[](https://aluiziodeveloper.com.br/)

**Informação sobre tecnologia, dicas, tutoriais, mini-cursos e muito mais.**

**API Restful com Node.js, Express, Typescript, TypeORM, Postgres, Redis, Docker, ...**

**Estrutura do Projeto**

Estrutura de pastas:

config - configurações de bibliotecas externas, como por exemplo, autenticação, upload, email, etc.

modules - abrangem as áreas de conhecimento da aplicação, diretamente relacionados com as regras de negócios. A princípio criaremos os seguintes módulos na aplicação: customers, products, orders e users.

shared - módulos de uso geral compartilhados com mais de um módulo da aplicação, como por exemplo, o arquivo server.ts, o arquivo principal de rotas, conexão com banco de dados, etc.

services - estarão dentro de cada módulo da aplicação e serão responsáveis por todas as regras que a aplicação precisa atender, como por exemplo:

* A senha deve ser armazenada com criptografia;
* Não pode haver mais de um produto com o mesmo nome;
* Não pode haver um mesmo email sendo usado por mais de um usuário;
* E muitas outras...

Criando a estrutura de pastas:

mkdir -p src/config

mkdir -p src/modules

mkdir -p src/shared/http

mv src/server.ts src/shared/http/server.ts

Ajustar o arquivo package.json:

{

"scripts": {

"dev": "ts-node-dev --inspect --transpile-only --ignore-watch node\_modules src/shared/http/server.ts"

}

}

**Configurando as importações**

Podemos usar um recurso que facilitará o processo de importação de arquivos em nosso projeto.

Iniciamos configurando o objeto paths do tsconfig.json, que permite criar uma base para cada path a ser buscado no projeto, funcionando de forma similar a um atalho:

"paths": {

"@config/\*": ["src/config/\*"],

"@modules/\*": ["src/modules/\*"],

"@shared/\*": ["src/shared/\*"]

}

Nesta videoaula ficou faltando instalar a biblioteca que irá indicar ao nosso script do ts-node-dev, como interpretar os atalhos que configuramos iniciando com o caracter @.

O nome dessa biblioteca é tsconfig-paths, e para instalar execute o seguinte comando no terminal (na pasta do projeto):

yarn add -D tsconfig-paths

Depois de instalar o tsconfig-paths, ajustar o script dev no arquivo package.json, incluindo a opção -r tsconfig-paths/register. Deverá ficar assim:

"dev": "ts-node-dev -r tsconfig-paths/register --inspect --transpile-only --ignore-watch node\_modules src/shared/http/server.ts"

Observação: o comando acima deve ser incluído como uma linha única do script dev.

Agora, para importar qualquer arquivo no projeto, inicie o caminho com um dos paths configurados, usando o CTRL+SPACE para usar o autocomplete.