Definindo o Tipo da Aplicação

Aplicação Web SPA (Single Page Application) com back-end RESTful

Características principais:

SPA (Angular): Interface dinâmica, sem recarregamento de páginas, com navegação por rotas.

Back-end (Node.js + Express): API REST que expõe endpoints para login, tarefas, projetos, etc.

Cliente-Servidor separado: Front-end e back-end são hospedados separadamente e se comunicam via HTTP.

Autenticação via JWT: Controle de acesso por token.

Multiusuário: Cada usuário acessa seus próprios projetos/tarefas.

Tipo de Aplicação: Gerenciador de Tarefas Colaborativo Web

Você pode descrever assim no seu projeto (caso precise para iniciação científica ou documentação):

“A plataforma é um sistema web do tipo SPA com arquitetura cliente-servidor, destinado ao gerenciamento de tarefas e projetos com foco em colaboração, rastreamento por data e controle por usuário.”

Arquitetura Geral da Aplicação Web

1. Front-end (Angular)

Tipo: Single Page Application (SPA)

Responsável por:

Interface do usuário (UI)

Navegação entre páginas (com Angular Router)

Comunicação com o back-end via HTTP (HttpClient)

Organização recomendada:

Módulo de Autenticação (auth)

Módulo de Projetos (projects)

Módulo de Tarefas (tasks)

Módulo Compartilhado (shared)

2. Back-end (Node.js + Express)

Tipo: API RESTful

Responsável por:

Lógica de negócio

Autenticação (ex: JWT)

CRUD de usuários, projetos e tarefas

Conexão com banco de dados

3. Banco de Dados

Sugestão:

MongoDB (NoSQL) → flexível, fácil de integrar com Node.js

ou MySQL/PostgreSQL (SQL) → mais estruturado, bom para relacionamentos

4. Comunicação

Angular faz requisições HTTP para a API do Node.js

A API responde com dados em formato JSON

Próximos Passos Práticos

1. Inicializar Projeto Angular

bash

CopiarEditar

ng new protask-frontend

cd protask-frontend

ng serve

2. Inicializar Projeto Node.js

bash

CopiarEditar

mkdir protask-backend

cd protask-backend

npm init -y

npm install express cors mongoose jsonwebtoken bcryptjs dotenv

3. Estrutura de Pastas Sugerida

Front-end (Angular):

rust

CopiarEditar

src/app/

├── auth/ -> login, registro

├── projects/ -> criar e listar projetos

├── tasks/ -> tarefas por projeto

├── shared/ -> componentes reutilizáveis

└── core/ -> serviços e guards

Back-end (Node.js):

rust

CopiarEditar

protask-backend/

├── controllers/ -> lógica dos endpoints

├── models/ -> esquemas (MongoDB) ou modelos (SQL)

├── routes/ -> rotas da API

├── middleware/ -> autenticação JWT, erros

├── config/ -> conexão com banco, variáveis ambiente

└── server.js -> ponto de entrada da aplicação

🔹 4. Configurar Comunicação

Angular usa HttpClient para chamar endpoints da API

Ativar CORS no backend para permitir chamadas entre domínios

Objetivo

Criar uma aplicação web moderna, modular, responsiva e escalável, com:

Interface reativa e agradável (Angular)

Back-end seguro e bem estruturado (Node.js + Express)

Banco de dados conectado (MongoDB ou SQL)

Possibilidade de crescimento futuro (mobile, APIs públicas, dashboard, etc.)