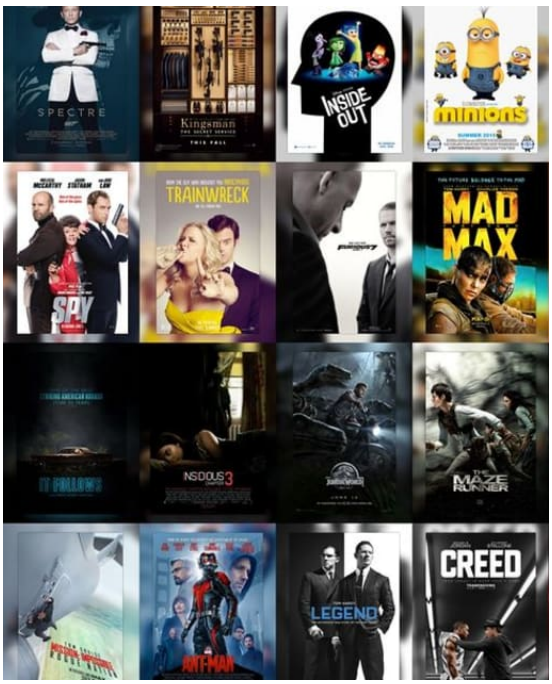
Movies Battle

25/02/2022



Kauane Splett da Costa
Curitiba, Paraná

Visão geral

O projeto Movies Battle teve como objetivo criar uma API REST para um jogo no estilo card game, onde são informados dois filmes e o jogador deve acertar aquele que possui melhor avaliação no IMDB.

Especificações

O projeto foi criado utilizando Spring Web, Boot, Data, JPA e Security. A versão do java utilizado foi 11 e a IDE foi o Eclipse. O Maven foi utilizado para gerenciamento de dependência. Através do Spring Framework é utilizado o conceito da injeção de dependências. O Spring utiliza um container chamado Spring IoC Container que coordena as dependências do projeto automaticamente.

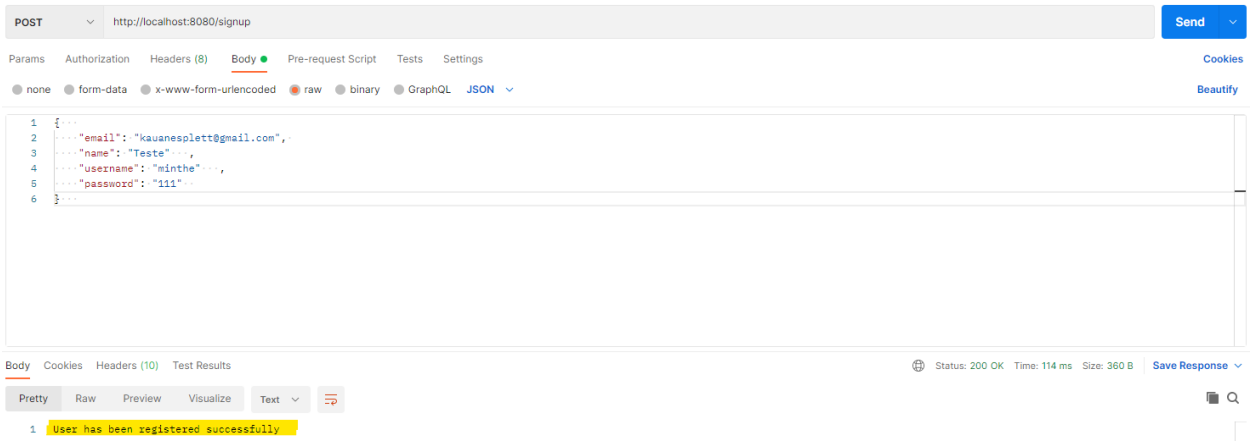
Integração com OMDBApi e Web scraping do IMD Top 250

Ao inicializar a aplicação é feito o carregamento de 30 filmes. Para realizar esse carregamento foi realizado o web scraping, que é o processo de extrair informações relevantes de determinado site, do link IMDB 250 Top Movies. Após capturar aleatoriamente os IMDB ID's, é feita a consulta no OMDB Api. Esse processo foi escolhido para inserir na aplicação filmes conhecidos, não apenas filmes antigos e não populares.

