Desenvolvimento WEB

-

Full Stack Completo: Java + React

Prof. Roberto Alencar

Criando o Projeto: Back-end

Projeto que será desenvolvido

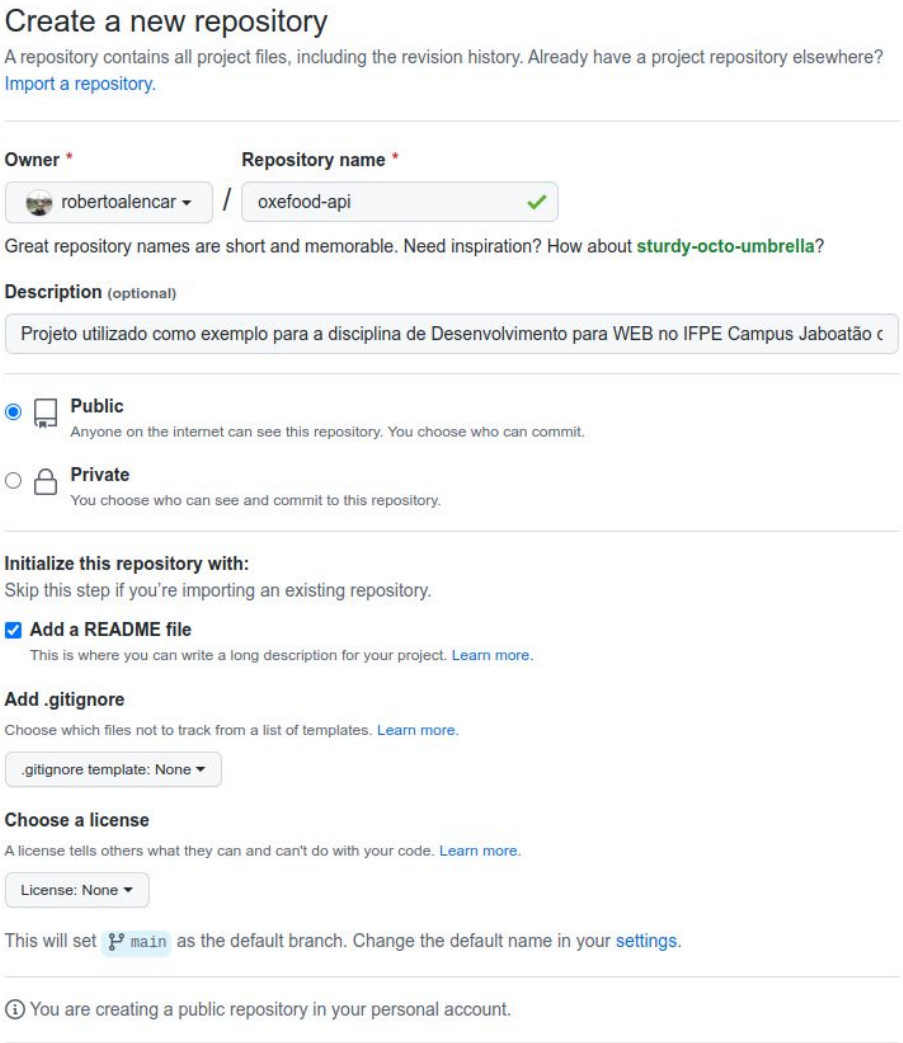
Nossa meta

Iremos desenvolver um sistema web de comércio eletrônico de produtos alimentícios (tipo o iFood)

