Aula 02

1. Preparação do ambiente de desenvolvimento geral
2. Trabalhando com git e github no VSCode
3. Introdução a React e Typescript e Preparação do ambiente de desenvolvimento

---

01. Instalando VSCode e extensão do git e github e fluxo de trabalho

O primeiro passo para começarmos trabalhar é instalar o VSCode. Caso já tenha instalado, podemos prosseguir para o próximo passo. Caso não tenha basta instalar seguindo a documentação clicando aqui (https://code.visualstudio.com/download).

O segundo passo é instalar a CLI do git clicando aqui (https://git-scm.com/download/win).

A seguir vamos instalar a CLI do gh (github) clicando aqui (https://github.com/cli/cli#installation)

Em seguida vamos escolher ou criar uma pasta para trabalhar e iniciar um repositório git nesse diretório:

`git init .`

Após instalar vamos rodar o comando:

`gh repo create teste` ou `gh repo create teste --private`

E vamos criar nosso readme.md e fazer o primeiro commit nesse repositório:

```
echo "# README" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/seu_usuario/teste.git
git push -u origin main
```

---

02. Trabalhando com git e github no VSCode

Vamos criar um arquivo javascript só para fazermos uma vez o fluxo de trabalho com o git e github. Para isso precisamos abrir um espaço de trabalho (workspace) no VSCode:

```
code .
```

E em seguida rodar o comando `mkdir script.js` ou pela interface clicar no botão de criar um novo arquivo.

Vamos abrir o novo arquivo e escrever algum código (ou texto qualquer), somente queremos ter algum conteúdo nesse arquivo. Feito isso, podemos verificar o estado do nosso repositório local git, rodando o comando:

```
git status
```

E para versionar o nosso arquivo rodamos:

`git add script.js` ou para adicionar todos os arquivos unstaged `git add .`

Para commitar a alteração (nesse caso a criação do arquivo contendo determinado conteúdo):

```
git commit -m "mensagem do commit"
```

E para enviar para o repositório remoto (no github):

```
git push
```

Tendo terminado esse processo, teremos feito nosso segundo commit. É dessa forma que devemos continuar trabalhando após fazer novas alterações no código do nosso projeto.

---

03. Introdução a React e Typescript e Preparação do ambiente de desenvolvimento

Antes de qualquer outra coisa, vamos instalar o node e npm. Rode os seguintes comandos de linha de comando no **Windows PowerShell x86**. Para instalar o fnm (Fast Node Manager):

```
winget install Schniz.fnm
```

Configurar o fnm :

```
fnm env --use-on-cd | Out-String | Invoke-Expression
```

Fazer download do Node.js:

```
fnm use --install-if-missing 22
```

Verificar versão do Node.js. Deve retornar "v22.6.0":

`node -v`

Verificar se instalou o npm (Node Package Manager) e qual sua versão. Deve retornar "10.8.2":

`npm -v`

Se preferir siga o tutorial oficial clicando aqui (https://nodejs.org/en/download/package-manager)

Agora, podemos inicializar um projeto Node e instalar as dependências do React e parcel. Podemos rodar no CMD mesmo os comandos:

`npm i react react-dom npm i -D parcel typescript @types/react @types/react-dom`

Agora vamos estruturar nosso projeto:

`mkdir src "" > src/index.html "" > src/index.tsx "" > src/App.tsx "" > src/tsconfig.json`

No arquivo index.html, dentro do `<body>` teremos o seguinte código:

```
<div id="app"></div>
<script type="module" src="index.tsx"></script>
```

No arquivo index.tsx, o seguinte:

```
import { createRoot } from 'react-dom/client';
import { App } from './App';

const root = createRoot(document.getElementById('app') as HTMLElement);

root.render(<App />);
```

E no App.tsx, o código:

```
const App = () => <div>App</div>;

export { App };
```

No arquivo tsconfig.json, adicione o seguinte código para configurar a formatação do tipescrypt:

```
{
  "compilerOptions": {
    "module": "ESNext",
    "target": "ESNext",
    "moduleResolution": "node",
    "lib": [
      "DOM",
      "ESNext"
    ],
    "jsx": "react-jsx"
  }
}
```

No arquivo package.json gerado adicione os seguintes trechos de código:

```
"source": "src/index.html"
```

estado

```
"scripts": {
  "start": "parcel",
  "build": "parcel build"
}
```

Finalmente, para rodar a aplicação no CMD:

```
npm start
```

Caso de erro de pnpm failed to install modules adicione ao package.json o seguinte:

```
"alias": {
  "process": false
}
```