

 devsuperior / sds2

Code

Issues 1

Pull requests


Actions

Projects

Wiki

Security

Insights

 master ▼

...

sds2 / aula1 /



acenelio ...

3 days ago



..



README.md

3 days ago

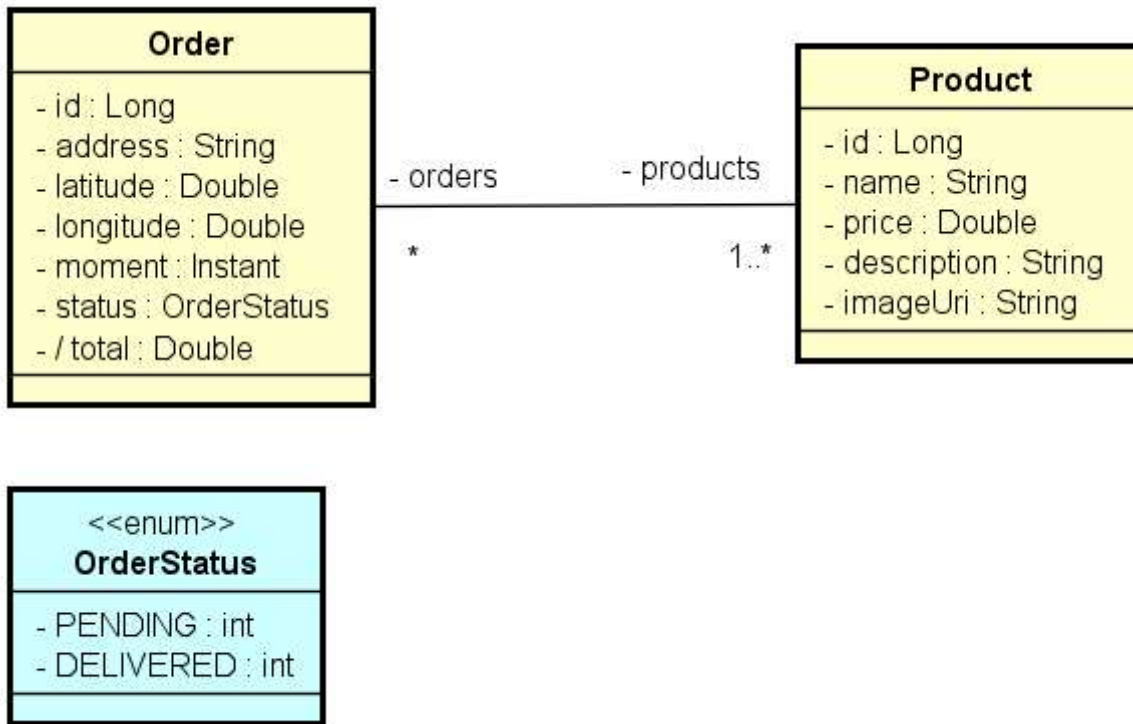
README.md

Aula 1 - Back end

Nivelamento: back end, front end, padrão camadas, MVC, REST

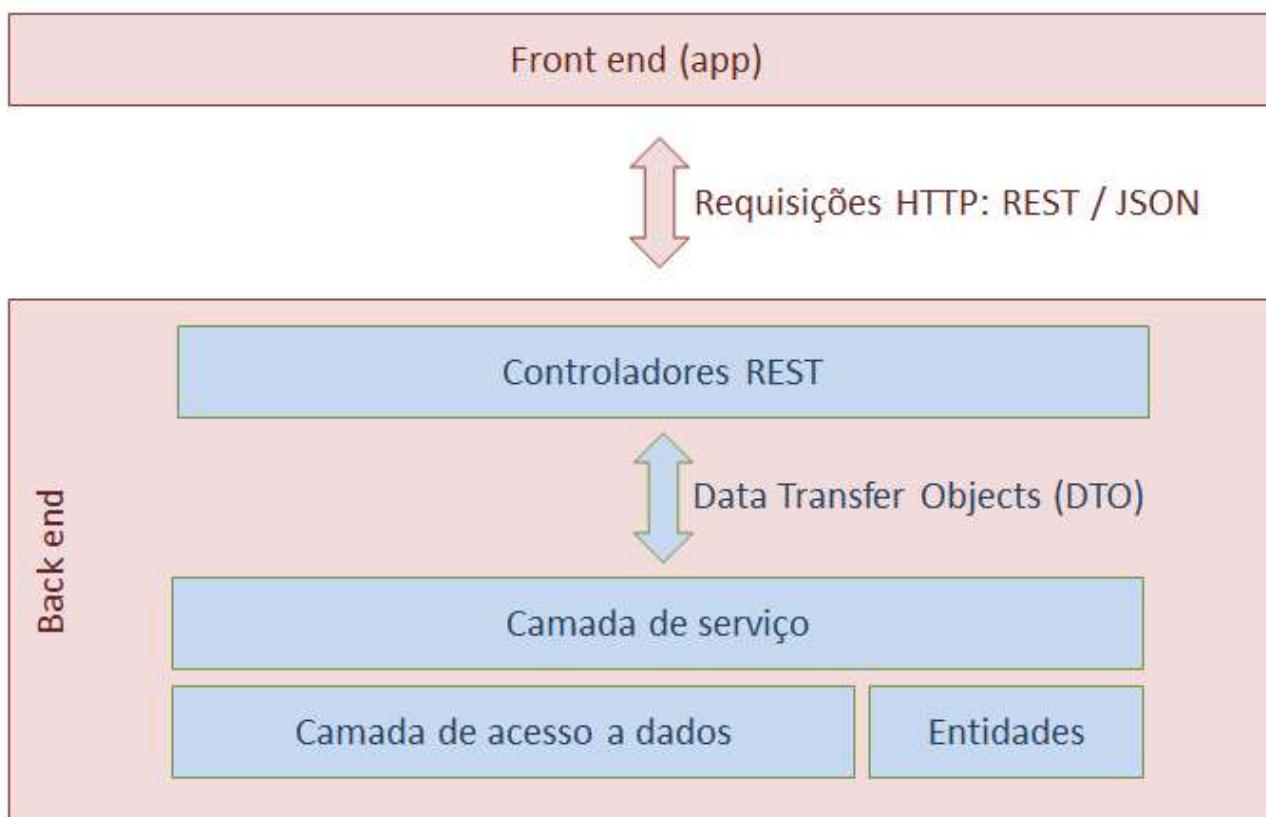


Modelo conceitual



powered by Astah

Padrão camadas adotado



Checklist

- Setup inicial do projeto

- Dependências
- Arquivos .properties
- Configuração de segurança
- Modelo de domínio
 - Entidades e relacionamentos
 - Mapeamento objeto-relacional
 - Seed
- Criar endpoints
 - [GET] /products
 - [GET] /orders
 - [POST] /orders
 - [PUT] /orders/{id}/delivered
- Validar perfil dev
 - Base de dados Postgres local
 - Testar todos endpoints
- Preparar projeto para implantação
 - Arquivo system.properties
 - Profile prod -> commit
- Implantar projeto no Heroku
 - Criar app e provisionar Postgres
 - Criar base de dados remota
 - Executar comandos no Heroku CLI

```
heroku login
heroku git:remote -a <nome-do-app>
git remote -v
git subtree push --prefix backend heroku main
```

Dependências Maven

```
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-web</artifactId>
</dependency>

<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-data-jpa</artifactId>
</dependency>

<dependency>
  <groupId>com.h2database</groupId>
```

```

        <artifactId>h2</artifactId>