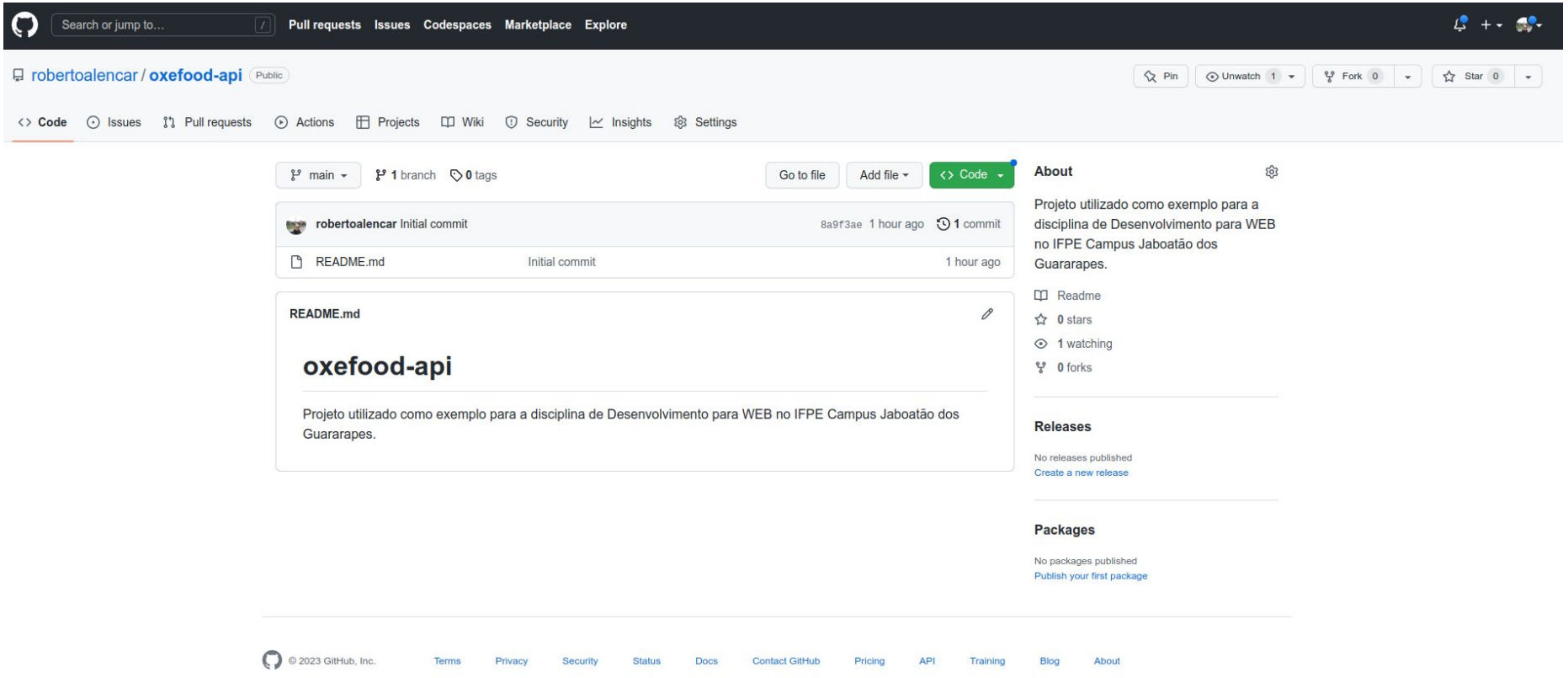
Criando o Projeto

Criando o Projeto

1) Crie um repositório público no seu Github para armazenar o projeto que será trabalhado na disciplina