Fluxo e documentação

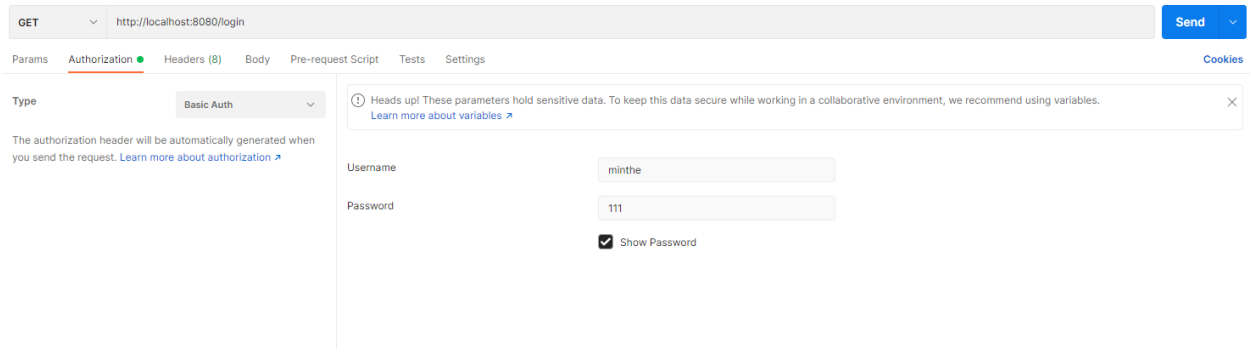
I. Criação de um novo jogador

A criação de um novo usuário é realizada através do endpoint **POST /signup**, informando os dados do novo jogador.



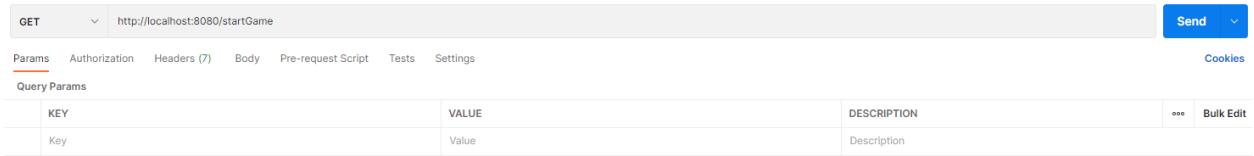
II. Login

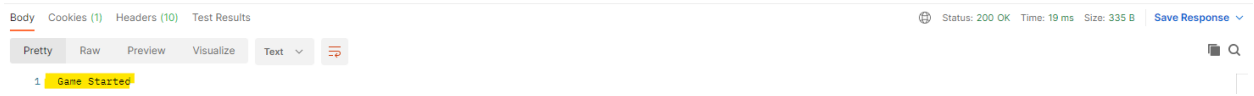
Através do endpoint **/login** é possível realizar o login na aplicação. O tipo de autenticação utilizada no projeto foi o Basic Auth e o Spring Security utilizado para gerenciamento de acessos.



III. Iniciar um novo jogo

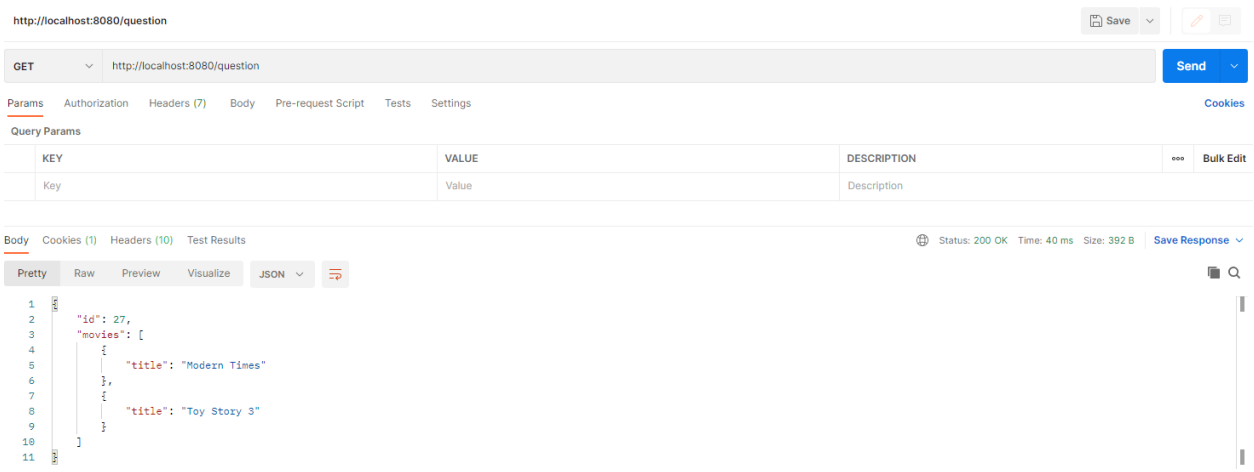
Através do endpoint **GET /startGame** é possível iniciar um novo jogo. Só é possível iniciar um jogo por vez. Se tentar iniciar mais de um jogo será apresentada uma mensagem.





