Minhas anotações do projeto:

API Application Interface - é o que fica exposta para WEB. é a porta de entrada do seu back-end.

REST - é um padrão que especifica como uma API deve ser implementada para ser considerada REST. Seguindo os padrões do REST você terá uma API REST :

1 - Tem que ser uma aplicação cliente/servidor;

2 - Comunicação é stateless ( não guarda status, ou seja o resultado da requisição não

depende de algo que o sistema deva guardar para retornar o resultado. Ex. o retorno

não depende de algo que precisa estar armazenado previamente na sessão do usuário

para retornar )

3 - Interface uniforme com formato padronizado utilizando os verbos http( utilização

dos verbos http para realização das operações )

GET - obter

POST - gravar

PUT - alterar ( verbo idempotente significa que uma requisição realizada 1 vez

ou 10 vez o resultado será o mesmo )

DELETE - deletar

4 - Utilização de Cache

5 - Sistema em camadas

6 - Código sob demanda ( opcional )

Uma porta com maçaneta podemos atribuir a maçaneta como sendo a API ( a interface )

a porta de entrada para o seu back-end ( que podemos considerar sendo a porta )

Camadas:

- Front End

- Back End

- Controladores REST ( Os controladores são as portas de entrada/interface = API )

- Camada do Serviços ( realizada as transações )

- Camada de Acesso a Dados ( realizada as transações )

Configuração do projeto dslist ( pasta: API\_REST\_LojaDeGames )

Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Dependências do projeto: pom.xml

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Como solução para resolver alguns casos de incompatibilidade do maven, devemos indicar a versão do maven que desejamos utilizar no pom.xml: ( opcional, caso aconteça o problema segue a solução )

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-resources-plugin</artifactId>

<version>3.1.0</version>

</plugin>

Configurar o arquivo application.properties para ter vários perfis de ambiente: teste, homologação ...

1 – no application.properties ( perfil de teste )

#criando um perfil de teste

spring.profiles.active=${APP\_PROFILE:test}

spring.jpa.open-in-view=false

cors.origins=${CORS\_ORIGINS:http://localhost:5173,http://localhost:3000}

2 – criar um outro application.properties com o nome application-test.properties

#H2 Connection - dados de conexao com o banco de dados

spring.datasource.url=jdbc:h2:mem:testedb

spring.datasource.username=sa

spring.datasource.password=

#H2 Client - habilita o console para ser utilizado no browse

spring.h2.console.enabled=true

spring.h2.console.path=/h2-console

#Show H2 - habilita a exibicao das queries SQL no console do STS

spring.jpa.show-sql=true

spring.jpa.properties.hibernate.format\_sql=true

3 – Seed no tabela de game. Criar o arquivo import.sql na pasta resource.

Primeiro insert na tabela tb\_game:

INSERT INTO tb\_game (title, score, game\_year, genre, platforms, img\_url, short\_description, long\_description) VALUES ('Mass Effect Trilogy', 4.8, 2012, 'Role-playing (RPG), Shooter', 'XBox, Playstation, PC', 'https://raw.githubusercontent.com/devsuperior/java-spring-dslist/main/resources/1.png', 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Odit esse officiis corrupti unde repellat non quibusdam! Id nihil itaque ipsum!', 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Delectus dolorum illum placeat eligendi, quis maiores veniam. Incidunt dolorum, nisi deleniti dicta odit voluptatem nam provident temporibus reprehenderit blanditiis consectetur tenetur. Dignissimos blanditiis quod corporis iste, aliquid perspiciatis architecto quasi tempore ipsam voluptates ea ad distinctio, sapiente qui, amet quidem culpa.');

INSERT INTO tb\_game (title, score, game\_year, genre, platforms, img\_url, short\_description, long\_description) VALUES ('Red Dead Redemption 2', 4.7, 2018, 'Role-playing (RPG), Adventure', 'XBox, Playstation, PC', 'https://raw.githubusercontent.com/devsuperior/java-spring-dslist/main/resources/2.png', 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Odit esse officiis corrupti unde repellat non quibusdam! Id nihil itaque ipsum!', 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Delectus dolorum illum placeat eligendi, quis maiores veniam. Incidunt dolorum, nisi deleniti dicta odit voluptatem nam provident temporibus reprehenderit blanditiis consectetur tenetur. Dignissimos blanditiis quod corporis iste, aliquid perspiciatis architecto quasi tempore ipsam voluptates ea ad distinctio, sapiente qui, amet quidem culpa.');