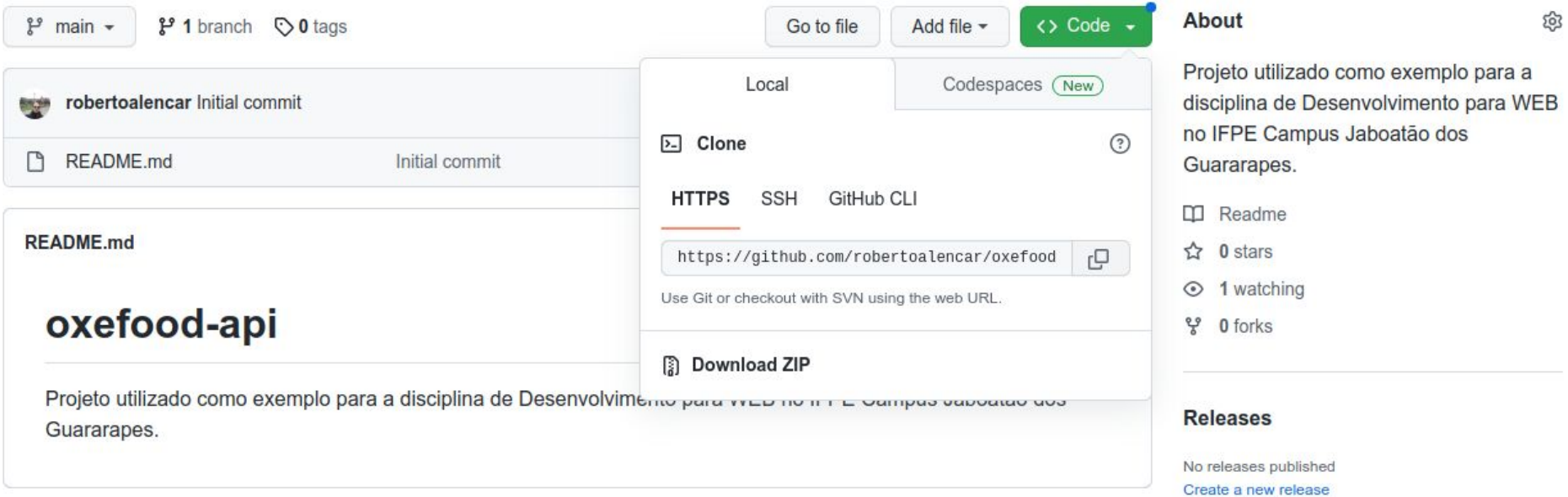
Criando o Projeto



Criando o Projeto

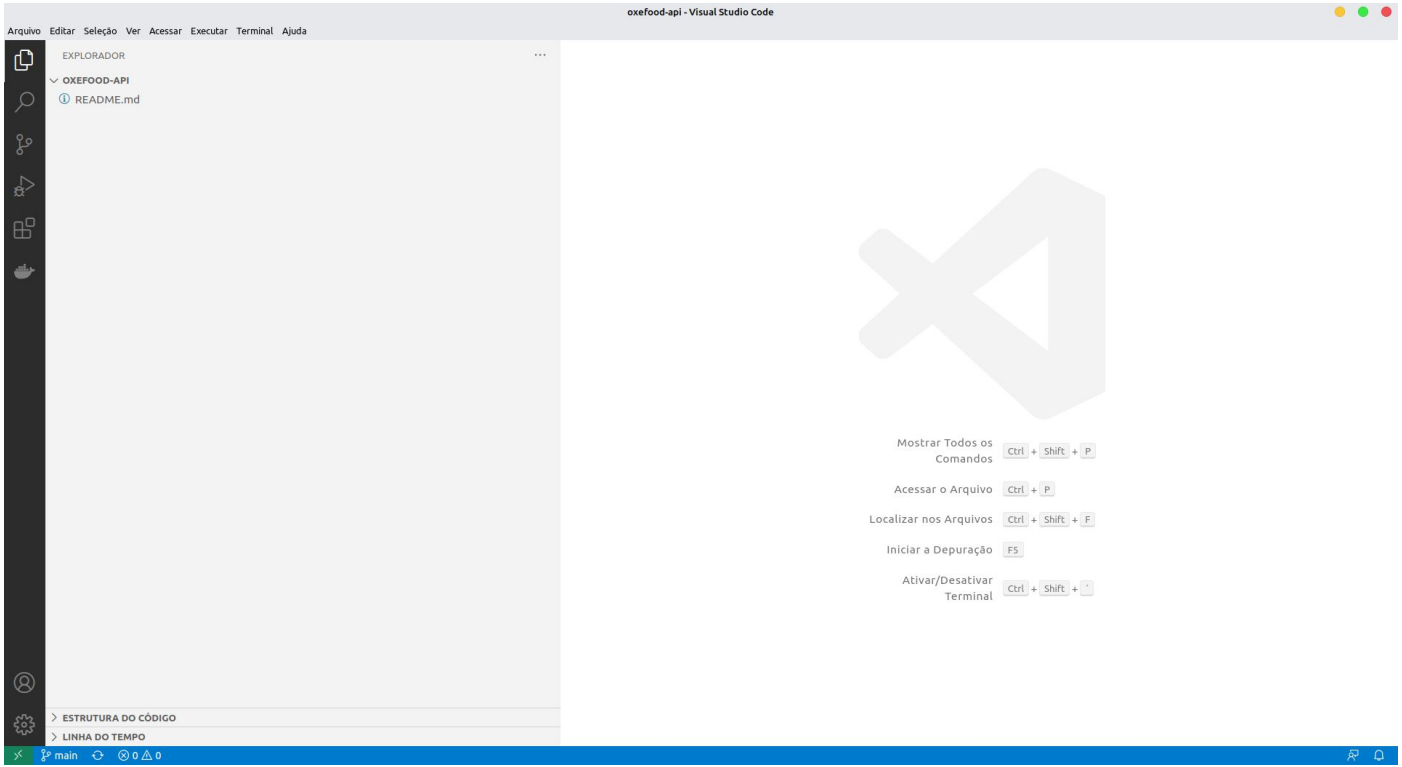
2) Copiando a URL do repositório criado, clone o mesmo para sua máquina local.

