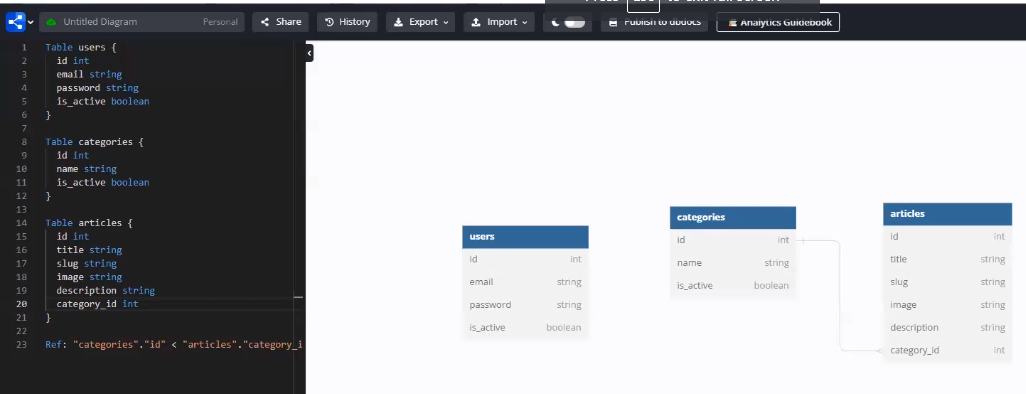
**DOKUMENTASI CHALLENGE CHAPTER 6**

1. Membuat DBDesigner



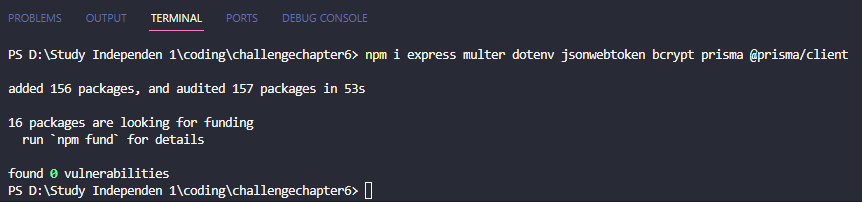
1. Membuat file **index.js**
2. Lakukan init - Perintah berikut untuk membuat file package.json

**npm init**



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

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



* **npm i express =** digunakan untuk membangun aplikasi web di Node.js. Untuk

menginstalExpress.js.

* **dotenv** **=** Instal dotenv: dotenv adalah modul tanpa ketergantungan yang

memuatvariabel lingkungan dari file .env ke process.env.

* **bcrypt =** Ini adalah library hash kriptografis untuk Node.js yang menawarkan

perlindungan terhadap password yang terenkripsi.

* **Jsonwebtoken =** JWT (JSON Web Token) adalah standar format token yang digunakan

untuk melakukan otentikasi menggunakan JSON. JWT digunakan untuk

melakukan otentikasi antara aplikasi, seperti pada Single Sign-On (SSO).

Dalam npm jsonwebtoken, kita akan mendapatkan akses untuk

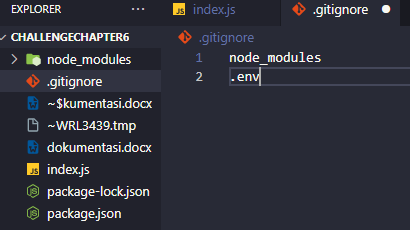
menghasilkan dan mengonfirmasi JWT.

* **@prisma/cient =** Untuk koneksikan database clien. Dapat menggunakan perintah import atau

require('@prisma/client') untuk mengimpor dan menggunakan fungsi dari

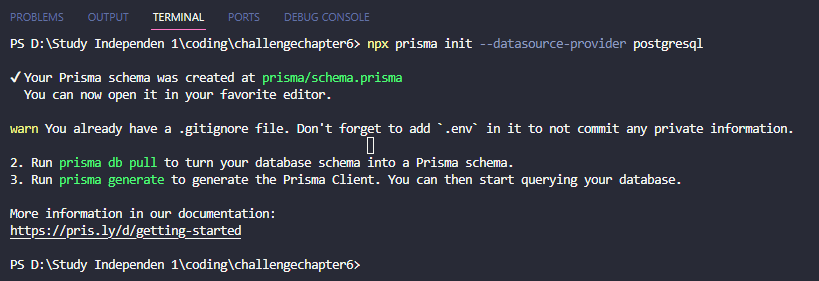
package ini pada kode program Anda

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

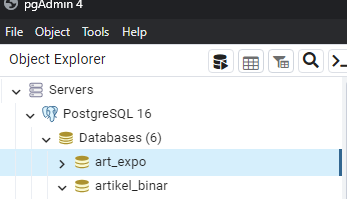
**npx prisma init --datasource-provider postgresql**



* Jika Anda menggunakan npm, gunakan perintah **npx prisma init**. Perintah ini akan membuat berkas konfigurasi Prisma (berkas **schema.prisma**) dan folder prisma di direktori proyek Anda.
* Argumen **--datasource-provider postgresql** memberitahu Prisma bahwa sumber data yang digunakan adalah PostgreSQL. Ini akan menciptakan konfigurasi yang sesuai di berkas **schema.prisma**.