        <scope>runtime</scope>
    </dependency>

    <dependency>
        <groupId>org.postgresql</groupId>
        <artifactId>postgresql</artifactId>
        <scope>runtime</scope>
    </dependency>

    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-validation</artifactId>
    </dependency>

    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-security</artifactId>
    </dependency>

    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
    </dependency>

```

Classe de configuração de segurança

```

@Configuration
@EnableWebSecurity
public class SecurityConfig extends WebSecurityConfigurerAdapter {

    @Autowired
    private Environment env;

    @Override
    protected void configure(HttpSecurity http) throws Exception {
        if (Arrays.asList(env.getActiveProfiles()).contains("test")) {
            http.headers().frameOptions().disable();
        }

        http.cors().and().csrf().disable();
        http.sessionManagement().sessionCreationPolicy(SessionCreationPolicy.
            http.authorizeRequests().anyRequest().permitAll();
    }

    @Bean
    CorsConfigurationSource corsConfigurationSource() {
        CorsConfiguration configuration = new CorsConfiguration().applyPermit
            configuration.setAllowedMethods(Arrays.asList("POST", "GET", "PUT", "
            final UrlBasedCorsConfigurationSource source = new UrlBasedCorsConfig
            source.registerCorsConfiguration("/**", configuration);
    }

