Guia de configuração Projeto Fullstack

Introdução: guia de projeto fullstack seguindo o trabalho do Felipe Rocha, posteriormente pode ser adicionado mais formas para cada componente do projeto. Inicialmente em um projeto fullstack temos os componentes de Frontend e Backend.

Antes de tudo, é preciso criar o repositório do projeto, pois será priorizado o padrão dos commits. A página Conventional Commits, tem algumas explicações e exemplos de como devemos chamaro commit.

Os commits devem ser feitos a cada nova configuração, modificação ou adição de elementos ou componentes no projeto.

Os componentes de frontend nesta aplicação serão:

* Next.js.
* Typescript.
* Shadcn/ui.
* Store de imagens e informações simulando uma API.
* Prettier e Eslint (opcional).

Os componentes de backend:

* Postgres.
* Prisma ORM.
* Neon Postgres, Vercel Postgres ou Supabase Postgres.

Instalação:

Next:

* npx create-next-app@latest <nome\_projeto>
  + Criar o arquivo de seed.ts, que conterá os arquivos de informações, imagens e links que serão usados como API.
  + Os arquivos do Netx.js são server-components, ou seja, apenas seria possível acessar recursos que seriam acessíveis apenas no lado do servidor, portanto hooks e interatividade que o React tem, não funcionaria, então usamos a tag: “client-side” na primeira linha do arquivo page.tsx ou jsx, que será qual centralizará os outros componentes. Agora será renderizado parcialmente no cliente-side também.

Banco de dados:

* npm install prisma –save-dev.
* npx prisma init --datasource-provider postgresql.
* O npx irá criar dois arquivos:
  + .env: Deve colocar a url que foi gerado pela ferramenta serverless ou outra que forneça um banco de dados do Postgres, como neon no caso desta aplicação.
    - Atenção: a parte escrita “?sslmode=require” deve ser retirada, pois é uma parte de autenticação, que não é abordada ainda.
  + Schema.prisma: que conterá os códigos de database.
* Mover o seed.ts para dentro da pasta do Prisma.
* Instalar o ts-node: npm install -D ts-node.
* Adicionar depois de “scripts” a propriedade “prisma”: { “seed”: “ts-node prisma/seed.ts” }.
* Rodar: npx prisma db seed.
* Agora aquelas informações que estavam no arquivo seed, agora estão no banco de dados.

Frontend:

* Instalação e configuração do Tailwind:
  + npm install -D tailwindcss postcss autoprefixer
  + npx tailwindcss init -p.
  + No arquivo tailwind.config.js:

/\*\* @type {import('tailwindcss').Config} \*/

module.exports = {

content: [

"./app/\*\*/\*.{js,ts,jsx,tsx,mdx}",

"./pages/\*\*/\*.{js,ts,jsx,tsx,mdx}",

"./components/\*\*/\*.{js,ts,jsx,tsx,mdx}",

// Or if using `src` directory:

"./src/\*\*/\*.{js,ts,jsx,tsx,mdx}",

],

theme: {

extend: {},

},

plugins: [],

}

* + No arquivo globals.css ou no globalstyle.ts se estiver usando styled-components:

@tailwind base;

@tailwind components;

@tailwind utilities;

Agora pode usar as classes do Tailwindcss.

* Instalação do Shadcn/ui:
  + Npx shadcn-ui@latest init.
  + Por padrão usar cor slater.
  + Mover as pastas criadas “lib” e “components” para dentro da pasta app, e renomeá-las com um underline antes do nome tipo \_lib e \_components.
  + Adicionar componentes:
    - Entra na página do componente e executa o comando de instalação. Irá aparecer na pasta \_components.
* Estudar sobre configuração de variáveis do tailwind e shadcn/ui.
  + Aplicar as variáveis:
    - No arquivo layout.tsx ou jsx adicionar na tag html a className que foi definida as variáveis.