# Projeto SportStock

**RM: 553607**

Introdução

O projeto SportStock é uma aplicação desenvolvida em Java utilizando o framework Spring Boot. O objetivo do projeto é gerenciar produtos esportivos, permitindo operações de CRUD (Create, Read, Update, Delete) sobre os produtos.

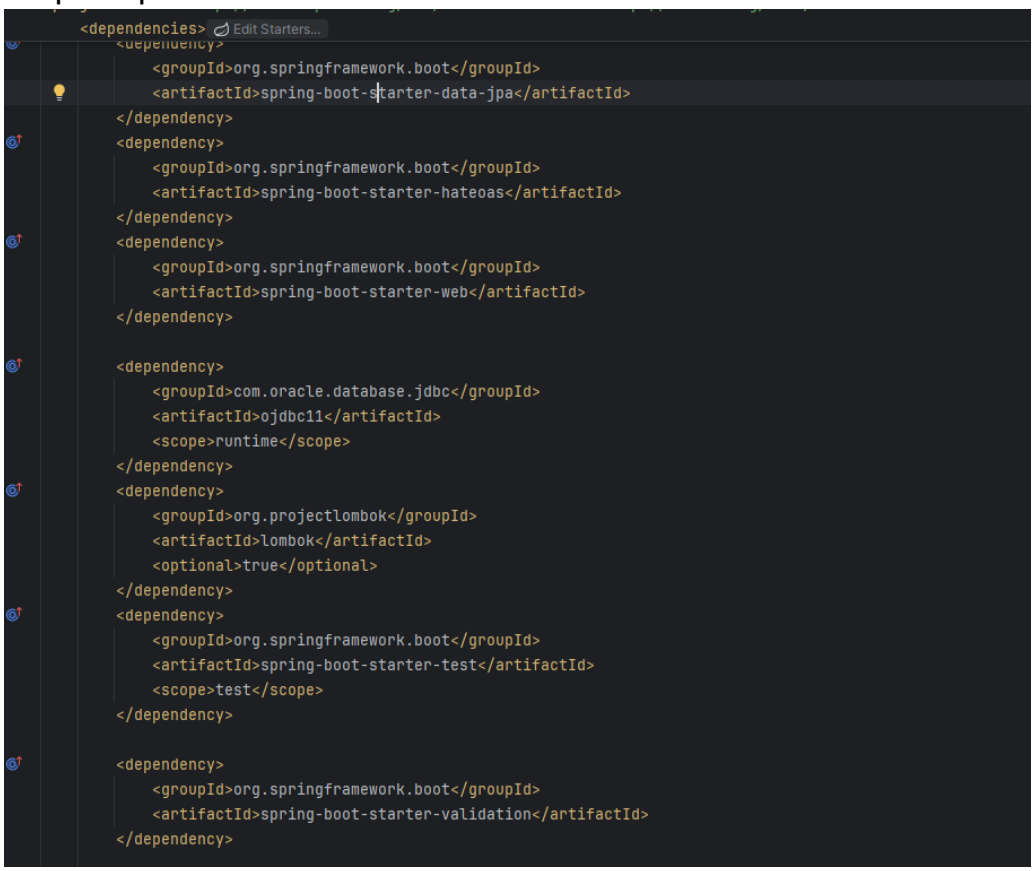
## Tecnologias Utilizadas

- Java
- Spring Boot
- Maven
- H2 Database
- Hibernate
- Lombok

## Configuração do Ambiente

### Dependências

Certifique-se de que as seguintes dependências estão presentes no arquivo pom.xml:



projeto Spring Boot, você pode usar o seguinte exemplo de configuração no arquivo application.yml:

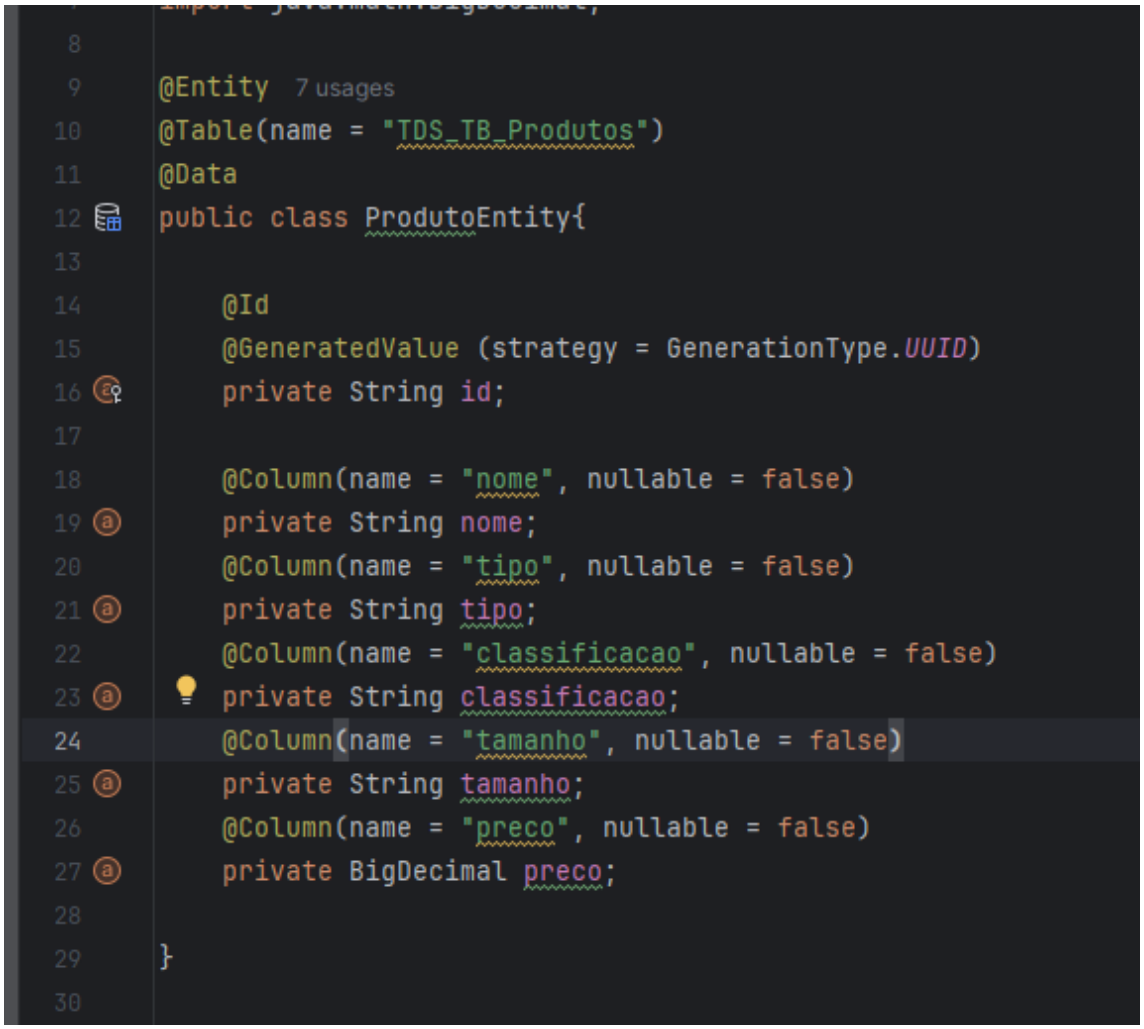
```yaml
spring:
  datasource:
    url: jdbc:oracle:thin:@oracle.fiap.com.br:1521:orcl
    username: "RM553607"
    password: "080694"
    driver-class-name: oracle.jdbc.OracleDriver
  jpa:
    hibernate:
      ddl-auto: create
```

Detalhes das Classes:

```java
@Data 4 usages
public class ProdutoPostRequest {

    @NotBlank(message = "O nome é obrigatório")
    @Size(max = 100, message = "O nome não pode ter mais que 100 caracteres")
    private String nome;

    @NotBlank(message = "O tipo é obrigatório")
    @Size(max = 50, message = "O tipo não pode ter mais que 50 caracteres")
    private String tipo;

    @NotBlank(message = "A classificação é obrigatória")
    @Size(max = 50, message = "A classificação não pode ter mais que 50 caracteres")
    private String classificacao;

    @NotBlank(message = "O tamanho é obrigatório")
    @Size(max = 10, message = "O tamanho não pode ter mais que 10 caracteres")
    private String tamanho;

    @DecimalMin(value = "0.0", inclusive = false, message = "O preço deve ser maior que zero")
    @DecimalMax(value = "999999.99", message = "O preço não pode ser maior que 999999.99")
    private BigDecimal preco;
}
```

