Código fonte: [GitHub](https://github.com/MykyBalb/Atividade_avaliativa)

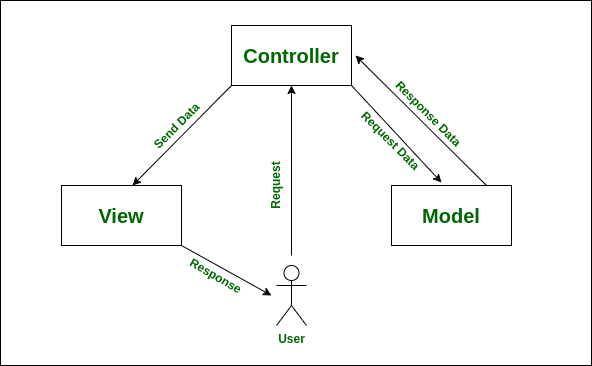
**Aplicação Biblioteca**

A aplicação será destinada à uma biblioteca que quer armazenar Clientes, Empréstimos e Livros. Para isso vamos usar Java Spring.

O sistema será construído com as dependências do Spring Web, que, resumidamente, é utilizado para criar um servidor web local Java, facilitando o trabalho do desenvolvedor em poucos cliques, coisas que não aconteceria se usássemos outra framework ou configurarmos uma do zero, tal qual o Tomcat.

Arquitetura MVC

A arquitetura a ser usada será o MVC(model, view, controller), que segue essa estrutura:

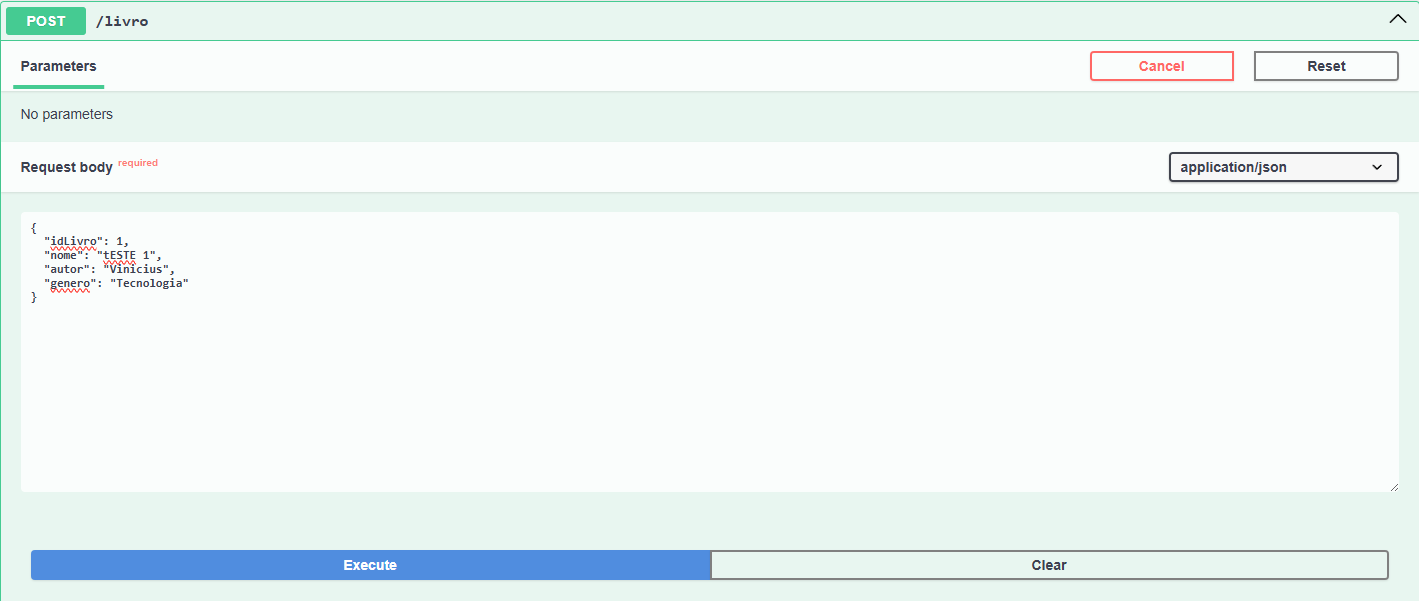


Dessa forma, a camada Model fica responsável por representar os dados da aplicação; a camada View é a interface do usuário, sendo responsável por mostrar ao usuário as informações; a camada Controller é encarregada de ser a base intermediária entre Model e View, recebendo as informações do usuário, processando e mostrando na View.