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

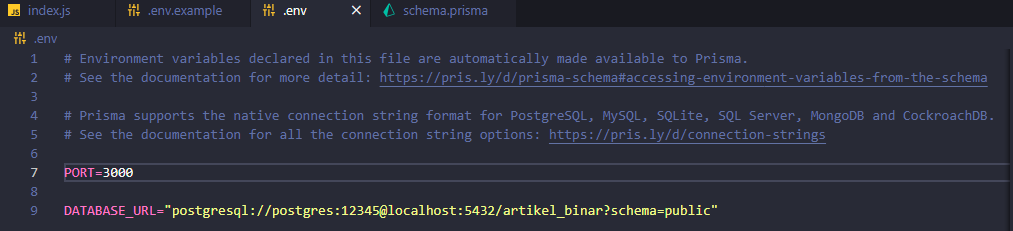
Dengan klik kanan pada Database dan Create



1. Akan muncul di file .env yaitu

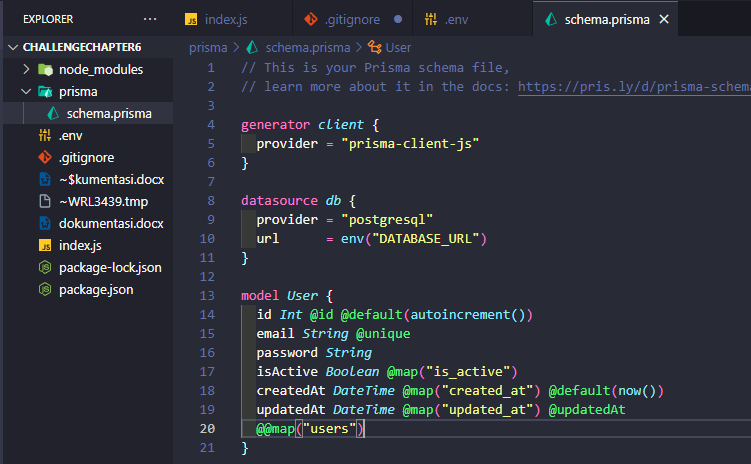
**DATABASE\_URL = "postgresql://postgres:12345@localhost:5432/artikel\_binar?schema=public"**