- **No meu caso:** `https://github.com/robertoalencar/oxefood-api.git`



Criando o Projeto

3) Abra a pasta do projeto no VS Code



Criando o Projeto

4) Crie o projeto com o Spring Initializr

- O Spring Boot traz a ideia de convenção em vez de configuração, e, para fazer jus a essa frase, a própria plataforma Spring disponibilizou a ferramenta Spring Initializr, que possibilita à pessoa desenvolvedora ter toda a configuração inicial de um projeto Spring com alguns cliques:

    - https://start.spring.io

Criando o Projeto

5) Crie o projeto conforme imagens:

spring initializr

Project

- ○ Gradle - Groovy
- ○ Gradle - Kotlin
- ● Maven

Language

- ● Java
- ○ Kotlin
- ○ Groovy

Spring Boot

- ○ 3.3.0 (SNAPSHOT)
- ○ 3.3.0 (M1)
- ○ 3.2.4 (SNAPSHOT)
- ● 3.2.3
- ○ 3.1.10 (SNAPSHOT)
- ○ 3.1.9

Project Metadata

Group	br.com.ifpe
Artifact	oxefood-api-roberto
Name	oxefood-api-roberto
Description	Demo project for Spring Boot
Package name	br.com.ifpe.oxefood
Packaging	● Jar ○ War
Java	○ 21 ● 17

Dependencies — ADD DEPENDENCIES... CTRL + B

Spring Boot DevTools DEVELOPER TOOLS
Provides fast application restarts, LiveReload, and configurations for enhanced development experience.

Lombok DEVELOPER TOOLS
Java annotation library which helps to reduce boilerplate code.

Spring Web WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Rest Repositories WEB
Exposing Spring Data repositories over REST via Spring Data REST.

Spring HATEOAS WEB
Eases the creation of RESTful APIs that follow the HATEOAS principle when working with Spring / Spring MVC.

Spring Security SECURITY
Highly customizable authentication and access-control framework for Spring applications.

Spring Data JPA SQL
Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

PostgreSQL Driver SQL
A JDBC and R2DBC driver that allows Java programs to connect to a PostgreSQL database using standard, database independent Java code.

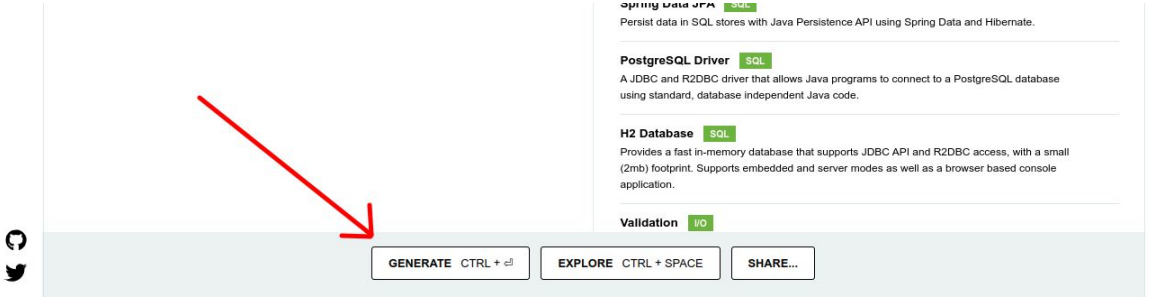
H2 Database SQL
Provides a fast in-memory database that supports JDBC API and R2DBC access, with a small (2mb) footprint. Supports embedded and server modes as well as a browser based console application.