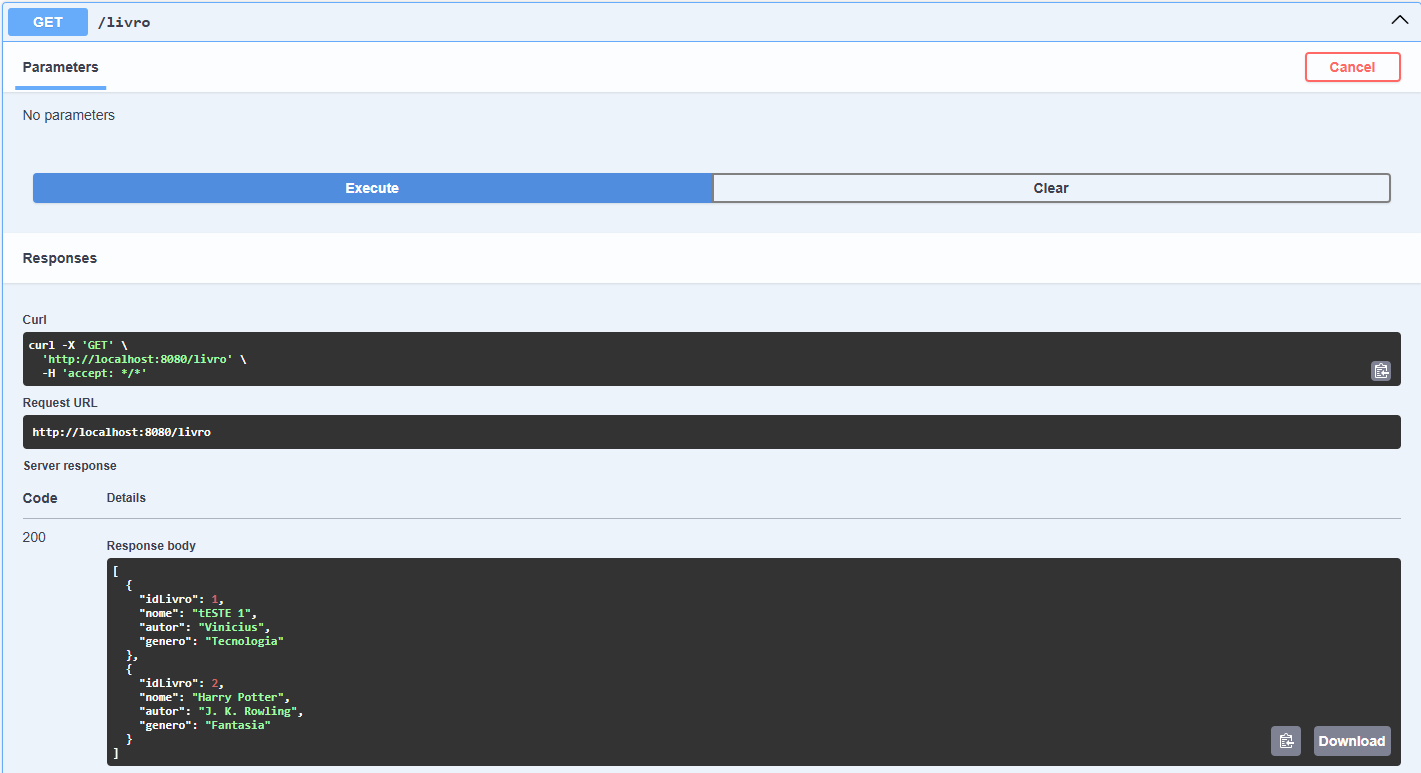
Mais Sobre o Sistema

O sistema deverá conter um sistema CRUD(create, read, update, delete), também conhecido como FUCK(find, update, create, kill) para registrar, ver, modificar e excluir os dados do nosso banco e faremos isso a partir de um sistema improvisado encarregado de ser nosso banco de dados em forma de um ArrayList.

Já na construção do Back-End, devemos receber dados e processá-los para armazenar nossos dados no “banco”. Iremos fazer isso através de endpoints que, futuramente, serão requisitados pelo Front-End, mas que, por enquanto, só será utilizado pelo Swagger, para melhor visualização.



Na imagem, há o verbo POST do http, que realiza a criação de um dado, no caso, os dados do livro, mas que funciona da mesma forma para as outras classes. Pelo swagger, a entrada é feita a partir de um arquivo JSON(JavaScript Object Notation), que é o padrão utilizado pelo Spring, mas que pode ser alterado para XML.



Aqui temos o verbo GET que é responsável por mostrar os dados armazenados, no nosso caso, na classe database. Porém, o padrão e mais recomendável é utilizar um banco de dados e Spring JPA para registrar da melhor forma possível.

E os próximos verbos são PUT e DELETE. O PUT é utilizado para modificar/atualizar um dado e é, geralmente modificado pelo o ID de um elemento que é atribuído em sua criação.

O DELETE também utiliza o ID como parâmetro para realizar sua função: deletar um elemento/tupla.

Lista Design Patterns

1. Padrões Criacionais:

* Factory Method
* Abstract Factory
* Builder
* Prototype
* Singleton

2. Padrões Estruturais:

* Adapter
* Bridge
* Composite
* Decorator
* Facade
* Flyweight
* Proxy

3. Padrões Comportamentais:

* Chain of Responsibility
* Command
* Interpreter
* Iterator
* Mediator
* Memento
* Observer
* State
* Strategy
* Template Method
* Visitor

Olhando para o nosso projeto, poderíamos usar o Singleton para unificar a instância de um objeto para todas as três classes e conseguir um relacionamento entre as mesmas, através deste objeto.