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

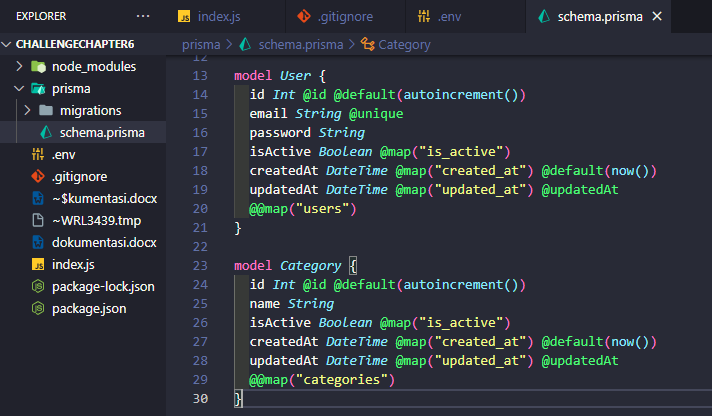
****

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

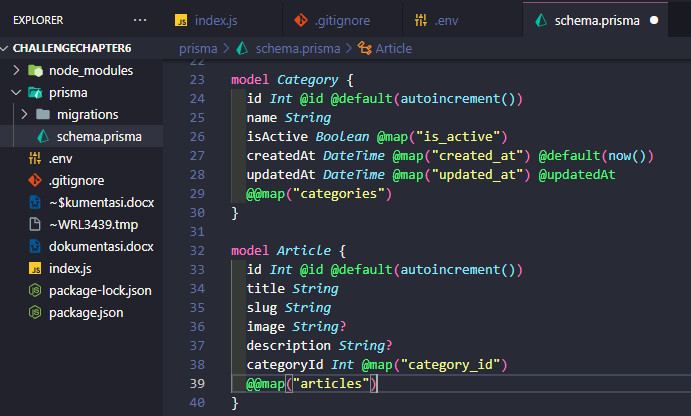
* Table User

****

* Table Category



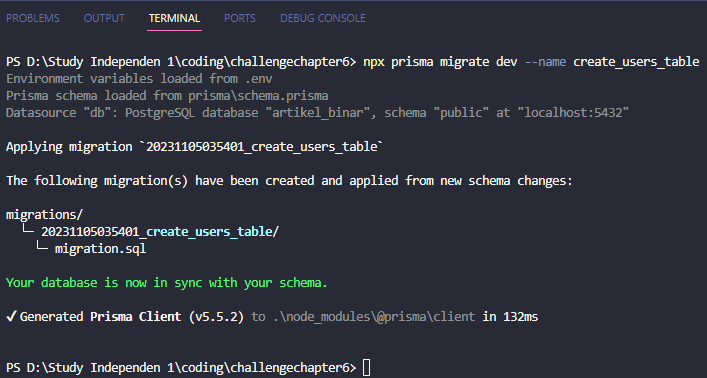
* Table Article



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

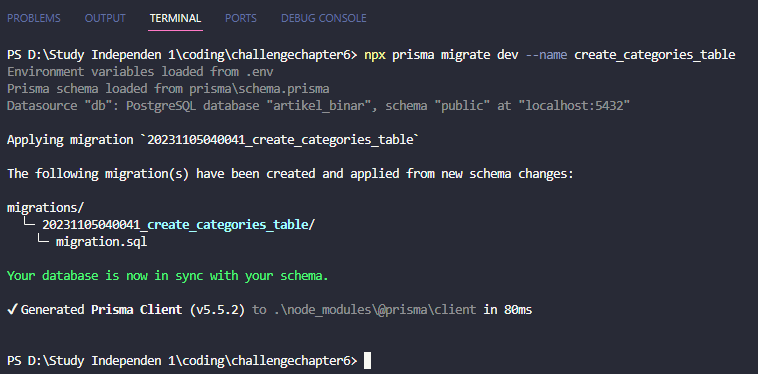
* Table User

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



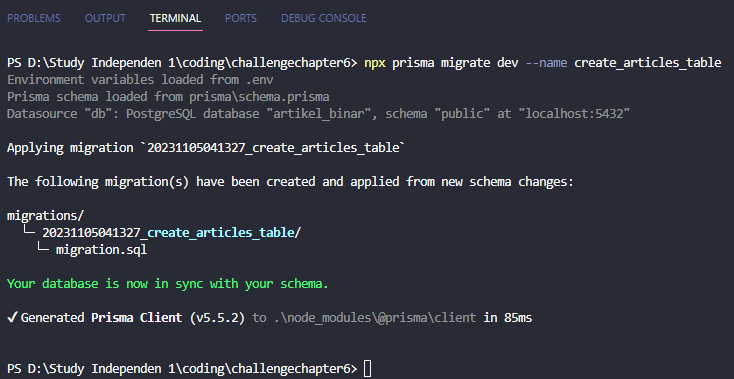
* Table Category

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



* Table Articles

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