INSERT INTO tb\_game (title, score, game\_year, genre, platforms, img\_url, short\_description, long\_description) VALUES ('The Witcher 3: Wild Hunt', 4.7, 2014, 'Role-playing (RPG), Adventure', 'XBox, Playstation, PC', 'https://raw.githubusercontent.com/devsuperior/java-spring-dslist/main/resources/3.png', 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Odit esse officiis corrupti unde repellat non quibusdam! Id nihil itaque ipsum!', 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Delectus dolorum illum placeat eligendi, quis maiores veniam. Incidunt dolorum, nisi deleniti dicta odit voluptatem nam provident temporibus reprehenderit blanditiis consectetur tenetur. Dignissimos blanditiis quod corporis iste, aliquid perspiciatis architecto quasi tempore ipsam voluptates ea ad distinctio, sapiente qui, amet quidem culpa.');

INSERT INTO tb\_game (title, score, game\_year, genre, platforms, img\_url, short\_description, long\_description) VALUES ('Sekiro: Shadows Die Twice', 3.8, 2019, 'Role-playing (RPG), Adventure', 'XBox, Playstation, PC', 'https://raw.githubusercontent.com/devsuperior/java-spring-dslist/main/resources/4.png', 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Odit esse officiis corrupti unde repellat non quibusdam! Id nihil itaque ipsum!', 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Delectus dolorum illum placeat eligendi, quis maiores veniam. Incidunt dolorum, nisi deleniti dicta odit voluptatem nam provident temporibus reprehenderit blanditiis consectetur tenetur. Dignissimos blanditiis quod corporis iste, aliquid perspiciatis architecto quasi tempore ipsam voluptates ea ad distinctio, sapiente qui, amet quidem culpa.');

INSERT INTO tb\_game (title, score, game\_year, genre, platforms, img\_url, short\_description, long\_description) VALUES ('Ghost of Tsushima', 4.6, 2012, 'Role-playing (RPG), Adventure', 'XBox, Playstation, PC', 'https://raw.githubusercontent.com/devsuperior/java-spring-dslist/main/resources/5.png', 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Odit esse officiis corrupti unde repellat non quibusdam! Id nihil itaque ipsum!', 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Delectus dolorum illum placeat eligendi, quis maiores veniam. Incidunt dolorum, nisi deleniti dicta odit voluptatem nam provident temporibus reprehenderit blanditiis consectetur tenetur. Dignissimos blanditiis quod corporis iste, aliquid perspiciatis architecto quasi tempore ipsam voluptates ea ad distinctio, sapiente qui, amet quidem culpa.');

INSERT INTO tb\_game (title, score, game\_year, genre, platforms, img\_url, short\_description, long\_description) VALUES ('Super Mario World', 4.7, 1990, 'Platform', 'Super Ness, PC', 'https://raw.githubusercontent.com/devsuperior/java-spring-dslist/main/resources/6.png', 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Odit esse officiis corrupti unde repellat non quibusdam! Id nihil itaque ipsum!', 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Delectus dolorum illum placeat eligendi, quis maiores veniam. Incidunt dolorum, nisi deleniti dicta odit voluptatem nam provident temporibus reprehenderit blanditiis consectetur tenetur. Dignissimos blanditiis quod corporis iste, aliquid perspiciatis architecto quasi tempore ipsam voluptates ea ad distinctio, sapiente qui, amet quidem culpa.');

INSERT INTO tb\_game (title, score, game\_year, genre, platforms, img\_url, short\_description, long\_description) VALUES ('Hollow Knight', 4.6, 2017, 'Platform', 'XBox, Playstation, PC', 'https://raw.githubusercontent.com/devsuperior/java-spring-dslist/main/resources/7.png', 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Odit esse officiis corrupti unde repellat non quibusdam! Id nihil itaque ipsum!', 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Delectus dolorum illum placeat eligendi, quis maiores veniam. Incidunt dolorum, nisi deleniti dicta odit voluptatem nam provident temporibus reprehenderit blanditiis consectetur tenetur. Dignissimos blanditiis quod corporis iste, aliquid perspiciatis architecto quasi tempore ipsam voluptates ea ad distinctio, sapiente qui, amet quidem culpa.');

INSERT INTO tb\_game (title, score, game\_year, genre, platforms, img\_url, short\_description, long\_description) VALUES ('Ori and the Blind Forest', 4, 2015, 'Platform', 'XBox, Playstation, PC', 'https://raw.githubusercontent.com/devsuperior/java-spring-dslist/main/resources/8.png', 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Odit esse officiis corrupti unde repellat non quibusdam! Id nihil itaque ipsum!', 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Delectus dolorum illum placeat eligendi, quis maiores veniam. Incidunt dolorum, nisi deleniti dicta odit voluptatem nam provident temporibus reprehenderit blanditiis consectetur tenetur. Dignissimos blanditiis quod corporis iste, aliquid perspiciatis architecto quasi tempore ipsam voluptates ea ad distinctio, sapiente qui, amet quidem culpa.');