```

```
        return source;
    }
}
```

Arquivos .properties de cada profile do projeto

application.properties

```
spring.profiles.active=test

spring.jpa.open-in-view=false
```

application-test.properties

```
spring.datasource.url=jdbc:h2:mem:testdb
spring.datasource.username=sa
spring.datasource.password=

spring.h2.console.enabled=true
spring.h2.console.path=/h2-console
```

application-dev.properties

```
#spring.jpa.properties.javax.persistence.schema-generation.create-source=metadata
#spring.jpa.properties.javax.persistence.schema-generation.scripts.action=create
#spring.jpa.properties.javax.persistence.schema-generation.scripts.create-
target=create.sql
#spring.jpa.properties.hibernate.hbm2ddl.delimiter=;

spring.datasource.url=jdbc:postgresql://localhost:5432/dsdeliver
spring.datasource.username=postgres
spring.datasource.password=1234567

spring.jpa.properties.hibernate.jdbc.lob.non_contextual_creation=true
spring.jpa.hibernate.ddl-auto=none
```

application-prod.properties

```
spring.datasource.url=${DATABASE_URL}
```

Script SQL de instanciação da base de dados

```
INSERT INTO tb_product (name, price, image Uri, description) VALUES ('Pizza Bacon', 4
INSERT INTO tb_product (name, price, image Uri, description) VALUES ('Pizza Moda da C
INSERT INTO tb_product (name, price, image Uri, description) VALUES ('Pizza Portugues
INSERT INTO tb_product (name, price, image Uri, description) VALUES ('Risoto de Carne
INSERT INTO tb_product (name, price, image Uri, description) VALUES ('Risoto Funghi',
INSERT INTO tb_product (name, price, image Uri, description) VALUES ('Macarrão Espagu
INSERT INTO tb_product (name, price, image Uri, description) VALUES ('Macarrão Fusili
INSERT INTO tb_product (name, price, image Uri, description) VALUES ('Macarrão Penne'
```

```
INSERT INTO tb_order (status, latitude, longitude, address, moment) VALUES (0, -23.56
INSERT INTO tb_order (status, latitude, longitude, address, moment) VALUES (1, -22.94
INSERT INTO tb_order (status, latitude, longitude, address, moment) VALUES (0, -25.43
INSERT INTO tb_order (status, latitude, longitude, address, moment) VALUES (0, -23.56
INSERT INTO tb_order (status, latitude, longitude, address, moment) VALUES (1, -23.56
INSERT INTO tb_order (status, latitude, longitude, address, moment) VALUES (0, -23.56
INSERT INTO tb_order (status, latitude, longitude, address, moment) VALUES (0, -23.56
```

```
INSERT INTO tb_order_product (order_id, product_id) VALUES (1 , 1);
INSERT INTO tb_order_product (order_id, product_id) VALUES (1 , 4);
INSERT INTO tb_order_product (order_id, product_id) VALUES (2 , 2);
INSERT INTO tb_order_product (order_id, product_id) VALUES (2 , 5);
INSERT INTO tb_order_product (order_id, product_id) VALUES (2 , 8);
INSERT INTO tb_order_product (order_id, product_id) VALUES (3 , 3);
INSERT INTO tb_order_product (order_id, product_id) VALUES (3 , 4);
INSERT INTO tb_order_product (order_id, product_id) VALUES (4 , 2);
INSERT INTO tb_order_product (order_id, product_id) VALUES (4 , 6);
INSERT INTO tb_order_product (order_id, product_id) VALUES (5 , 4);
INSERT INTO tb_order_product (order_id, product_id) VALUES (5 , 6);
INSERT INTO tb_order_product (order_id, product_id) VALUES (6 , 5);
INSERT INTO tb_order_product (order_id, product_id) VALUES (6 , 1);
INSERT INTO tb_order_product (order_id, product_id) VALUES (7 , 7);
INSERT INTO tb_order_product (order_id, product_id) VALUES (7 , 5);
```