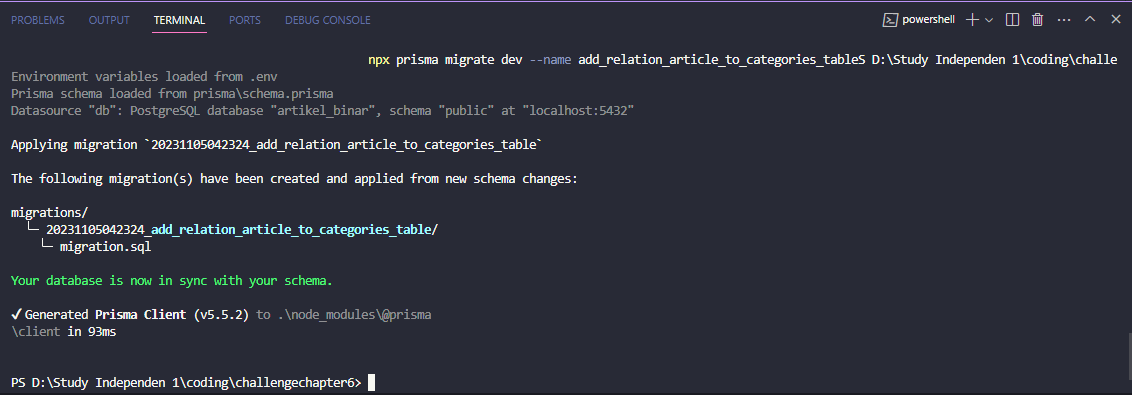


1. Membuat migrationnya

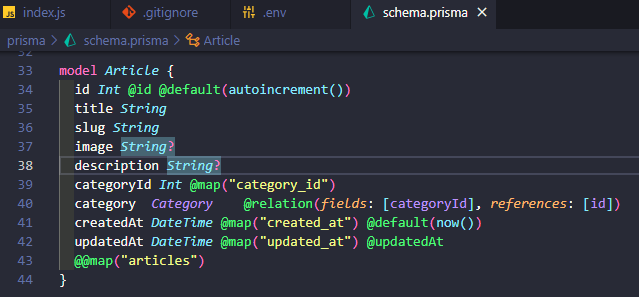
Tambahkan 2 baris dari bawah untuk melakukan relation nya.



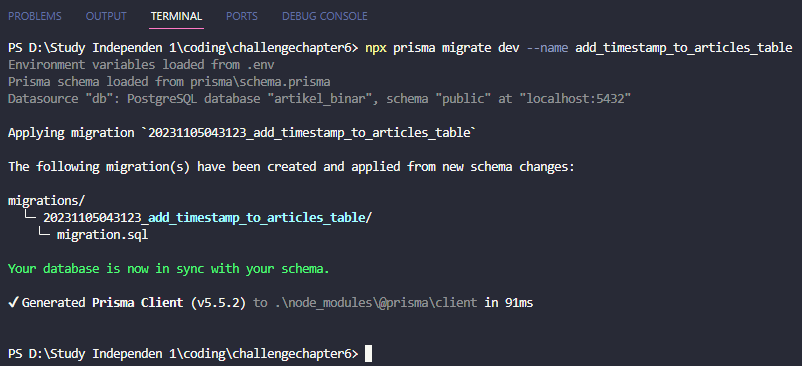
**npx prisma migrate dev --name add\_relation\_article\_to\_categories\_table**



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

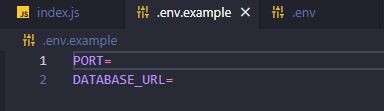


**npx prisma migrate dev --name add\_timestamp\_to\_articles\_table**



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

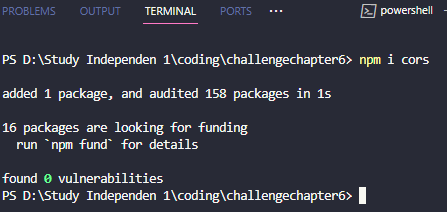
File ini digunakan agar developm lain tau apa aja yang harus ada di file **.env**

