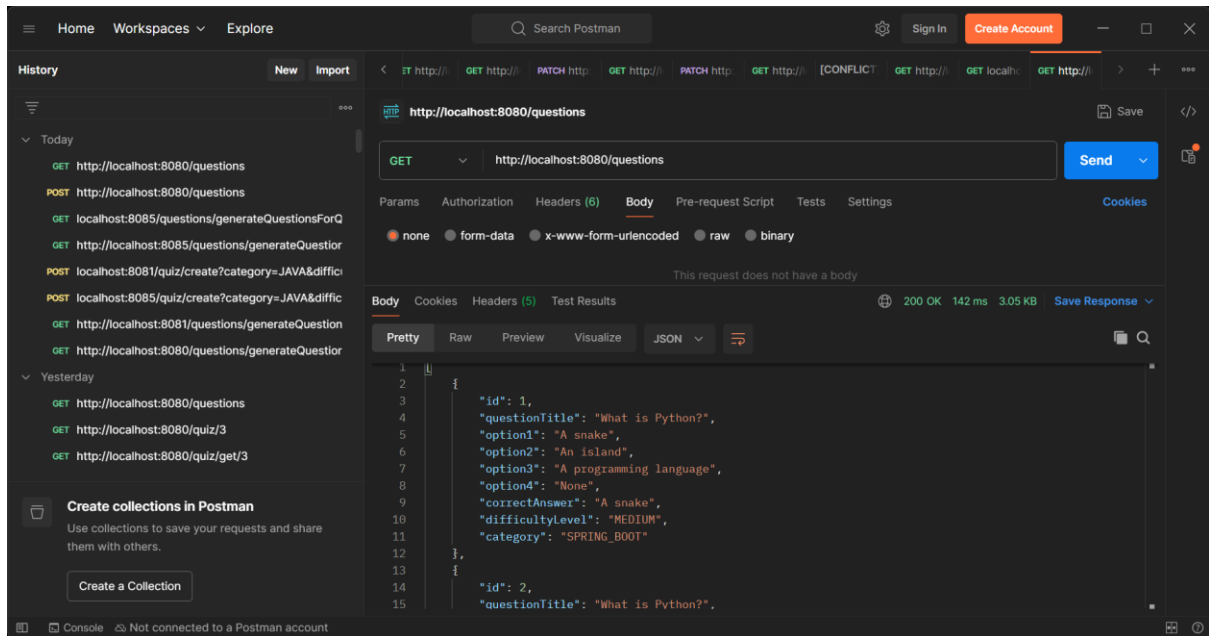
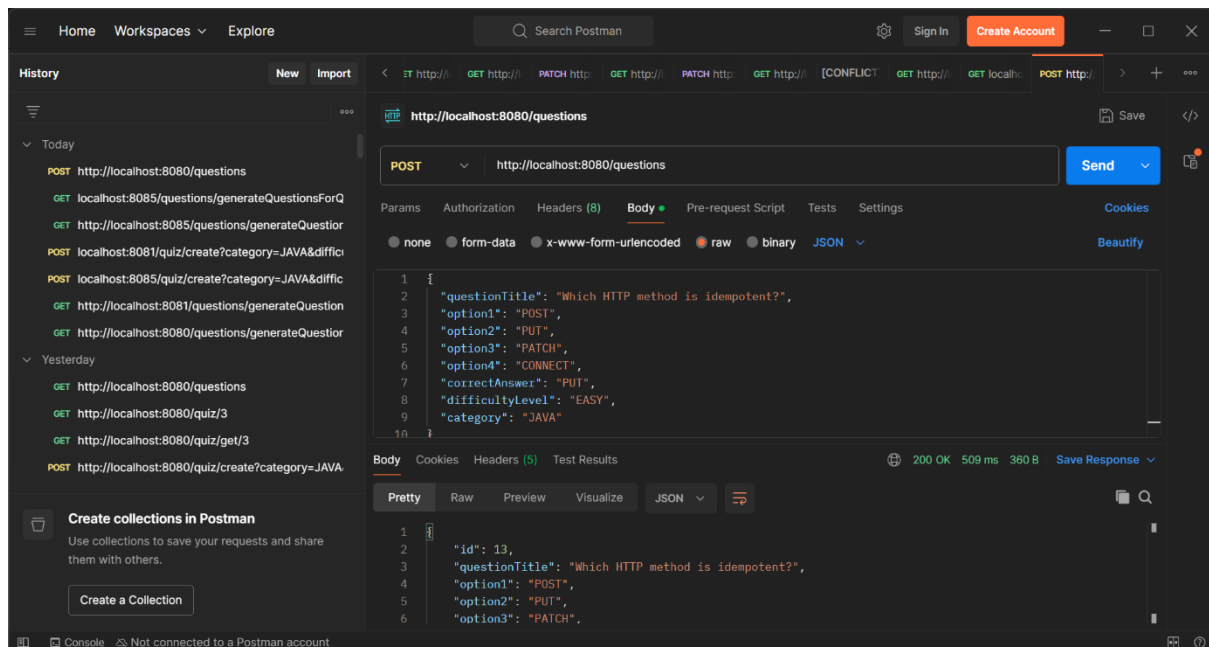