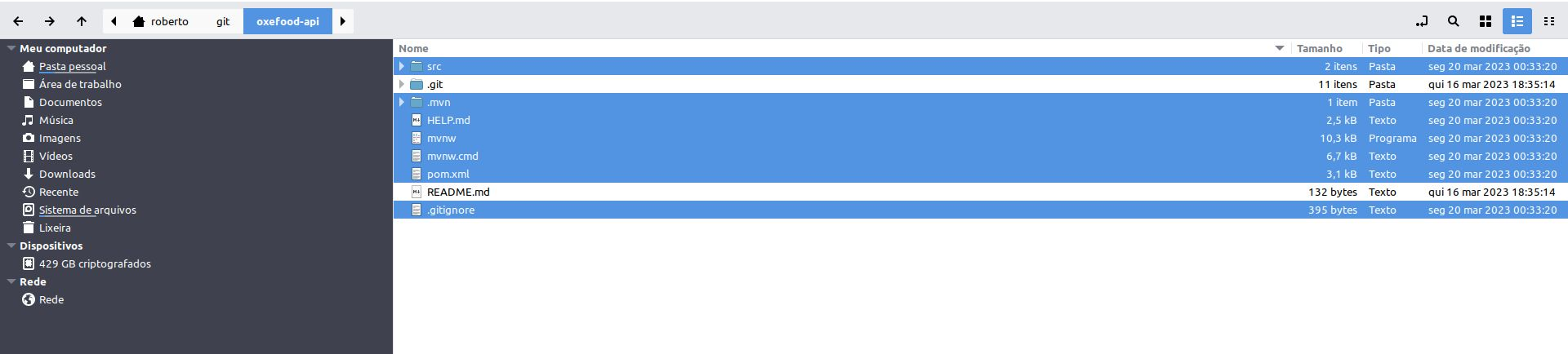
Validation I/O
Bean Validation with Hibernate validator.

Thymeleaf TEMPLATE ENGINES
A modern server-side Java template engine for both web and standalone environments. Allows HTML to be correctly displayed in browsers and as static prototypes.

Criando o Projeto

6) Após baixar o .zip gerado pelo Spring Initializr, descompacte o projeto criado e copie seu conteúdo para o diretório do seu projeto no git.





Criando o Projeto

7) Configuração necessária para rodar o projeto

- Crie um novo arquivo chamado `docker-compose.yml` na raiz do projeto
- Copie e cole o conteúdo abaixo no arquivo `docker-compose.yml`:

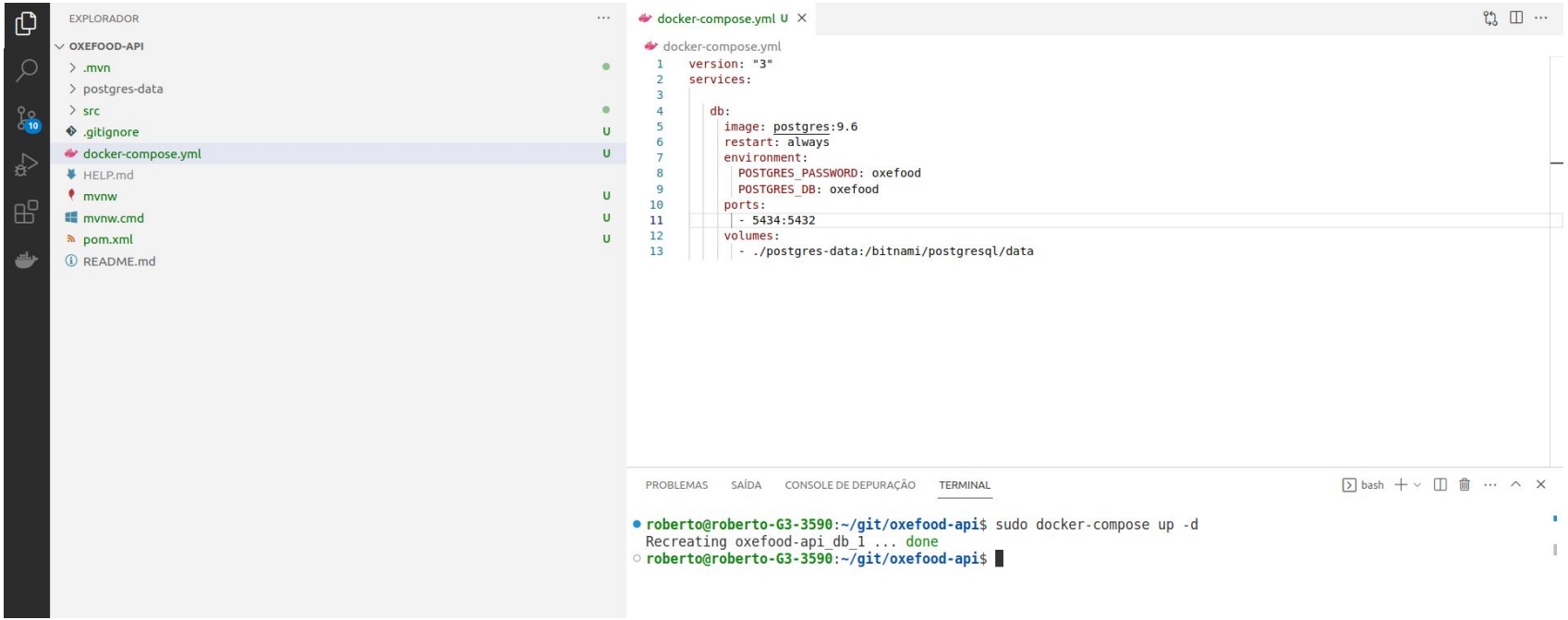
```
version: "3"
services:

   db:
     image: postgres:9.6
     restart: always
     environment:
       POSTGRES_PASSWORD: oxefood
       POSTGRES_DB: oxefood
     ports:
       - 5435:5432
     volumes:
       - ./postgres-data:/bitnami/postgresql/data
```