INSERT INTO tb\_game (title, score, game\_year, genre, platforms, img\_url, short\_description, long\_description) VALUES ('Cuphead', 4.6, 2017, 'Platform', 'XBox, Playstation, PC', 'https://raw.githubusercontent.com/devsuperior/java-spring-dslist/main/resources/9.png', 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Odit esse officiis corrupti unde repellat non quibusdam! Id nihil itaque ipsum!', 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Delectus dolorum illum placeat eligendi, quis maiores veniam. Incidunt dolorum, nisi deleniti dicta odit voluptatem nam provident temporibus reprehenderit blanditiis consectetur tenetur. Dignissimos blanditiis quod corporis iste, aliquid perspiciatis architecto quasi tempore ipsam voluptates ea ad distinctio, sapiente qui, amet quidem culpa.');

INSERT INTO tb\_game (title, score, game\_year, genre, platforms, img\_url, short\_description, long\_description) VALUES ('Sonic CD', 4, 1993, 'Platform', 'Sega CD, PC', 'https://raw.githubusercontent.com/devsuperior/java-spring-dslist/main/resources/10.png', 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Odit esse officiis corrupti unde repellat non quibusdam! Id nihil itaque ipsum!', 'Lorem ipsum dolor sit amet consectetur adipisicing elit. Delectus dolorum illum placeat eligendi, quis maiores veniam. Incidunt dolorum, nisi deleniti dicta odit voluptatem nam provident temporibus reprehenderit blanditiis consectetur tenetur. Dignissimos blanditiis quod corporis iste, aliquid perspiciatis architecto quasi tempore ipsam voluptates ea ad distinctio, sapiente qui, amet quidem culpa.');

**Concluímos a branch Aula\_01**

**Criação da branch Aula\_02:**

1. Criamos a classe Belonging ( tabela de relacionamento entre as tabelas Game e GameList )
2. Aprendemos que é necessário criar a classe BelongingPK para criar uma única chave quando existe uma tabela de relacionamento.
3. Seed na tabela tb\_game\_list. Adicionar os inserts ao arquivo import.sql na pasta resource.

INSERT INTO tb\_game\_list (name) VALUES ('Aventura e RPG');

INSERT INTO tb\_game\_list (name) VALUES ('Jogos de plataforma');

1. Seed no tabela tb\_belonging. Adicionar os inserts ao arquivo import.sql na pasta resource.

INSERT INTO tb\_belonging (list\_id, game\_id, position) VALUES (1, 1, 0);

INSERT INTO tb\_belonging (list\_id, game\_id, position) VALUES (1, 2, 1);

INSERT INTO tb\_belonging (list\_id, game\_id, position) VALUES (1, 3, 2);

INSERT INTO tb\_belonging (list\_id, game\_id, position) VALUES (1, 4, 3);

INSERT INTO tb\_belonging (list\_id, game\_id, position) VALUES (1, 5, 4);

INSERT INTO tb\_belonging (list\_id, game\_id, position) VALUES (2, 6, 0);

INSERT INTO tb\_belonging (list\_id, game\_id, position) VALUES (2, 7, 1);

INSERT INTO tb\_belonging (list\_id, game\_id, position) VALUES (2, 8, 2);

INSERT INTO tb\_belonging (list\_id, game\_id, position) VALUES (2, 9, 3);

INSERT INTO tb\_belonging (list\_id, game\_id, position) VALUES (2, 10, 4);

1. Fizemos uso do método copyProperties da classe BeanUtils na classe GameDTO. Esse método atribui/copia todos os atributos de uma classe para outra, desde que ambas tenham os mesmos atributos e tipos. Ex. BeanUtils.copyProperties(entityGame, this);

Obs.: Se utilizar este “macete” a classe DTO deverá ter todos os get e set, e não somente os get.

1. No GameService utilizamos a anotação @Transaction(readOnly=true) para garantir que ocorra a transação e o atributo informe ao banco que é uma transação de leitura e não de escrita, tornando o processo mais rápido.
2. Criando consulta SQL nativa:

Query / método adicionado na GameRepository

Sempre que utilizar a query nativa, o resultado da query vai derivar da interface Projection, assim para o nosso projeto criamos a interface GameMinProjection.

@Query(nativeQuery = true, value = """

SELECT tb\_game.id, tb\_game.title, tb\_game.game\_year AS `year`,

tb\_game.img\_url AS imgUrl,

tb\_game.short\_description AS shortDescription, tb\_belonging.position

FROM tb\_game

INNER JOIN tb\_belonging ON tb\_game.id = tb\_belonging.game\_id

WHERE tb\_belonging.list\_id = :listId

ORDER BY tb\_belonging.position

""")

List<GameMinProjection> searchByList(Long listId);

1. Na interface Projection deve ser declarada um método get para cada campo a ser retornado pela query. Ex.: tb\_game.id = Long getId();

tb\_game.title = String getTitle();

1. Criação do endpoint <http://localhost:8080/lists/2/games> ( leitura: traga a lista da categoria 2 dos games )

**Concluímos a branch Aula\_02**