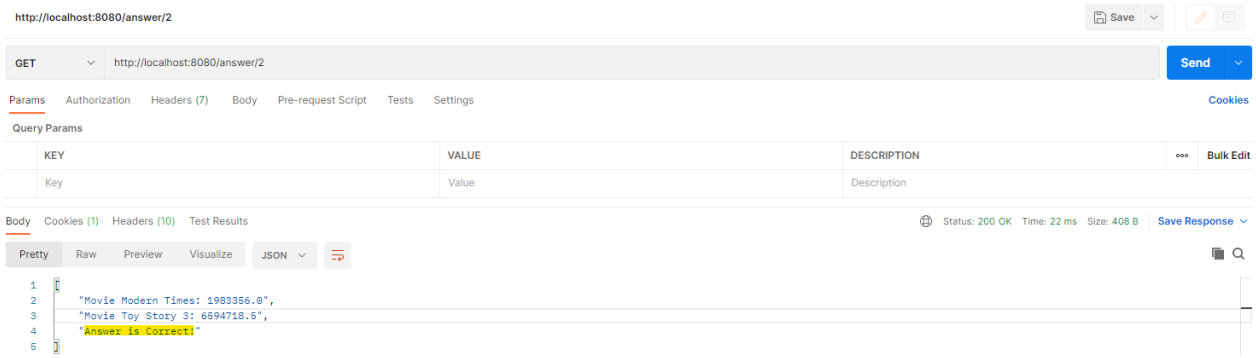
IV. Verificar a questão corrente

No endpoint **GET /question** é possível verificar a pergunta corrente. Ao acessar esse endpoint é apresentada a questão corrente, ela só irá se alterar após a resposta ou finalização do jogo.



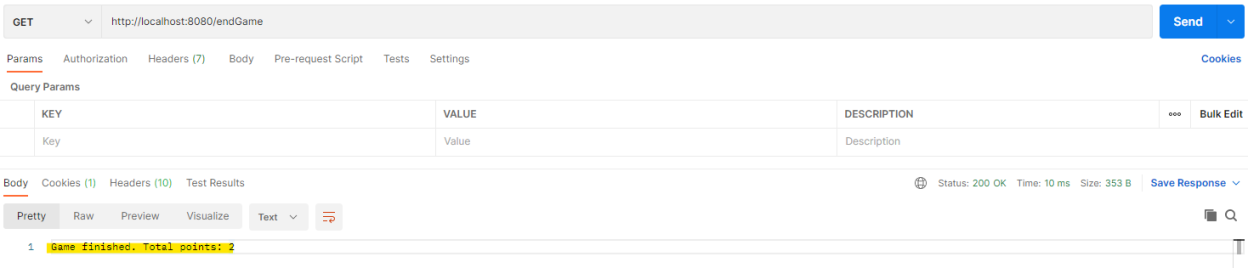
V. Responder a pergunta atual

O endpoint **GET /answer/{opcao}** permite ao jogador responder a pergunta, escolhendo o filme 1 ou o filme 2, e informa se a resposta estava correta ou não. Caso acerte, o ponto é salvo no jogo corrente. Ao errar 3 vezes o jogo é finalizado.



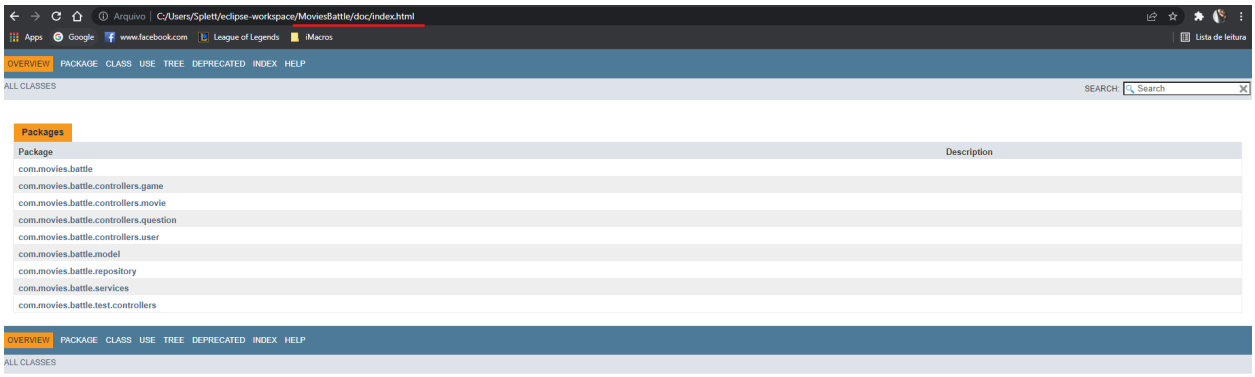
VI. Finalizar o jogo

No endpoint **GET /endGame** é possível finalizar o jogo. Como resposta é apresentado o total de pontos do jogo finalizado.



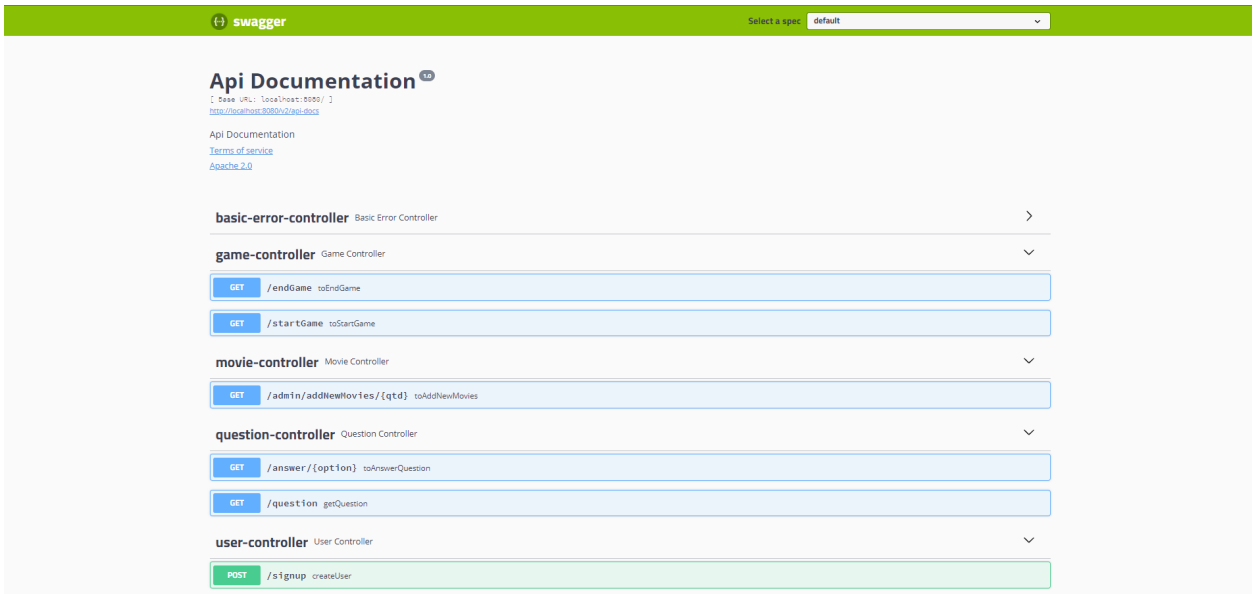
VII. Documentação do código via Java Doc

Javadoc é um gerador de documentação para a linguagem Java com o objetivo de gerar a documentação a partir do código-fonte Java no formato HTML. A documentação está disponível na pasta Doc do programa.



VIII. Documentação dos endpoints via Swagger UI

O Swagger é uma aplicação open source que auxilia a definir, criar, documentar e consumir APIs REST. Foi utilizado no projeto com a finalidade de gerar a documentação dos endpoints com base no Open API.



IX. Testes via JUnit

O JUnit foi utilizado para criação e execução dos testes automatizados e estão disponíveis na pasta **/src/test/java**.