1. Set up project

  a. *Dependency*

   Here are all dependencies that I used :

   • Spring web
   • Lombok
   • Swagger2
   • Swagger UI
   • Spring MongoDB

  b. *Docker*
   I set up a docker compose file in order to run mongodb and mongo-express container

```yaml
version: "3.8"
services:
  mongodb:
    image: mongo
    container_name: mongodb
    ports:
      - 27017:27017
    volumes:
      - data:/data
    environment:
      - MONGO_INITDB_ROOT_USERNAME=rootuser
      - MONGO_INITDB_ROOT_PASSWORD=rootpass
  mongo-express:
    image: mongo-express
    container_name: mongo-express
    restart: always
    ports:
      - 8081:8081
    environment:
      - ME_CONFIG_MONGODB_ADMINUSERNAME=rootuser
      - ME_CONFIG_MONGODB_ADMINPASSWORD=rootpass
      - ME_CONFIG_MONGODB_SERVER=mongodb
volumes:
  data: {}

networks:
  default:
    name: mongodb_network
```

  c. *Swagger configuration*

```java
@EnableSwagger2
@Configuration
public class SwaggerConfig {
    @Bean
    public Docket api() {
        return new Docket(DocumentationType.SWAGGER_2)
                .select()
                .apis(RequestHandlerSelectors.any())
                .paths(PathSelectors.any())
                .build();
    }
}
```

2.   API Rest

- FizzBuzz Document: to save into my mongo database
- FizzBuzz Repository: extends MongoRespository
- FizzBuzzController: where I build my APIs REST

   a.   GetMapping("/fizz")

This API Rest with method GET will receive 5 Params as int1, int2, limit, str1, str2, and send to client a List of String contain Fizz Buzz game. In the means time, I check the request if this request is already in my database. If the request is exist, I will update the counter attribute, if not, insert a new request to database with default counter = 1.
   Query:

```
43          }
44          Query query = new Query().addCriteria(Criteria.where( key: "int1").is(int1))
45                  .addCriteria((Criteria.where( key: "int2").is(int2)))
46                  .addCriteria((Criteria.where( key: "limit").is(limit)))
47                  .addCriteria((Criteria.where( key: "str1").is(str1)))
48                  .addCriteria((Criteria.where( key: "str2").is(str2)))
49                  ;
50          FizzBuzz check = mongoTemplate.findOne(query,FizzBuzz.class);
51          if (check != null){
52              check.setCount(check.getCount()+1);
53              mongoTemplate.save(check);
54          }
55          else{
56              FizzBuzz fizz = new FizzBuzz(int1,int2,limit,str1,str2);
57              this.fizzBuzzRepository.insert(fizz);
58          }
```

   b.   GetMapping("/")

An API to send the client the most frequent request has been hits. I sort all database by counter and get the first element.

```
@GetMapping("/")
public FizzBuzz getMostRequest(){
    return fizzBuzzRepository.findAll(Sort.by(Sort.Direction.DESC, ...properties: "count")).get(0);
}
}
```

3.   Generate API Documentation

Here is API Documentation with Swagger UI

{·} swagger		default (/v2/api-docs) ⌄	Explore

Api Documentation

Api Documentation

Apache 2.0

| **basic-error-controller** : Basic Error Controller	Show/Hide	List Operations	Expand Operations
fizz-buzz-controller : Fizz Buzz Controller	Show/Hide	List Operations	Expand Operations

GET	/	getMostRequest
GET	/fizz	fizz

[BASE URL: /, API VERSION: 1.0]

Or we can send Get request to URL http://localhost:8080/v2/api-docs
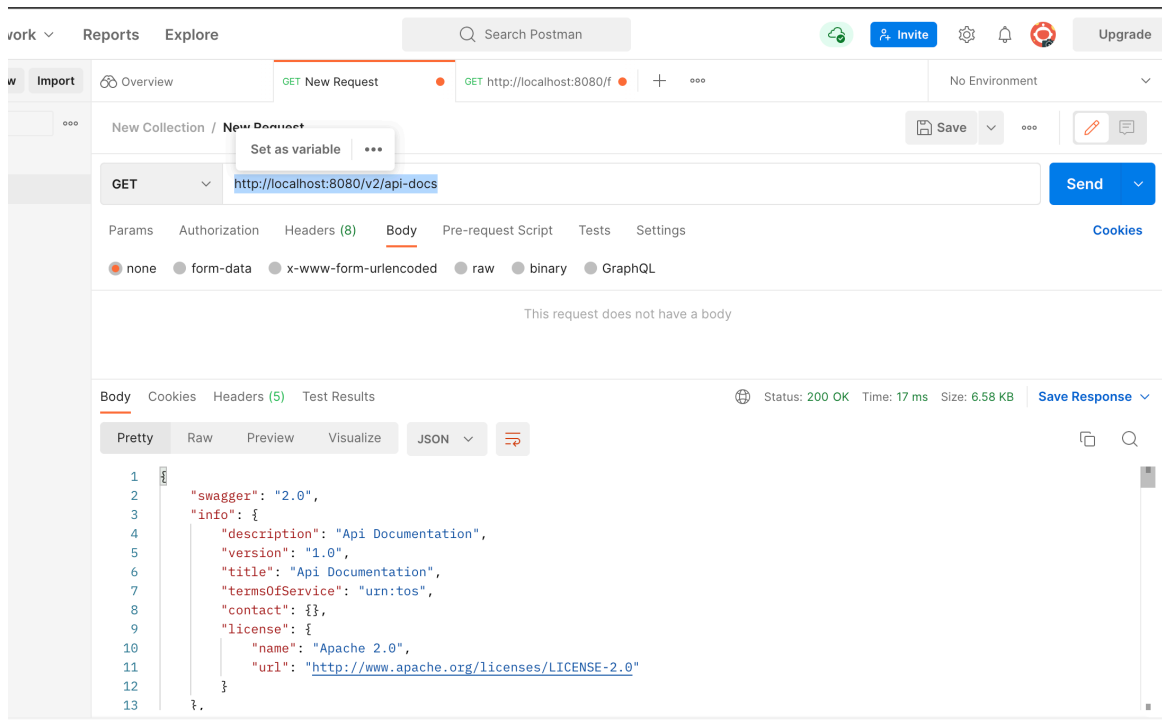
4. Test JUnit