****

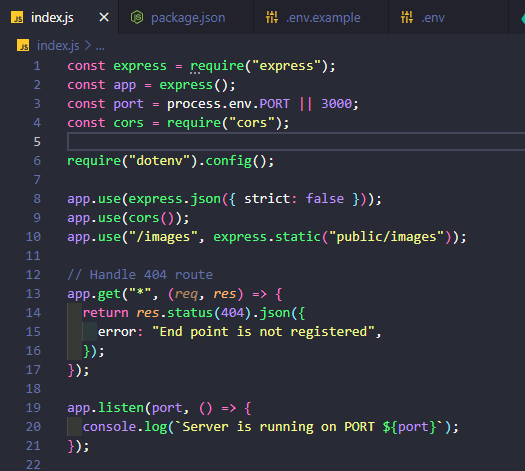
1. Install cors

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

**npm i cors**

****

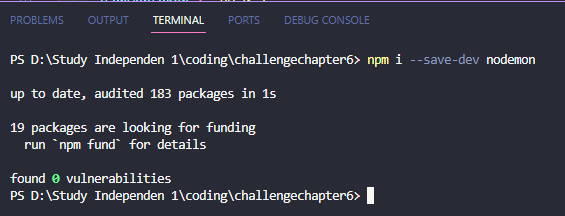
1. Isikan di file utama **index.js**



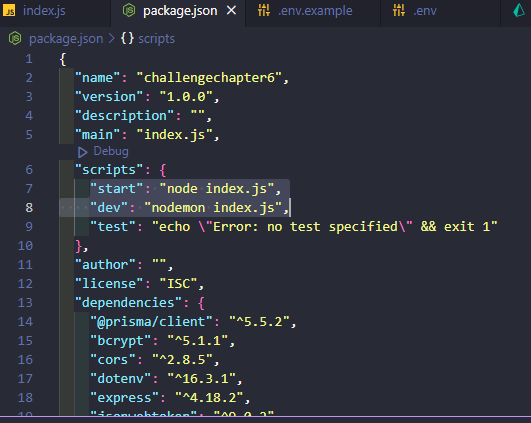
1. Install nodemon

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

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

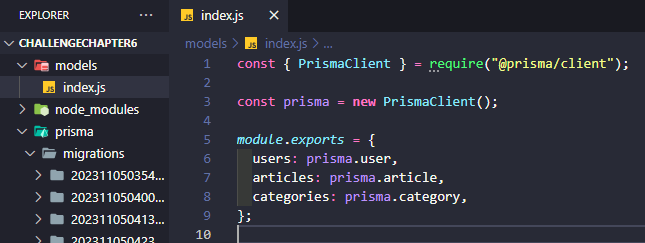


1. Tambahkan pada bagian file **package.json** untuk mengatur start dan dev nya

****

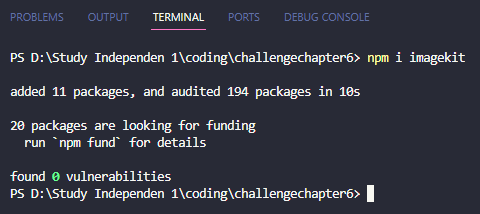
1. Tambahkan folder **models**

Di dalam folder ditambahkan file index.js

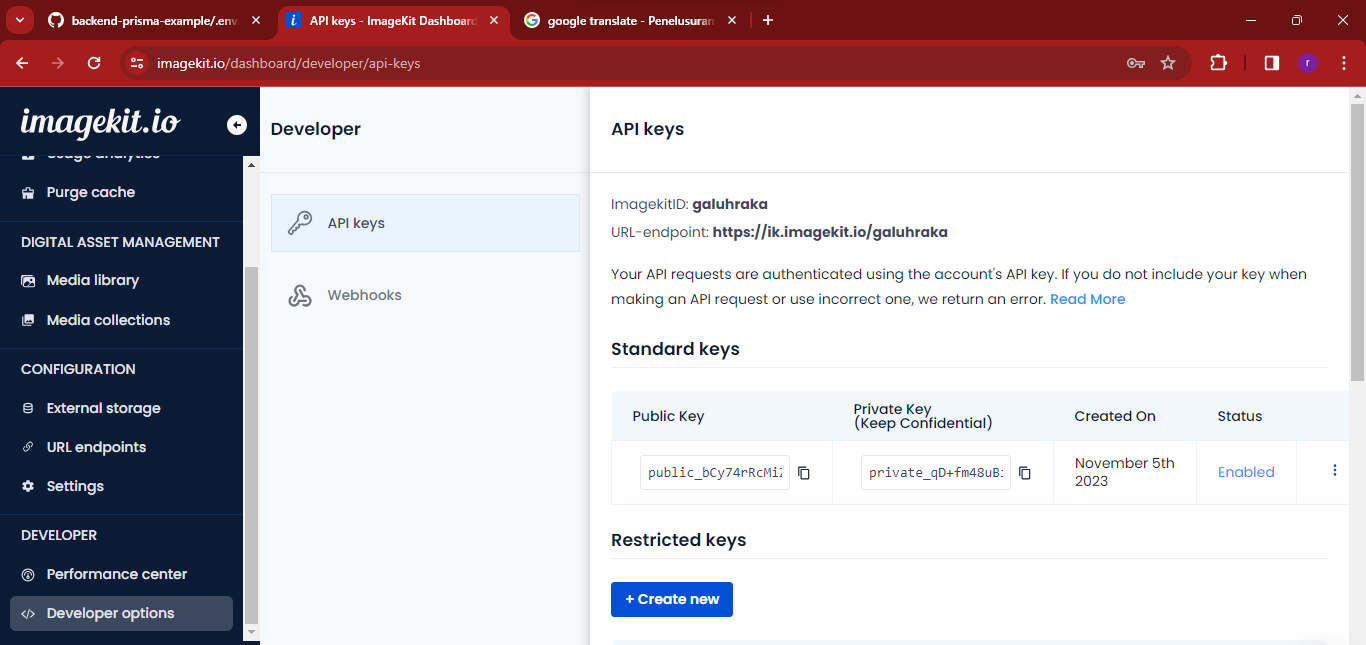


1. Install imageKit

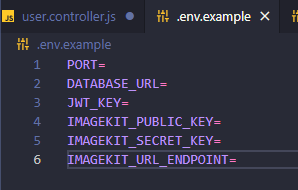
**npm i imagekit**



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

****

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

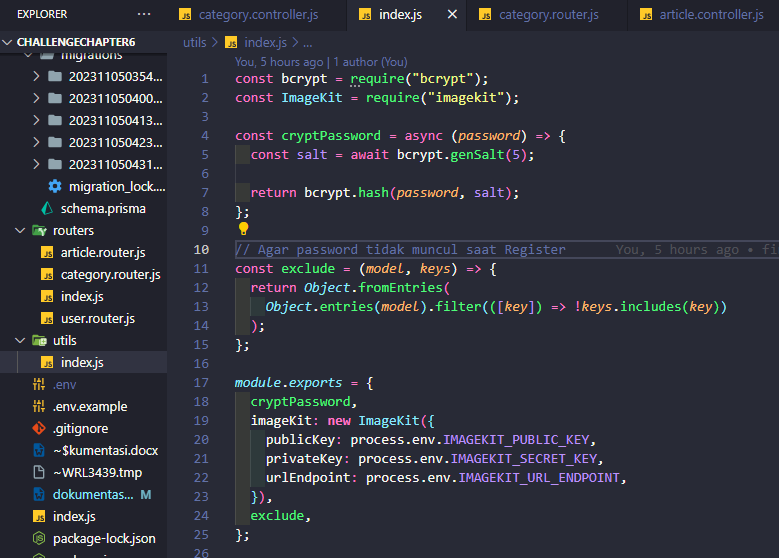