ProdutoEntity

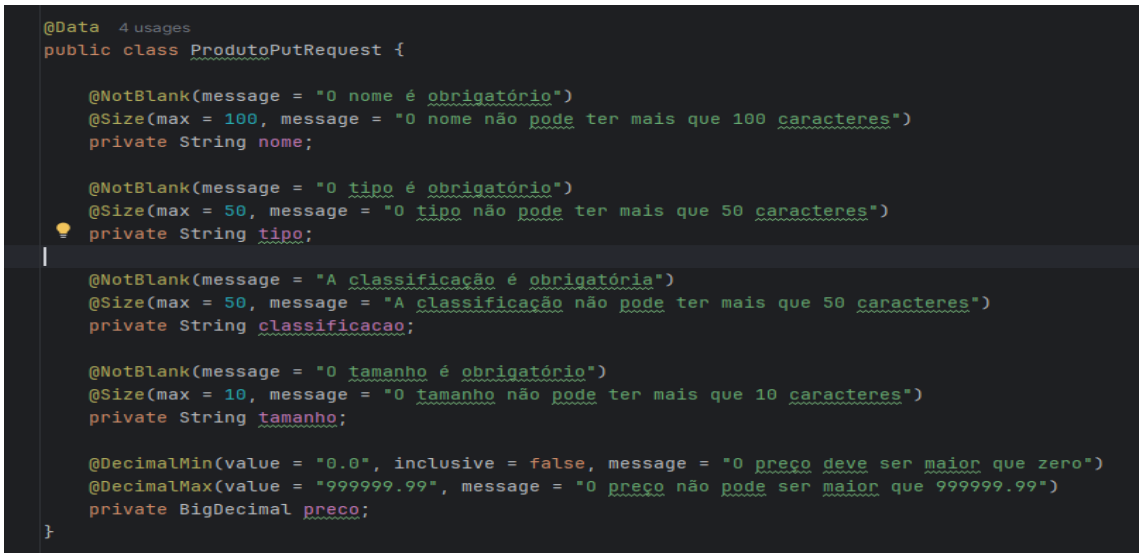
Representa a entidade do produto no banco de dados.



```java
@Entity  7 usages
@Table(name = "TDS_TB_Produtos")
@Data
public class ProdutoEntity{

    @Id
    @GeneratedValue (strategy = GenerationType.UUID)
    private String id;

    @Column(name = "nome", nullable = false)
    private String nome;
    @Column(name = "tipo", nullable = false)
    private String tipo;
    @Column(name = "classificacao", nullable = false)
    private String classificacao;
    @Column(name = "tamanho", nullable = false)
    private String tamanho;
    @Column(name = "preco", nullable = false)
    private BigDecimal preco;

}
```

ProdutoPutRequest

Representa a requisição para atualizar um produto.



```java
@Data  4 usages
public class ProdutoPutRequest {

    @NotBlank(message = "O nome é obrigatório")
    @Size(max = 100, message = "O nome não pode ter mais que 100 caracteres")
    private String nome;

    @NotBlank(message = "O tipo é obrigatório")
    @Size(max = 50, message = "O tipo não pode ter mais que 50 caracteres")
    private String tipo;

    @NotBlank(message = "A classificação é obrigatória")
    @Size(max = 50, message = "A classificação não pode ter mais que 50 caracteres")
    private String classificacao;

    @NotBlank(message = "O tamanho é obrigatório")
    @Size(max = 10, message = "O tamanho não pode ter mais que 10 caracteres")
    private String tamanho;

    @DecimalMin(value = "0.0", inclusive = false, message = "O preço deve ser maior que zero")
    @DecimalMax(value = "999999.99", message = "O preço não pode ser maior que 999999.99")
    private BigDecimal preco;
}
```

Para configurar a conexão com o banco de dados Oracle no seu projeto Spring Boot, você pode usar o seguinte exemplo de configuração no arquivo application.yml: