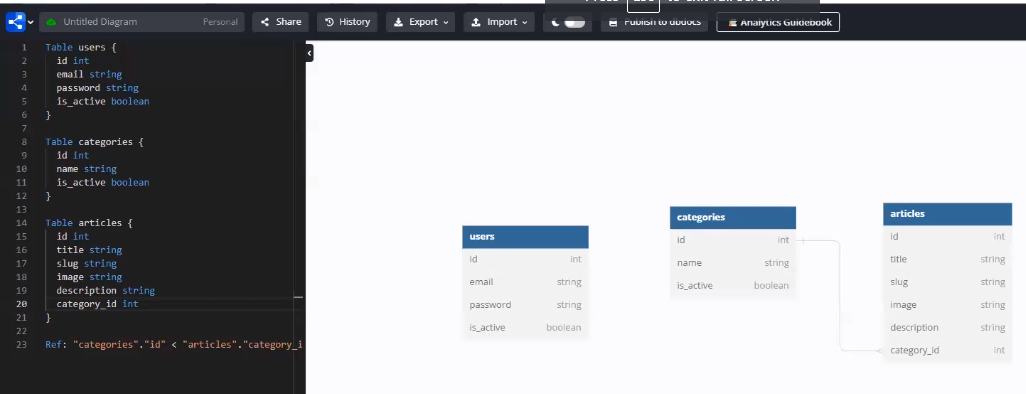
**DOKUMENTASI CHALLENGE CHAPTER 6**

1. Membuat DBDesigner



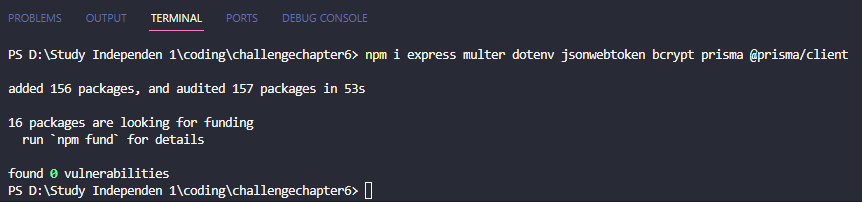
1. Membuat file **index.js**
2. Lakukan init - Perintah berikut untuk membuat file package.json

**npm init**

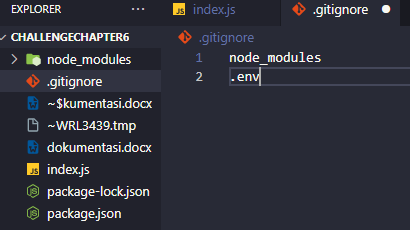


1. Lakukan instalisasi express, multer, dotenv, jsonwebtoken, bcrypt, prisma dan prisma client

**npm i express multer dotenv jsonwebtoken bcrypt prisma @prisma/cient**

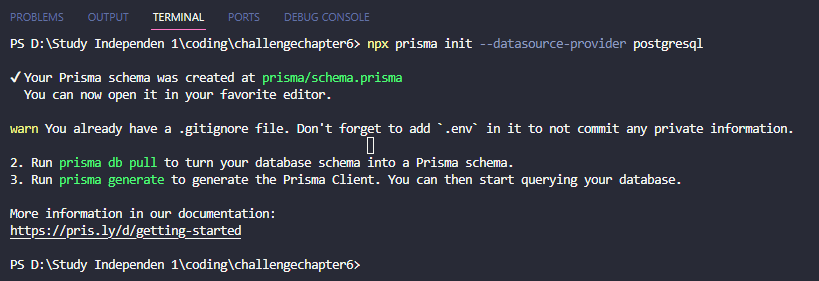


1. Buat file **.env** dan file **.env.example**
2. Buat file **.gitignore** agar node modules dan file **.env** nya tidak ter-push



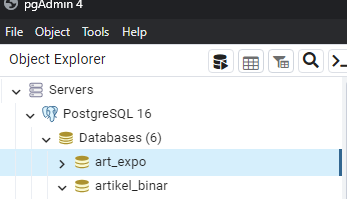
1. Eksekusi prisma atau konfigurasi Prisma dengan PostgreSQL

**npx prisma init --datasource-provider postgresql**



1. Membuat database = dengan nama atikel\_binar di PostgreSQL

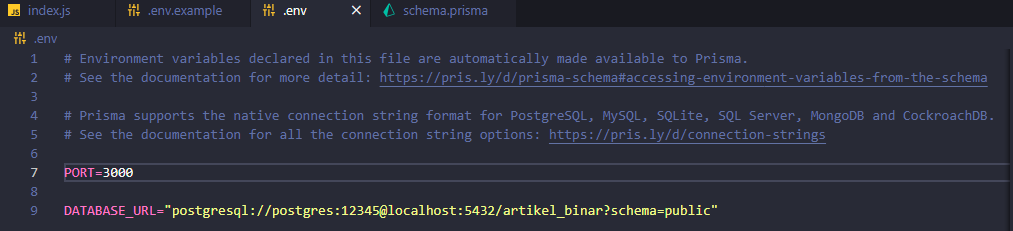
Dengan klik kanan pada Database dan Create



1. Akan muncul di file .env yaitu

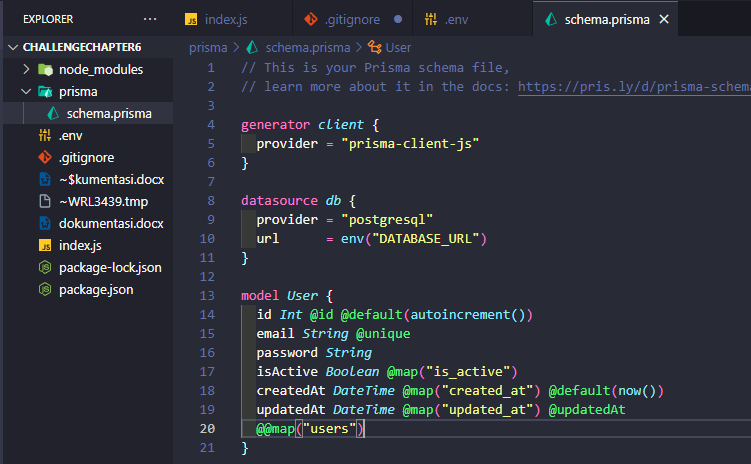
**DATABASE\_URL = "postgresql://postgres:12345@localhost:5432/artikel\_binar?schema=public"**

Ubah password menjadi **12345** dan nama database menjadi **artikel\_binar**

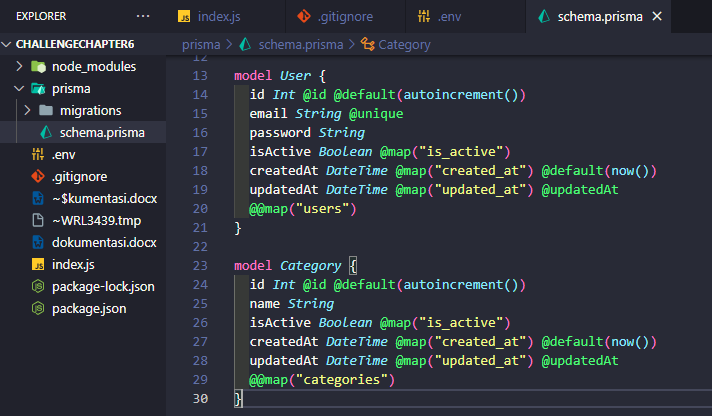
****

1. Isikan di folder **prisma** di file **schema.prisma**

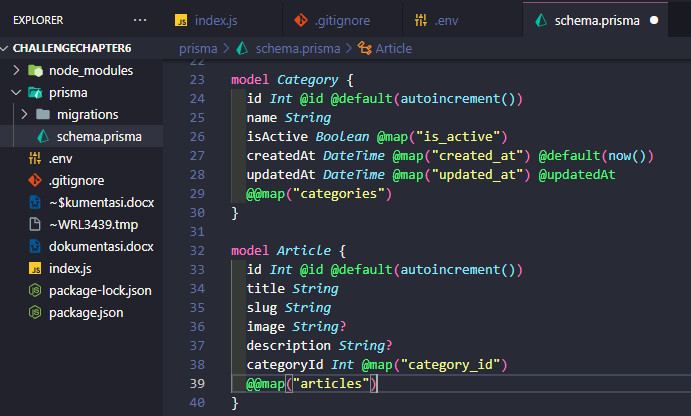
* Table User

****

* Table Category



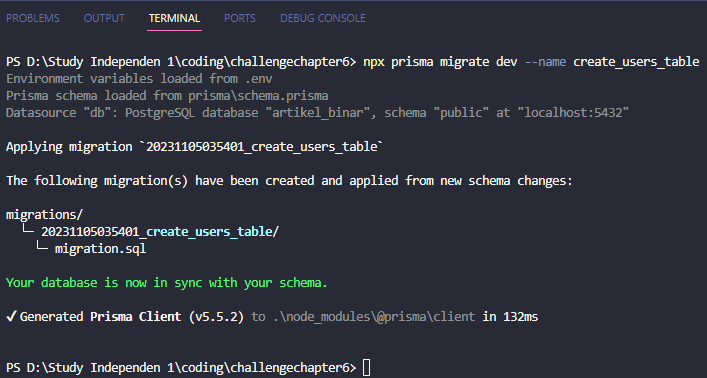
* Table Article



1. Best practise dan agar rapi dalam men-generate prisma, harus men-generate satu persatu setiap membuat tabel.

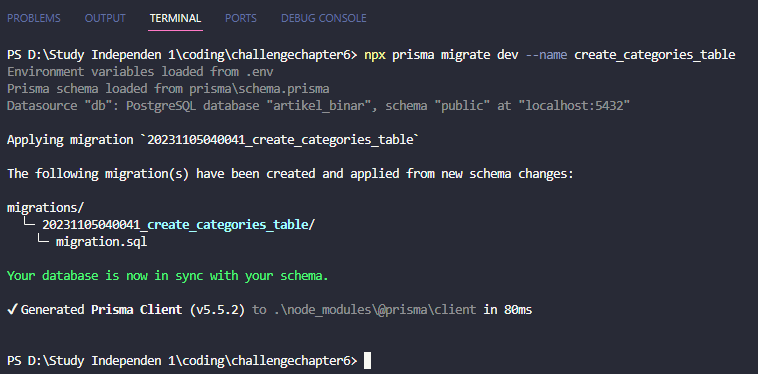
* Table User

**npx prisma migrate dev --name create\_users\_table**



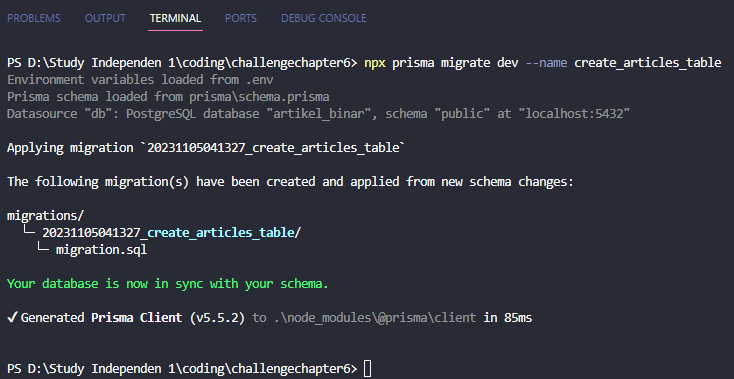
* Table Category

**npx prisma migrate dev --name create\_categories\_table**



* Table Articles

**npx prisma migrate dev --name create\_articles\_table**

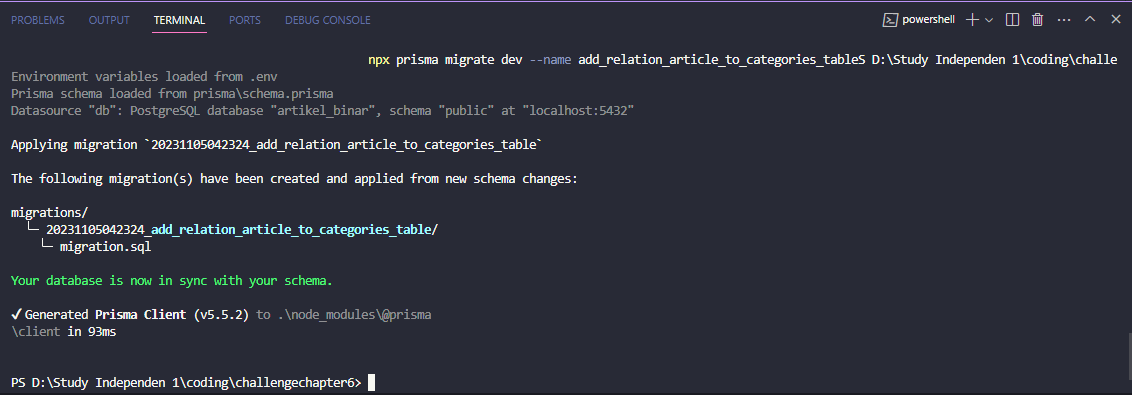


1. Membuat migrationnya

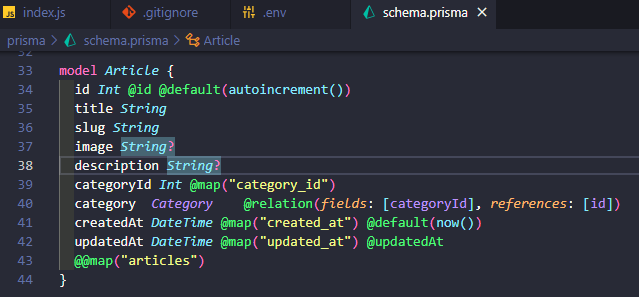
Tambahkan 2 baris dari bawah untuk melakukan relation nya.



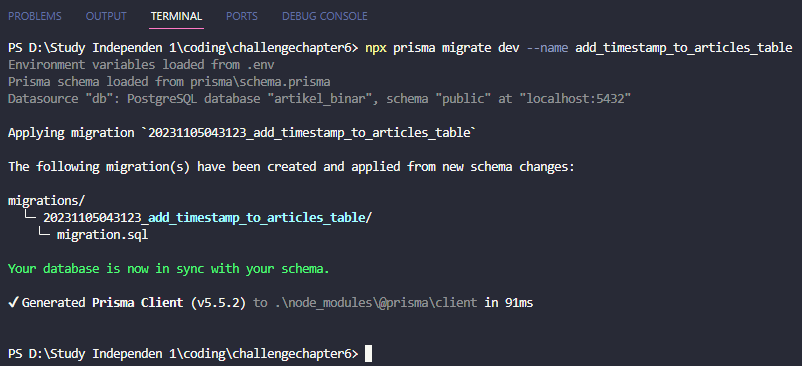
**npx prisma migrate dev --name add\_relation\_article\_to\_categories\_table**



1. Karena ada query tambahan di table Article, maka lakukan migrate kembali

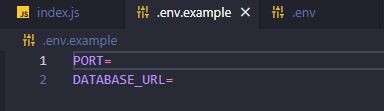


**npx prisma migrate dev --name add\_timestamp\_to\_articles\_table**



1. Isikan di file **.env.example**

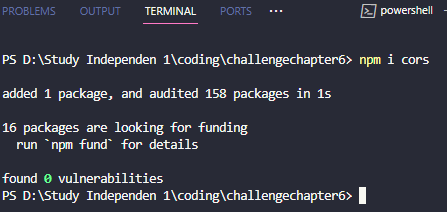
File ini digunakan agar developm lain tau apa aja yang harus ada di file **.env**

****

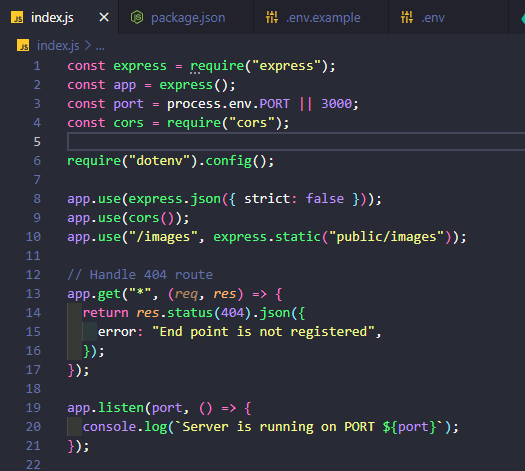
1. Install cors

**CORS** adalah paket node.js untuk menyediakan middleware Connect / Express yang dapat digunakan untuk mengaktifkan CORS dengan berbagai opsi.

**npm i cors**

****

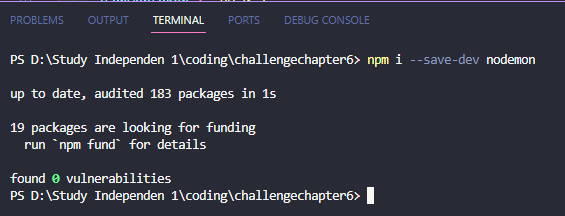
1. Isikan di file utama **index.js**



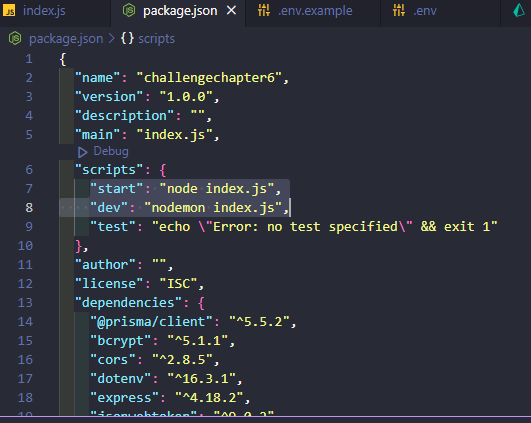
1. Install nodemon

**nodemon** = Jika ada perubahan dalam kode sumber, nodemon akan menghentikan server dan memulai ulangnya secara otomatis

**npm i --save-dev nodemon**

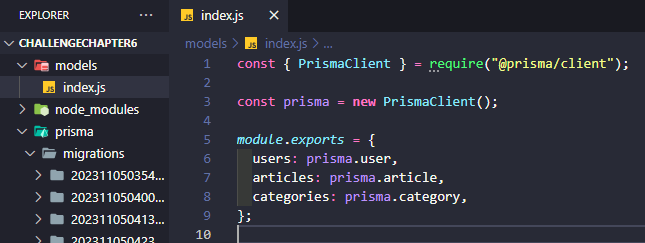


1. Tambahkan pada bagian file **package.json** untuk mengatur start dan dev nya

****

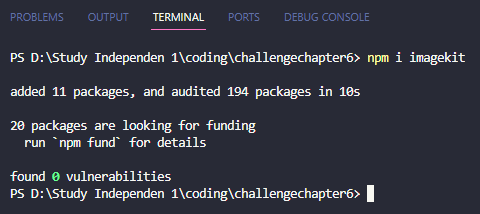
1. Tambahkan folder **models**

Di dalam folder ditambahkan file index.js

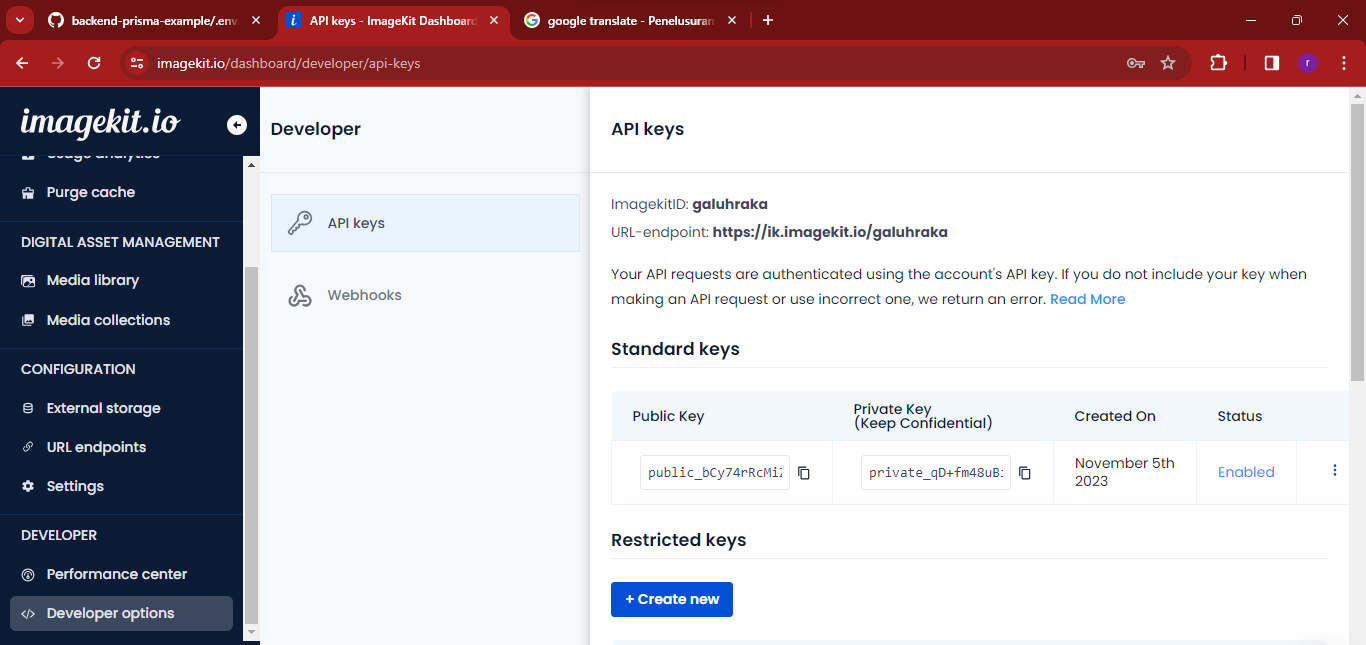


1. Install imageKit

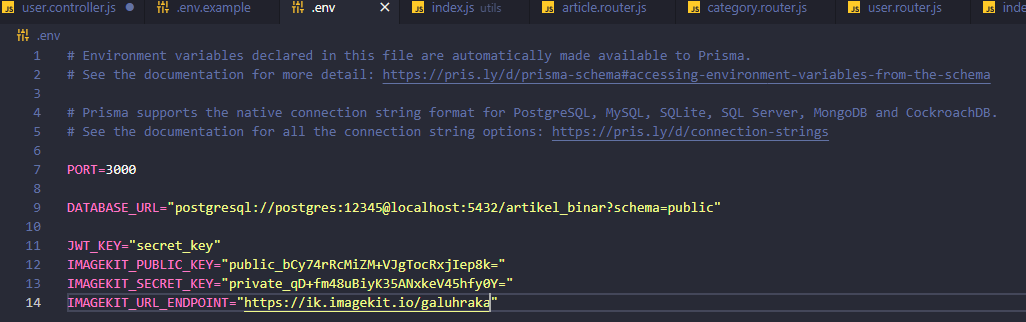
**npm i imagekit**



1. Buka ImageKit pada **setting -> Developer Option**

****

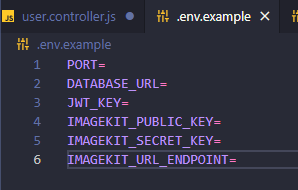
1. Tambahkan dibagian file .**env**



Di dapat dari ImageKit.io

**setting -> Developer Option**

1. Tambahkan dibagian file .**env.example**



1. Tambahkan folder **utils**

Di dalam folder tersebut tambahkan file **index.js**

1. Tambahkan folder **controllers**

Di dalam folder tersebut terdapat 3 file

* user.controller.js
* category.controller.js
* article.controller.js

1. Tambahkan folder **middlewares**

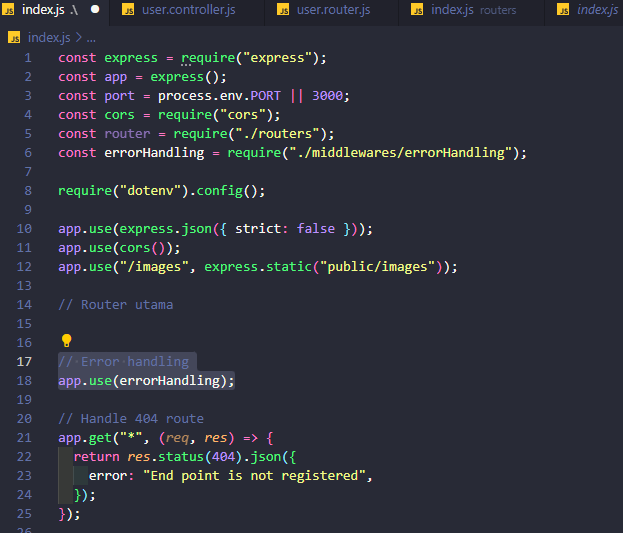
Di dalam folder tersebut tambahkan file **errorHandling.js**

Tujuannya untuk menghandling error kita



Gunakan function next untuk disemua controller

1. ErrorHandling harus di bawah router utama kita, maka edit tambahkan dibagian file utama **index.js**

****

1. Tambahkan folder **routers**

Di dalam folder tersebut terdapat 4 file

* user.router.js
* category. router.js
* article. router.js
* index.js