Microservice is calling a Microservice

Testing Inter-Service Communication

Now that we've set up our Quiz Service with Feign Client and registered both services with Eureka, we can test the communication between them. This represents a key milestone in our microservices journey - seeing two independent services working together seamlessly.

The Quiz Creation Process

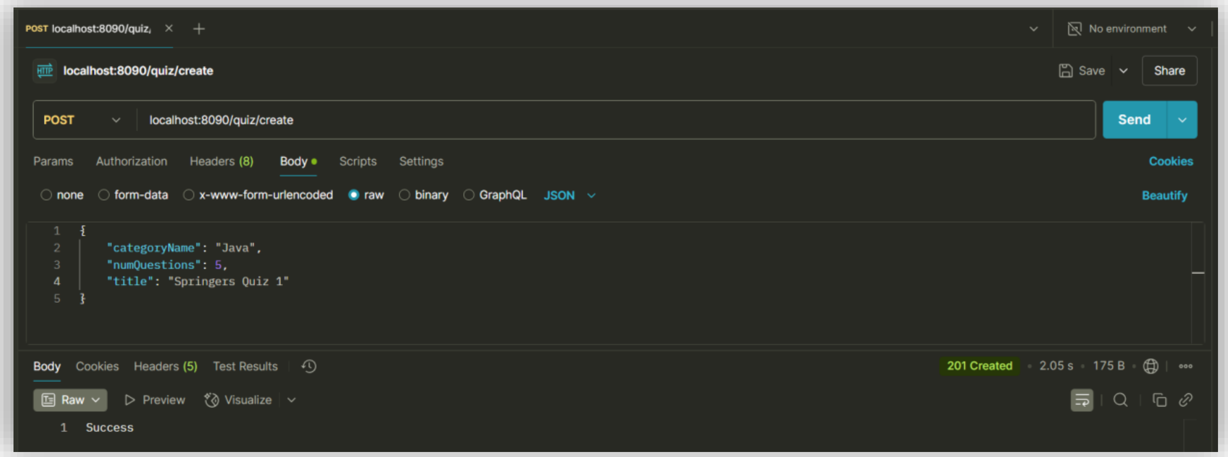
When a client sends a request to create a quiz, a chain of events occurs behind the scenes:

- The client sends a POST request to the Quiz Service with quiz parameters
- The Quiz Service uses Feign to call the Question Service
- The Question Service generates question IDs and returns them
- The Quiz Service stores these IDs and creates a new quiz entity

Testing with Postman

Let's test our API for creating a quiz using Postman. The endpoint expects a QuizDto object containing:

- **categoryName**: The category of questions to include
- **numQuestions**: How many questions to include
- **title**: The title of the quiz



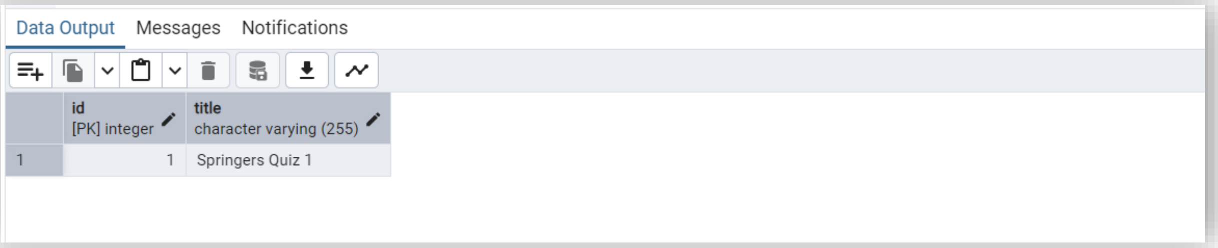
localhost:8090/quiz/create

POST localhost:8090/quiz/create

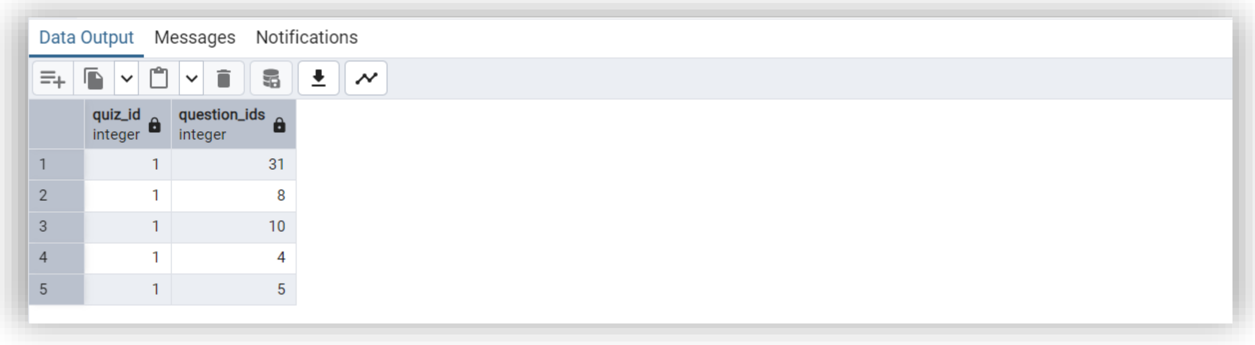
```
{
    "categoryName": "Java",
    "numQuestions": 5,
    "title": "Springers Quiz 1"
}
```

201 Created · 2.05 s · 175 B

```
1   Success
```



	id [PK] integer	title character varying (255)
1	1	Springers Quiz 1

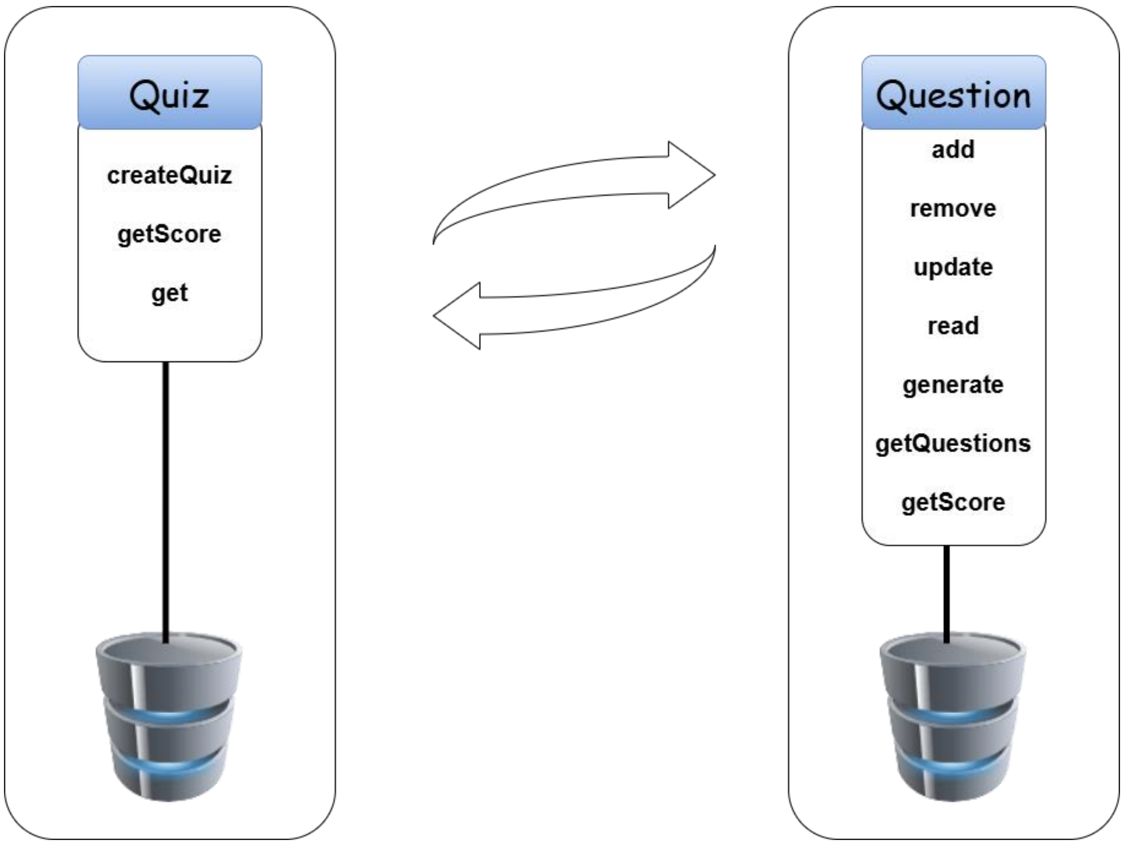


	quiz_id integer	question_ids integer
1	1	31
2	1	8
3	1	10
4	1	4
5	1	5

When we send this request to the Quiz Service, it internally communicates with the Question Service to fetch appropriate question IDs. We don't need to explicitly tell the Quiz Service how to reach the Question Service - that's handled automatically through service discovery.

Observing the Communication

By monitoring our application logs, we can observe that the Quiz Service is making requests to the Question Service. This confirms that our microservices architecture is functioning correctly:



This interaction demonstrates the power of microservices:

1. **Separation of Concerns**: Each service focuses on its specific functionality
2. **Independent Deployment**: Services can be updated separately
3. **Service Discovery**: No hardcoded URLs or service locations
4. **Declarative Communication**: Using Feign simplifies service-to-service calls

The Complete Flow

Let's review the complete flow of what happens when we create a quiz:

- **Client Request**: The client sends a POST request to **/quiz/create** with quiz details
- **Quiz Service**: Receives the request and extracts the QuizDto
- **Feign Client**: The Quiz Service uses the QuizInterface (Feign client) to call the Question Service
- **Service Discovery**: Eureka helps locate the Question Service instance
- **Question Service**: Processes the request and returns a list of question IDs
- **Database Operation**: The Quiz Service stores the quiz with the question IDs
- **Response**: The Quiz Service returns a success message to the client