Criando o Projeto

7) Configuração necessária para rodar o projeto

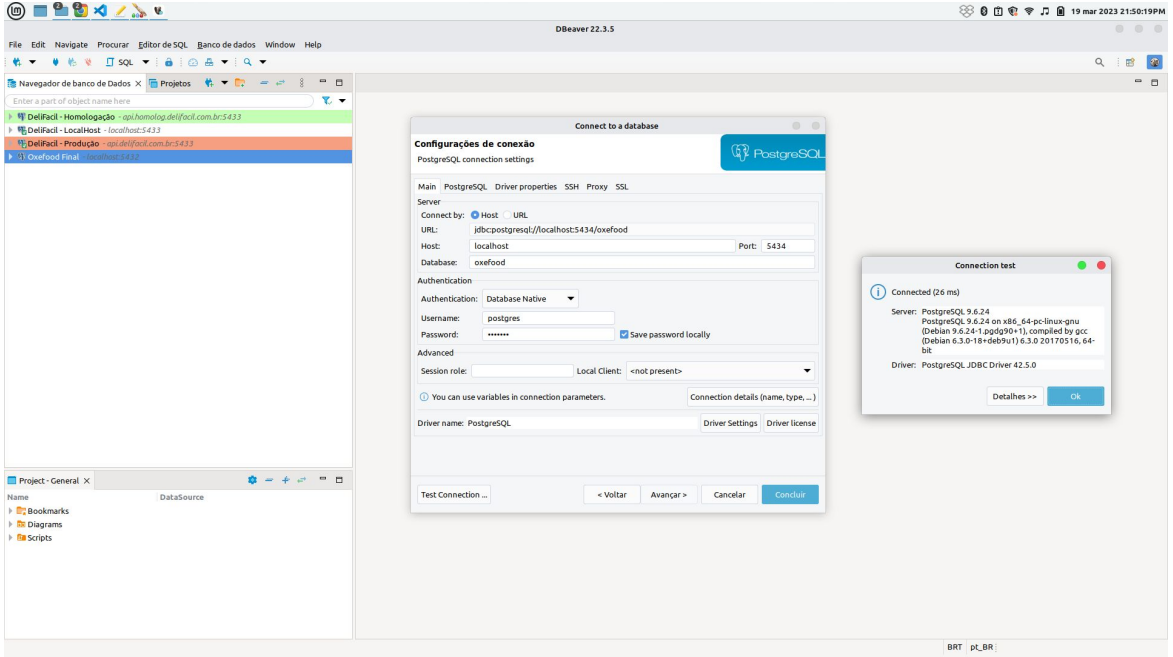
- Levante o container do banco de dados com o comando: `docker-compose up -d`



Criando o Projeto

7) Configuração necessária para rodar o projeto

- Teste se o banco está ok no programa `DBeaver`



Criando o Projeto

8) Configuração necessária para rodar o projeto

- Na pasta `src -> main -> resources`, adicione as configurações abaixo ao arquivo `application.properties` para a conexão com o banco de dados:

```
server.port=8080
spring.config.import=optional:file:.env[.properties]

# Datasource ( levantando a aplicação sem container )
spring.datasource.driver-class-name=org.postgresql.Driver
spring.datasource.url=jdbc:postgresql://localhost:5435/oxefood
spring.datasource.username=postgres
spring.datasource.password=oxefood

# JPA
spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=false
spring.jpa.properties.hibernate.dialect=org.hibernate.dialect.PostgreSQLDialect
spring.jpa.properties.hibernate.temp.use_jdbc_metadata_defaults=false
spring.jpa.database-platform=org.hibernate.dialect.PostgreSQL9Dialect
spring.jpa.generate-ddl=true
```

