8080/api/v1/quiz/create?category=JAVA&level=EASY&title=Java Quiz`. The request is successful, returning a 200 OK status. The response body is a JSON object:

```
{
  "id": 2,
  "questions": [
    {
      "questionId": 6,
      "questionTitle": "What is the default value of a local variable in Java?",
      "option1": "0",
      "option2": "null",
      "option3": "Depends on the type",
      "option4": "No default value",
      "correctAnswer": "No default value",
      "category": "JAVA",
      "level": "EASY"
    }
  ]
}
```

//Get Quiz ById

The screenshot shows the Postman interface with a GET request to `http://localhost:8080/api/v1/quiz/getQuizById/2`. The request is successful, returning a 200 OK status. The response body is a JSON array containing one quiz object:

```
[
  {
    "id": 6,
    "questionTitle": "What is the default value of a local variable in Java?",
    "option1": "0",
    "option2": "null",
    "option3": "Depends on the type",
    "option4": "No default value"
  }
]
```

//Submit the Quiz

The image shows the Postman application interface. On the left sidebar, there's a 'My Workspace' section with 'New' and 'Import' buttons, and a 'Collections' section with a search bar and a 'New Collection' button. The main area displays a POST request to the URL `http://localhost:8080/api/v1/quiz/submitQuiz/1`. The request body is set to 'raw' and 'JSON' format, containing a JSON array of two objects. The first object has `"questionId": 2` and `"response": "option1"`. The second object has `"questionId": 3` and `"response": "option3"`. The response status is `200 OK` with a response time of `25 ms` and a size of `165 B`. The response body is shown in JSON format.

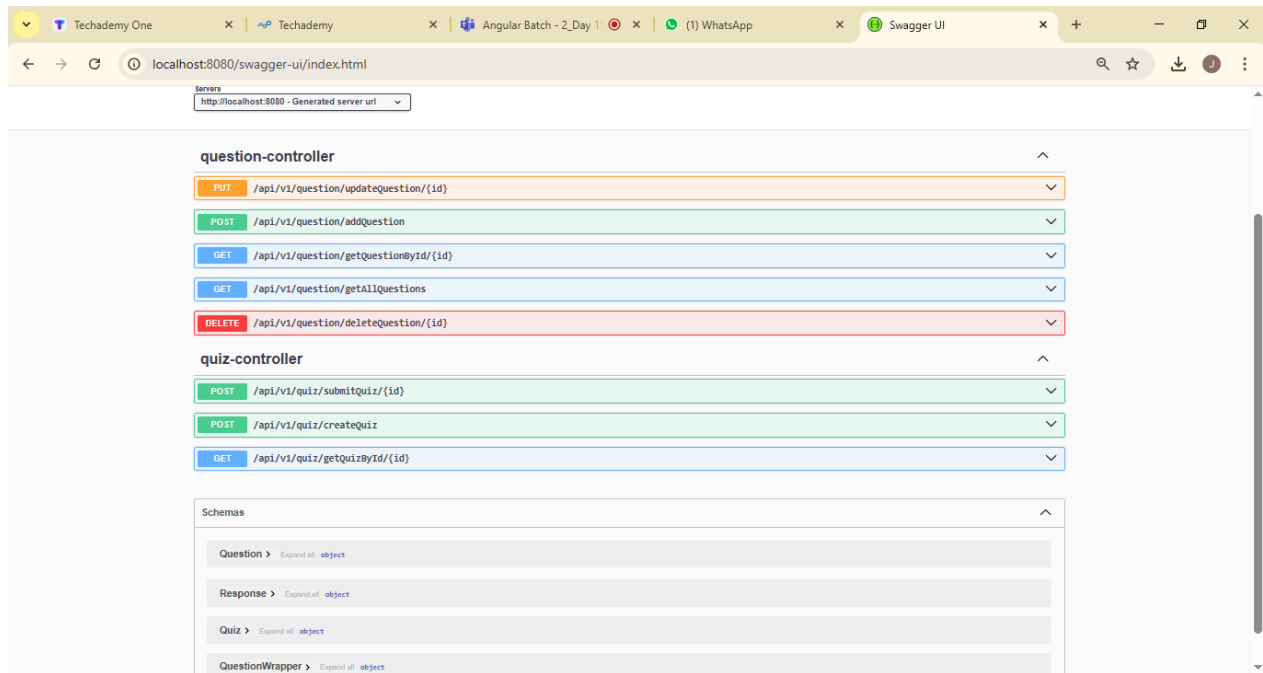
```
1 [
2   {
3     "questionId": 2,
4     "response": "option1"
5   },
6   {
7     "questionId": 3,
8     "response": "option3"
9   }
10 ]
11
```

Body Cookies Headers (5) Test Results

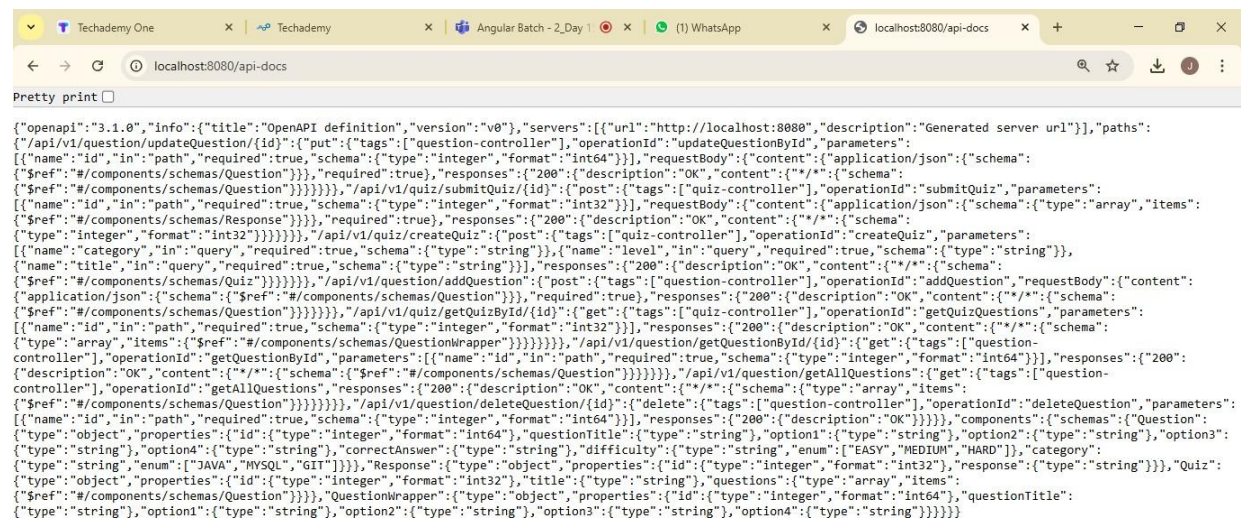
{ } JSON Preview Visualize

1

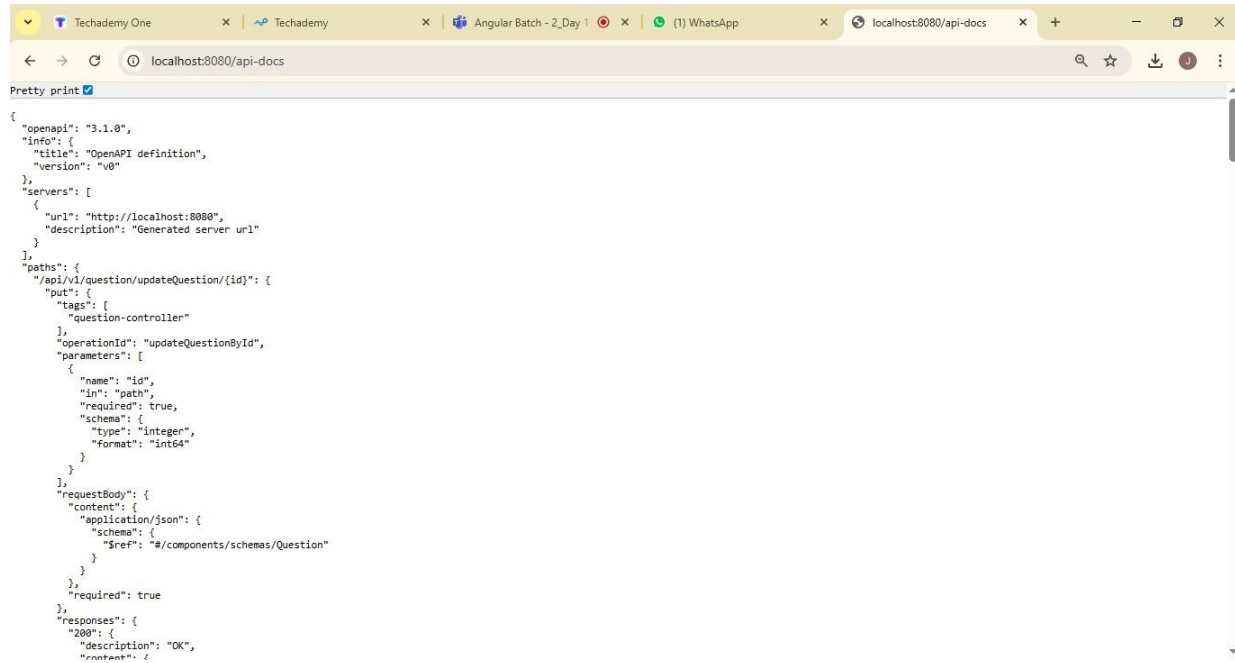
Swagger:



Api-Docs:



Pretty Print:



The screenshot shows a web browser with multiple tabs. The active tab is titled 'localhost:8080/api-docs'. The address bar shows the URL 'localhost:8080/api-docs'. The main content area displays a JSON object representing an OpenAPI definition, formatted in a 'Pretty print' view. The JSON structure includes metadata like 'openapi' version, 'info' (title, version), 'servers' (url, description), and a 'paths' section. The 'paths' section contains a single path '/api/v1/question/updateQuestion/{id}' with a 'put' method. This method has a 'question-controller' tag, an 'updateQuestionById' operationId, and a parameter 'id' of type 'integer' with format 'int64'. The request body is an object with a 'content' field containing an application/json schema that references '#/components/schemas/Question'. The response section shows a '200' status with a description 'OK'.

```
{
  "openapi": "3.1.0",
  "info": {
    "title": "OpenAPI definition",
    "version": "v0"
  },
  "servers": [
    {
      "url": "http://localhost:8080",
      "description": "Generated server url"
    }
  ],
  "paths": {
    "/api/v1/question/updateQuestion/{id}": {
      "put": {
        "tags": [
          "question-controller"
        ],
        "operationId": "updateQuestionById",
        "parameters": [
          {
            "name": "id",
            "in": "path",
            "required": true,
            "schema": {
              "type": "integer",
              "format": "int64"
            }
          }
        ],
        "requestBody": {
          "content": {
            "application/json": {
              "schema": {
                "$ref": "#/components/schemas/Question"
              }
            }
          },
          "required": true
        },
        "responses": {
          "200": {
            "description": "OK",
            "content": {}
          }
        }
      }
    }
  }
}
```