SISTEMA DE RESTAURANTE

Tecnologias

Estão sendo utilizadas as seguintes tecnologias no backend:

1. Java 23
2. Spring Boot 3.5.0-M2
3. MongoDB 8.0.5
4. Junit/Mockito
5. Docker

Estão sendo utilizadas as seguintes tecnologias no frontend:

1. Reactjs 19.0.0
2. Nextjs 15.2.2
3. Tailwind 4.0.14
4. Axios

Criação do projeto frontend

```
mkdir frontend
cd frontend

npm init -y
npm install next react react-dom
```

Editando o arquivo “package.json”

Trocar:

```
"scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
},
```

Por:

```
"scripts": {
  "dev": "next -p 3000",
  "build": "next build",
  "start": "next start"
}
```

Criando a primeira página

Crie a pasta **"pages"** e, dentro dela, o arquivo **"index.js"** que será o script principal. O arquivo **“index.js”** pode ter inicialmente o seguinte conteúdo:

```
const Index = () => (
    <h1>Funcionou!</h1>;
);

export default Index;
```

Criando o arquivo de estilo global

Crie a pasta "**styles**" no mesmo nível da pasta "**pages**" e, dentro dela, crie um arquivo chamado "**globals.css**". È nesse arquivo onde ficarão os estilos **css** puros.

Criando o arquivo “_app.js”

Crie o arquivo "**_app.js**" na raiz da pasta "**pages**" (mesmo nível de "**index.js**") com o seguinte conteúdo:

```
import '../styles/globals.css';

function MyApp({ Component, pageProps }) {
    return (
        <Component {...pageProps} />;
    );
}

export default MyApp;
```

Instalando o tailwind

Execute os seguintes comandos no terminal:

```
npm install tailwindcss @tailwindcss/postcss postcss
```

Crie o arquivo "**postcss.config.mjs**" na raiz do projeto (mesmo nível de "**package.json**") com o seguinte conteúdo:

```
const config = {
  plugins: {
    "@tailwindcss/postcss": {},
  },
};
export default config;
```

Edite o arquivo "**styles/globals.css**" e adicione a seguinte linha no início:

```
@import "tailwindcss";
```

Agora o **tailwind** pode ser utilizado nas páginas da aplicação.

Instalando o axios

Execute o seguinte comando:

```
npm i axios
```

Instalando o jest para testes unitários

Executar os seguintes comandos:

```
npm install --save-dev jest babel-jest @babel/preset-env @babel/preset-react
npm install -D jest jest-environment-jsdom @testing-library/react @testing-
library/dom @testing-library/jest-dom ts-node

npm install @types/jest

npm init jest@latest
```

Criar o arquivo **"jsconfig.json"** com o seguinte conteúdo:

```
{
    "typeAcquisition": {
        "include": [
            "jest"
        ]
    }
}
```

Criar o arquivo **"next.config.js"** com o seguinte conteúdo:

```
module.exports = {
    experimental: {
      forceSwcTransforms: true,
    },
}
```

Criar a pasta **"test"** na raiz do projeto. É nela onde ficam os testes

Rodando em modo de desenvolvimento

Para rodar em modo de desenvolvimento basta executar na raiz do projeto, pelo terminal, o seguinte zcomando:

```
npm run dev
```

E, então, abrir no navegador a página:

```
http://localhost:3000
```

E verá o conteúdo que se inicia na página **"index.js"** da pasta.