Criando o Projeto

9.1) Configuração necessária para rodar o projeto

- Abra o arquivo `pom.xml`, e adicione a dependência abaixo (em vermelho):

```
            ...

               <dependency>
                        <groupId>org.apache.commons</groupId>
                        <artifactId>commons-lang3</artifactId>
               </dependency>

</dependencies>
```

Criando o Projeto

9.2) Configuração necessária para rodar o projeto

- Ainda no arquivo `pom.xml`, substitua a marcação `<build>` pelo código abaixo:

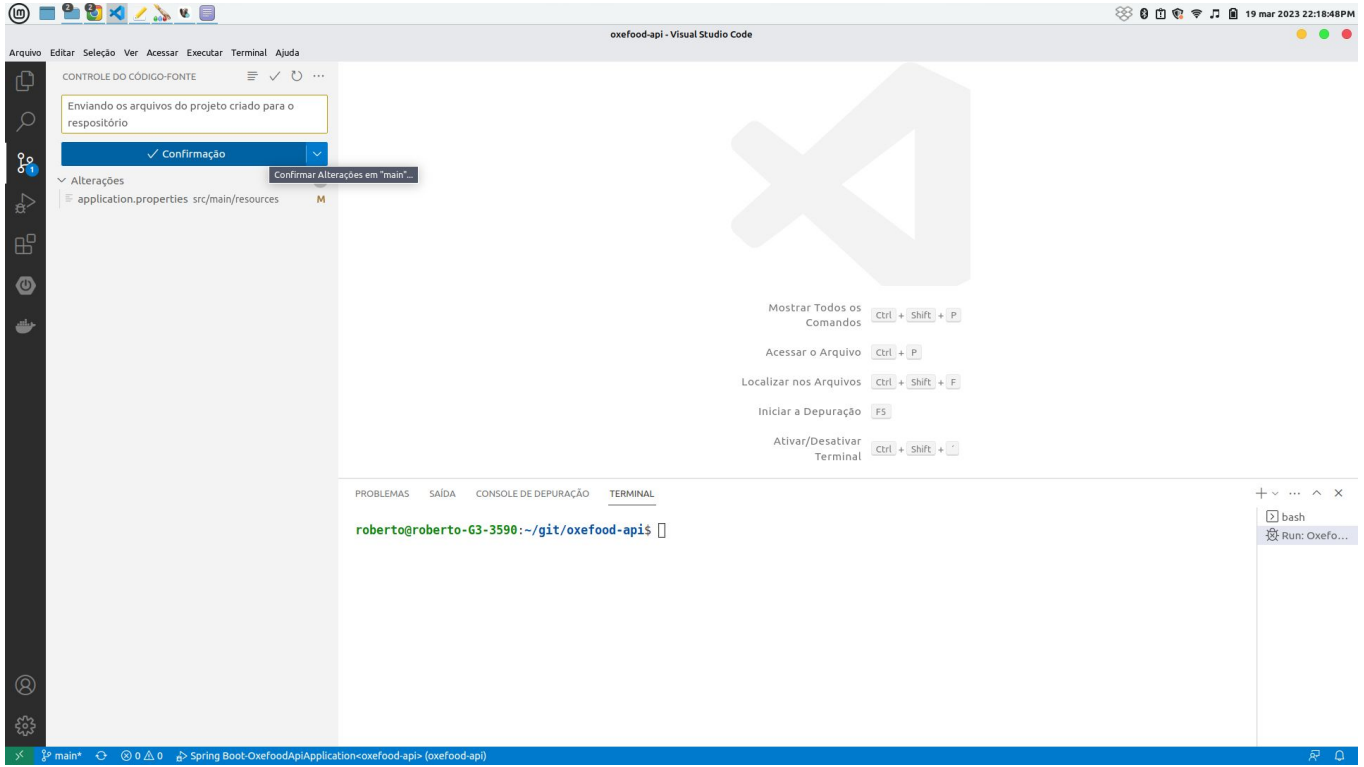
```
<build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot </groupId>
                <artifactId>spring-boot-maven-plugin </artifactId>
                <configuration>
                    <excludes>
                        <exclude>
                            <groupId>org.projectlombok </groupId>
                            <artifactId>lombok</artifactId>
                        </exclude>
                    </excludes>
                </configuration>
            </plugin>
            <plugin>
                <groupId>org.apache.maven.plugins </groupId>
                <artifactId>maven-resources-plugin </artifactId>
                <version>3.1.0</version>
            </plugin>
        </plugins>
    </build>
```