# OUTPUT SCREENSHOTS-QUIZAPP MONOREPO

## 1 Create a question



## 2. Get all questions



### 3. Pagination (get paged questions)

The screenshot shows the Postman application interface. The left sidebar displays a history of requests, including several GET requests to localhost:8080. The main panel shows a GET request to `http://localhost:8080/questions/paged?page=0&size=5`. The response is a JSON object with the following structure:

```
{
  "pageable": {
    "pageNumber": 0,
    "pageSize": 5,
    "sort": {
      "empty": true,
      "sorted": false,
      "unsorted": true
    }
  },
  "offset": 0,
  "paged": true,
  "unpaged": false
}
```

### 4. Update question title

The screenshot shows the Postman application interface. The left sidebar displays a history of requests, including several POST requests to localhost:8080. The main panel shows a PATCH request to `http://localhost:8080/questions/13/title`. The request body is a JSON object with the following structure:

```
{
  "id": 13,
  "questionTitle": "Which HTTP method is idempotent in java?",
  "option1": "POST",
  "option2": "PUT",
  "option3": "PATCH",
  "option4": "CONNECT",
  "correctAnswer": "PUT",
  "difficultyLevel": "EASY"
}
```

## 5. Update correct answer (store **real text**)

The screenshot shows the Postman interface with a PATCH request to `http://localhost:8080/questions/13/answer`. The request body is a JSON object: `{ "answer": "PUT" }`. The response is a JSON object: `{ "id": 13, "questionTitle": "Which HTTP method is idempotent in java?", "option1": "POST", "option2": "PUT", "option3": "PATCH", "option4": "CONNECT", "correctAnswer": "PUT", "difficultyLevel": "EASY" }`.

History:

- PATCH `http://localhost:8080/questions/13/answer`
- PATCH `http://localhost:8080/questions/13/title`
- PATCH `http://localhost:8080/questions/13/title`
- PATCH `http://localhost:8080/questions/13/title`
- PATCH `http://localhost:8080/questions/13/title`
- PATCH `http://localhost:8080/questions/13/title`
- PATCH `http://localhost:8080/questions/13/title`
- PATCH `http://localhost:8080/questions/13/title`
- GET `http://localhost:8080/questions/paged?page=0&s`
- GET `http://localhost:8080/questions`
- POST `http://localhost:8080/questions`
- GET `localhost:8085/questions/generateQuestionsForQ`

Create collections in Postman

Use collections to save your requests and share them with others.

Create a Collection

## 6. Delete question

The screenshot shows the Postman interface with a DELETE request to `http://localhost:8080/questions/12`. The response is a text message: `1 Question with ID 12 deleted successfully.`

History:

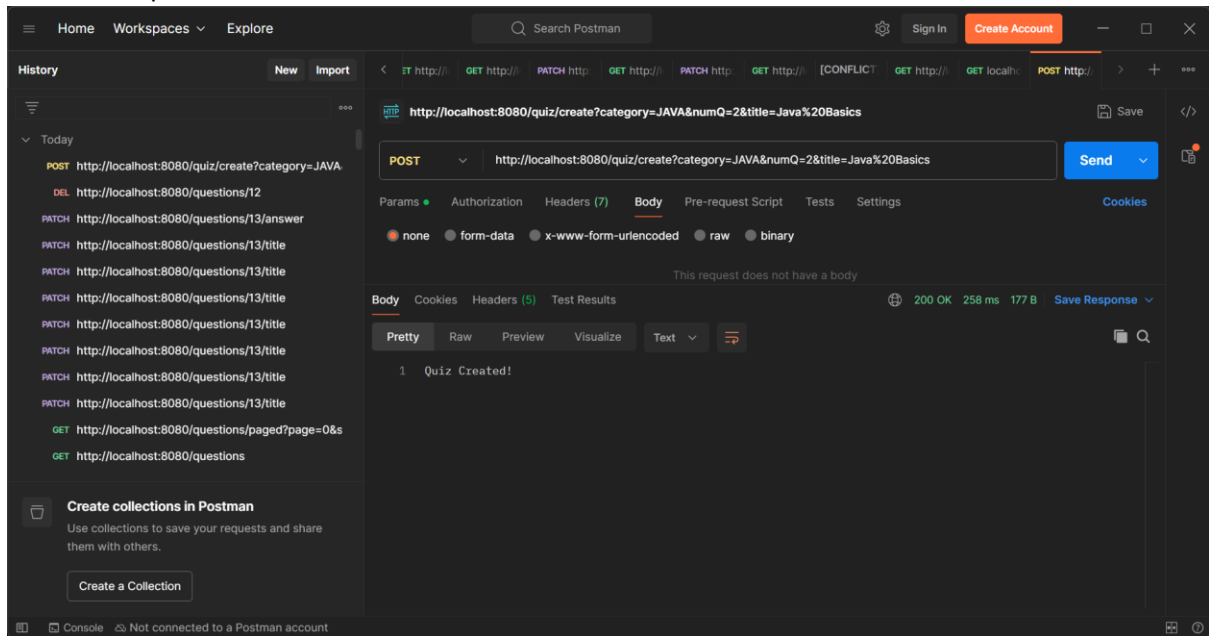
- DEL `http://localhost:8080/questions/12`
- PATCH `http://localhost:8080/questions/13/answer`
- PATCH `http://localhost:8080/questions/13/title`
- PATCH `http://localhost:8080/questions/13/title`
- PATCH `http://localhost:8080/questions/13/title`
- PATCH `http://localhost:8080/questions/13/title`
- PATCH `http://localhost:8080/questions/13/title`
- PATCH `http://localhost:8080/questions/13/title`
- PATCH `http://localhost:8080/questions/13/title`
- GET `http://localhost:8080/questions/paged?page=0&s`
- GET `http://localhost:8080/questions`
- POST `http://localhost:8080/questions`

Create collections in Postman

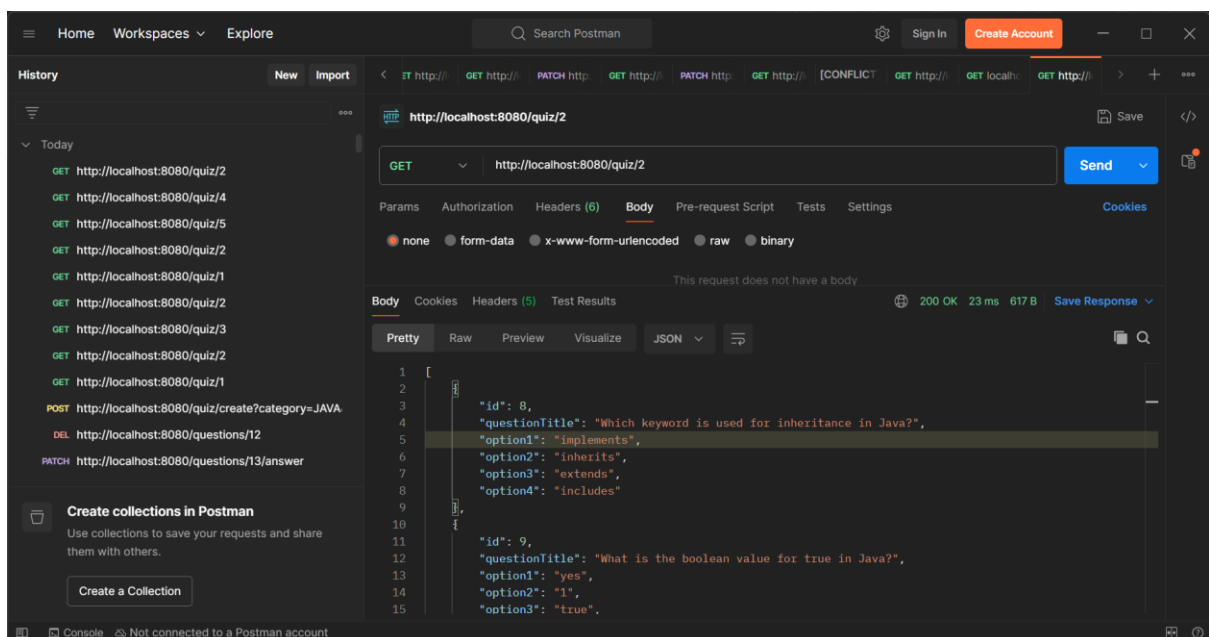
Use collections to save your requests and share them with others.

Create a Collection

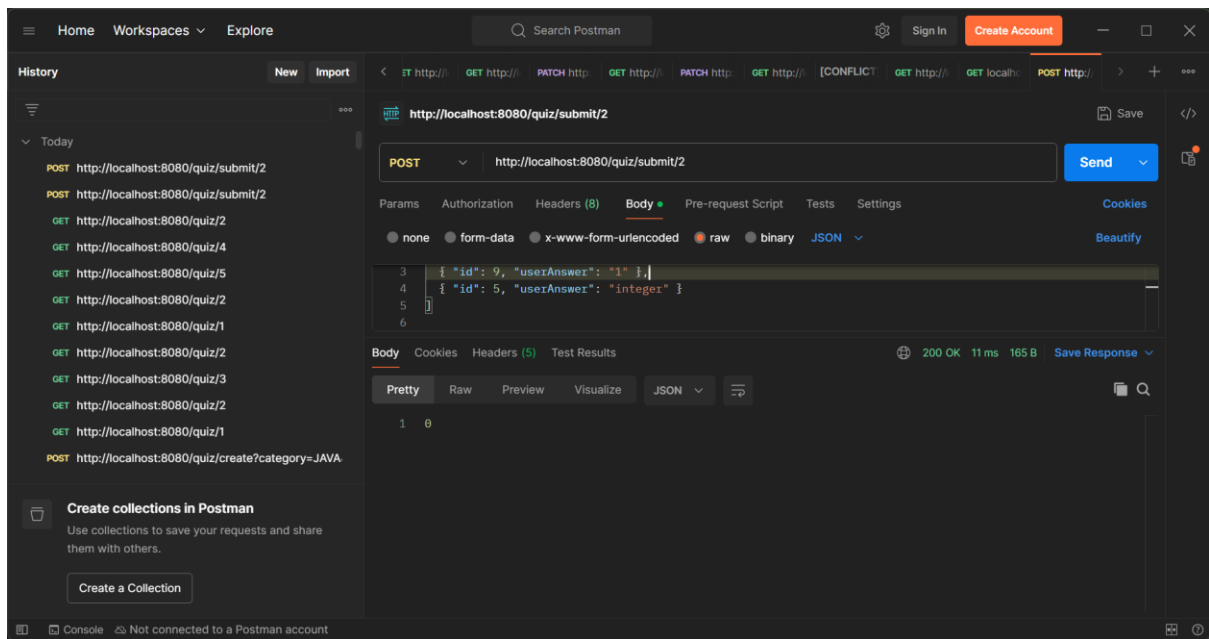
## 7. Create a quiz



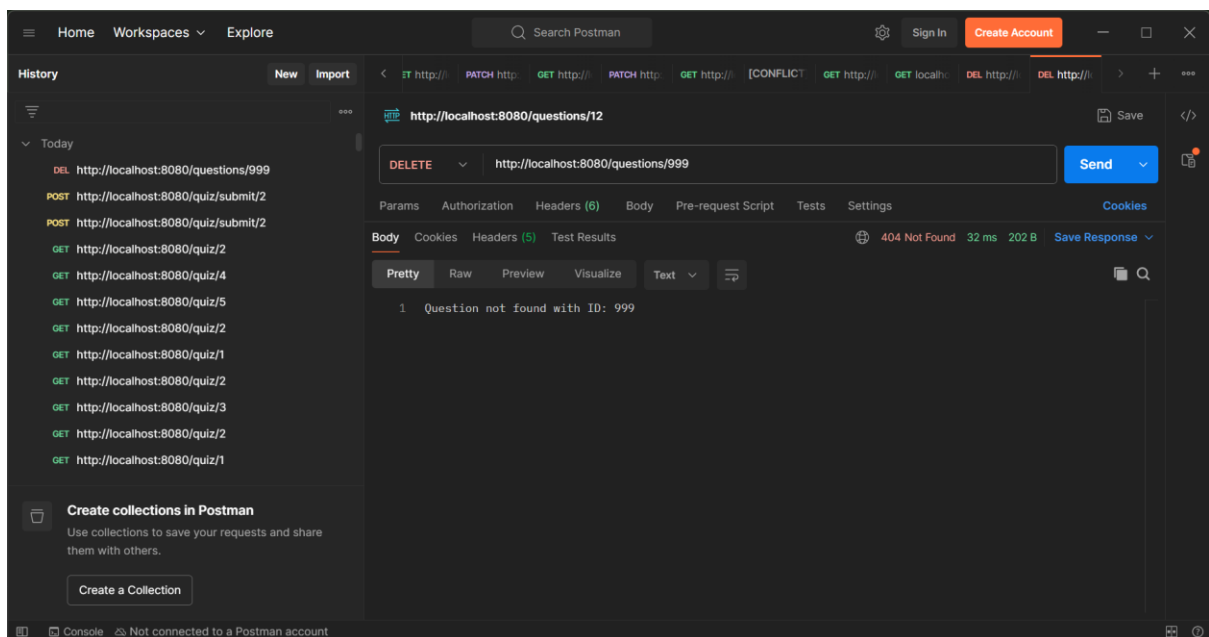
## 8. Get quiz questions (wrapper – no correct answers)



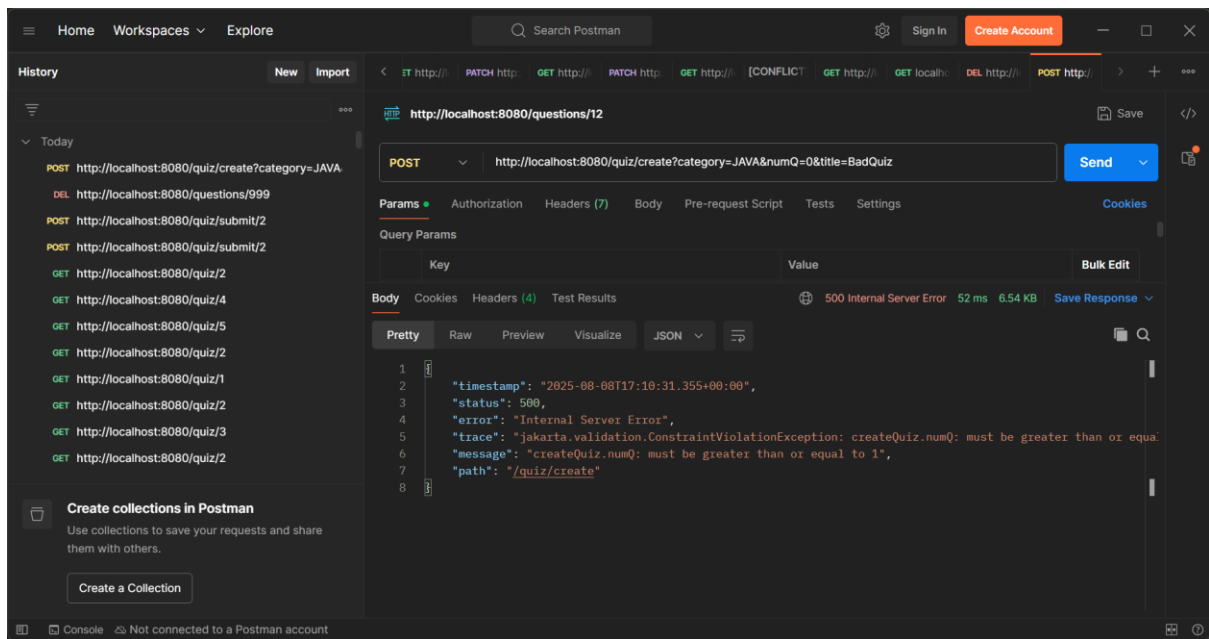
## 9. Submit quiz answers (using **real answer text**)



## 10. Custom Exception (404 from QuestionNotFoundException)



## 11. Validation



## 12. AOP (LoggingAspect)