I wrote a simple JUnit test to test my program and make sure the API's status is 200
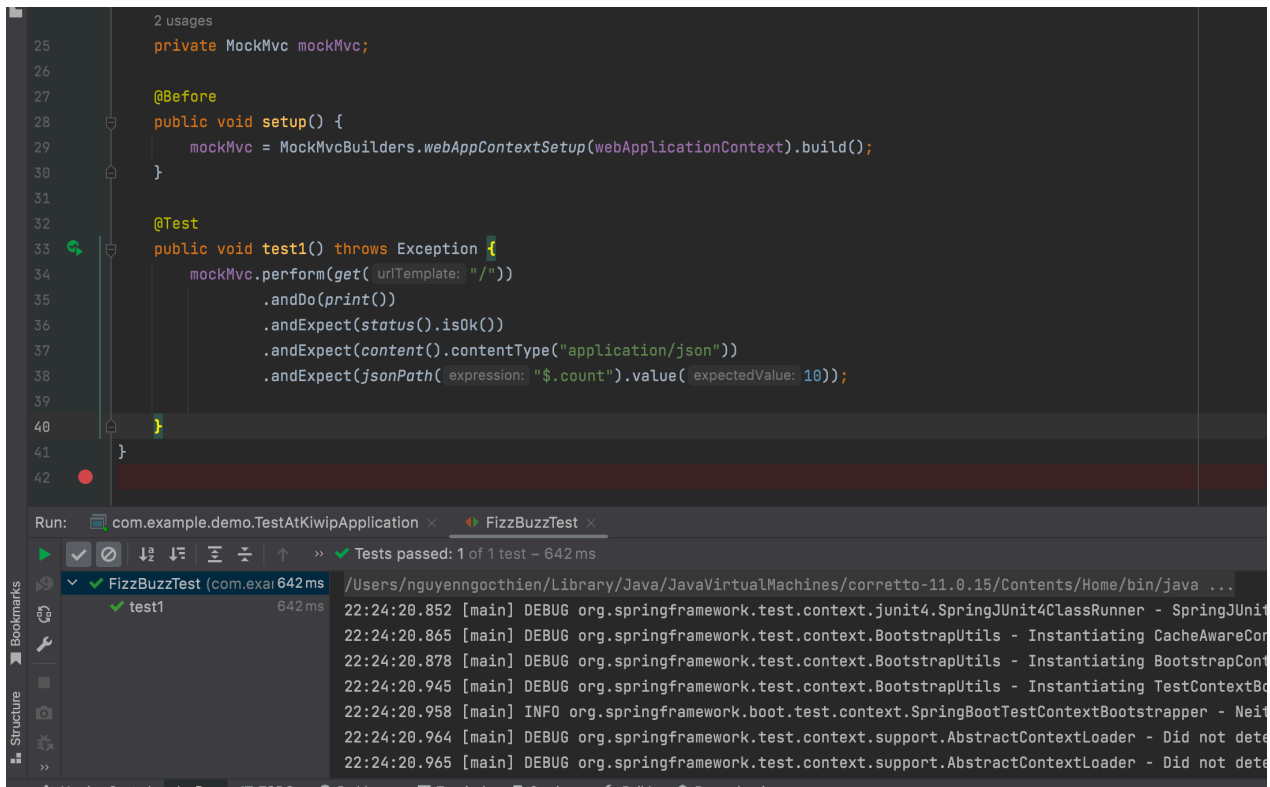
The database I used for the test (second lines is the most frequent request):

_id	int1	int2	limit	str1	str2	count	_class
62971f2c1d59c32c53a79e2b	3	5	100	fizz	buzz	1	com.example.demo.FizzBuzz
62976cf8a98cc60e2e8498c4	3	5	100	fizz	hehe	10	com.example.demo.FizzBuzz
62976ee8b3534309639bc03a	3	5	100	fizz	keke	4	com.example.demo.FizzBuzz

Interesting test :). It took me about 3h to complete all the task. The API Rest is really quick to finish but I spent a lot of time to debug some conflit between the configuration of dependencies (Swagger, MongoDB and JDK version also)