Criando o Projeto

10) Execute seu projeto e verifique se ele está rodando corretamente

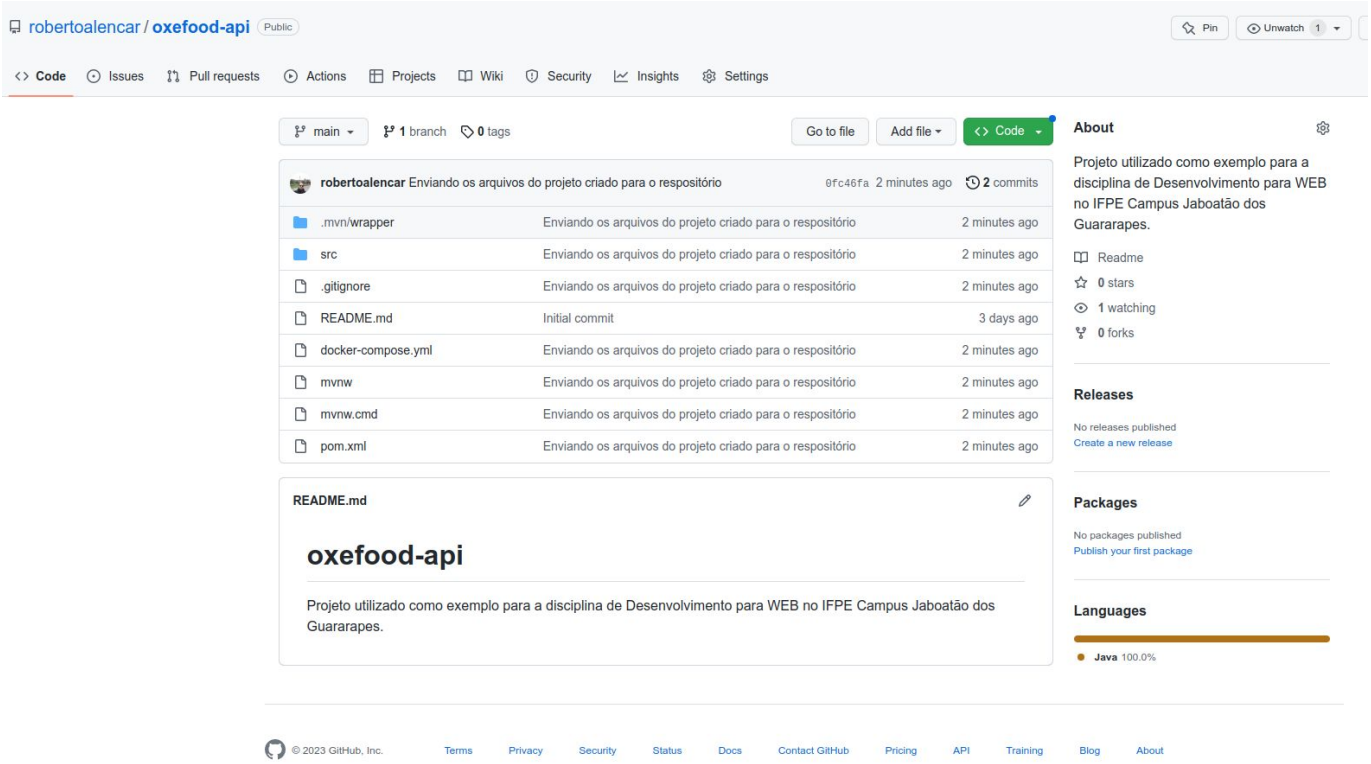
Criando o Projeto

11) Faça o `commit` / `push` dos arquivos para o repositório remoto



Criando o Projeto

11) Faça o `commit` / `push` dos arquivos para o repositório remoto



Dúvidas

Obrigado !