**IMAGEKIT\_PUBLIC\_KEY, IMAGEKIT\_SECRET\_KEY, IMAGEKIT\_URL\_ENDPOINT** .Di dapat dari ImageKit.io

**setting -> Developer Option**

1. Tambahkan folder **utils**

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

****

1. Tambahkan folder **middlewares**

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

Tujuannya untuk menghandling error kita



Gunakan function next untuk disemua controller

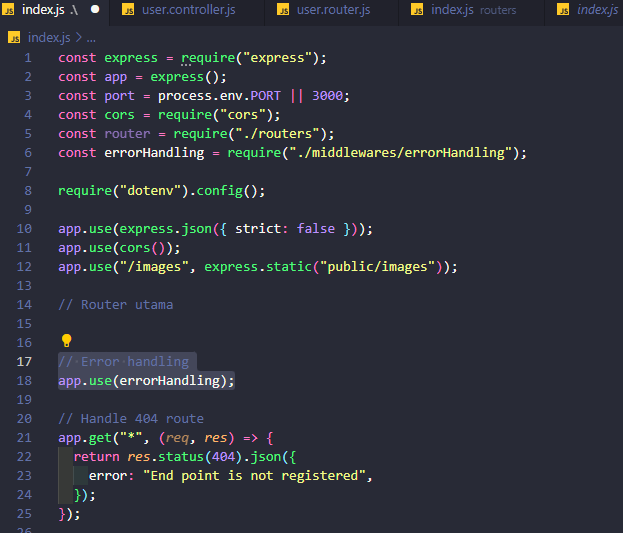


1. Tambahkan folder **controllers**

Di dalam folder tersebut terdapat 3 file

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

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

****

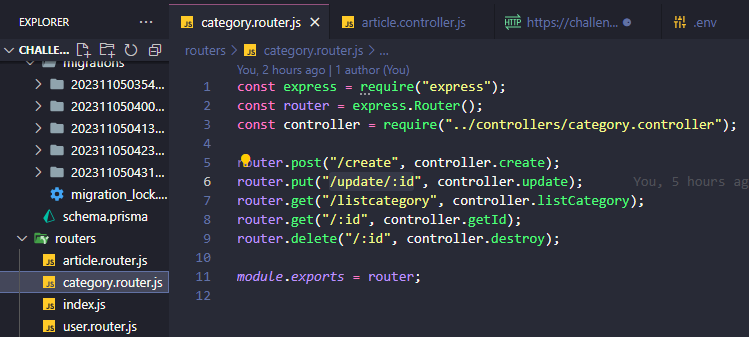
1. Tambahkan folder **routers**

Di dalam folder tersebut terdapat 4 file

* **user.router.js**



* **category.router.js**

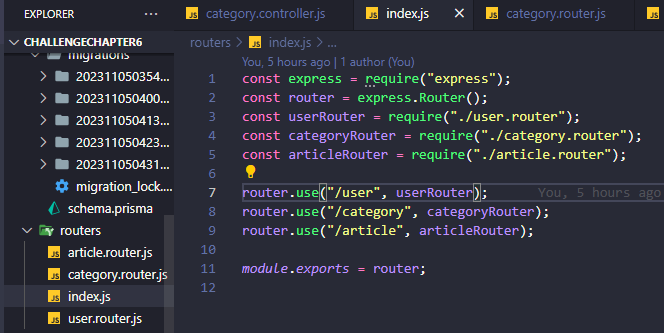


* **article.router.js**



* **index.js**

Untuk menghubungkan 3 router controllers



1. Sebelum men-deploy ke railway maka push ke github terlebih dahulu

Buat repository baru digithub terebih dahulu

Pada terminal ketikan:

**git init**

**git add .**

**git commit -m "first commit"**

**git branch -M main**

**git remote add origin https://github.com/rakkuy/challenge-chapter-6.git**

**git push -u origin main**

1. Dibagian file **package.json**

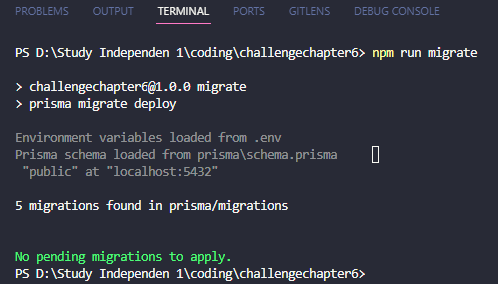
Tambahkan =

"migrate": "prisma migrate deploy"



1. Lalu test dengan=

**npm run migrate**



1. Lalu push ke github lagi

**git add .**

**git commit -m "add migrate script"**

**git push**

1. Buka **railway.app**
2. Lalu start a New Project
3. Pilih Deploy from GitHub repo
4. Pilih repository yg ingin di deploy
5. Deploy now
6. Karena masih error maka tambahkan bagian variablenya

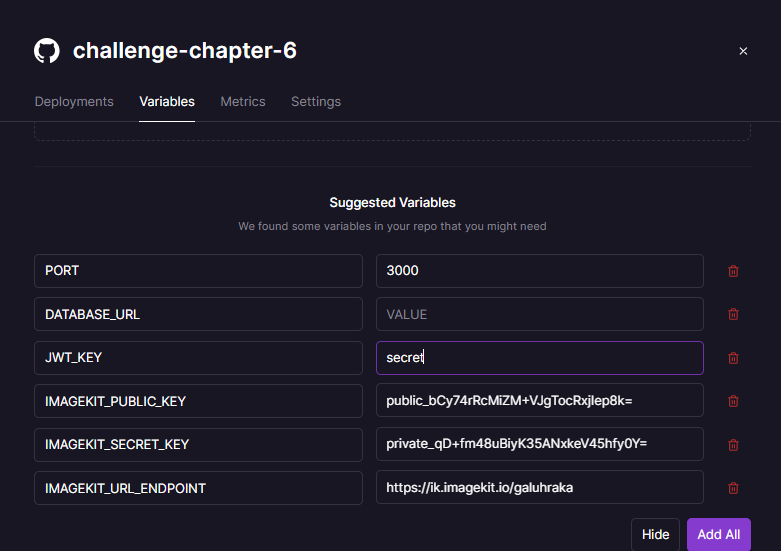
**Isikan Port,**

**JWT\_Key,**

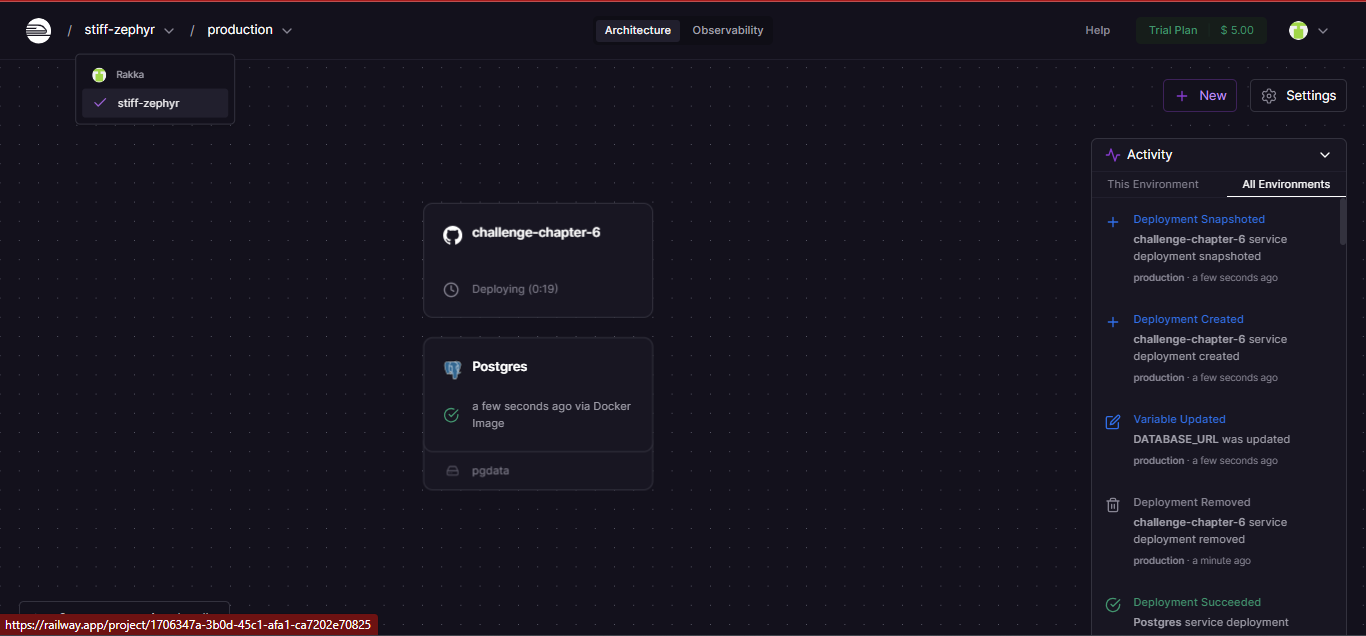
**IMAGEKIT\_PUBLIC\_KEY,**

**IMAGEKIT\_SECRET\_KEY,**

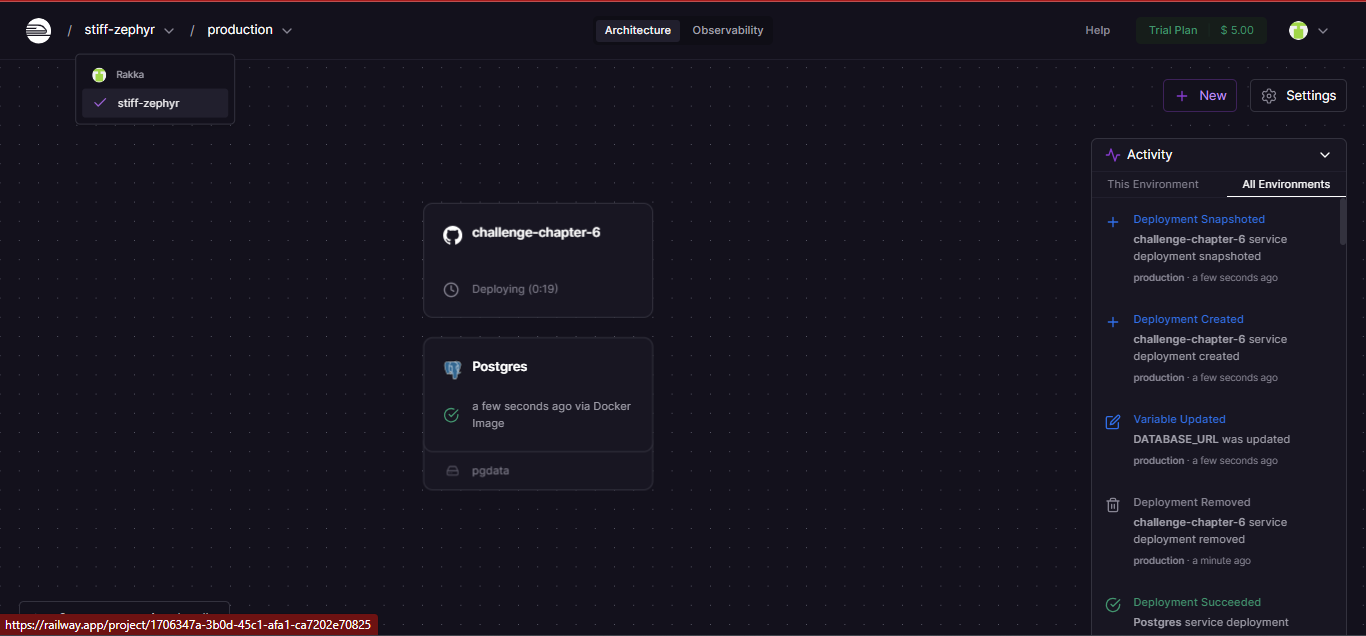
**IMAGEKIT\_URL\_ENDPOINT**



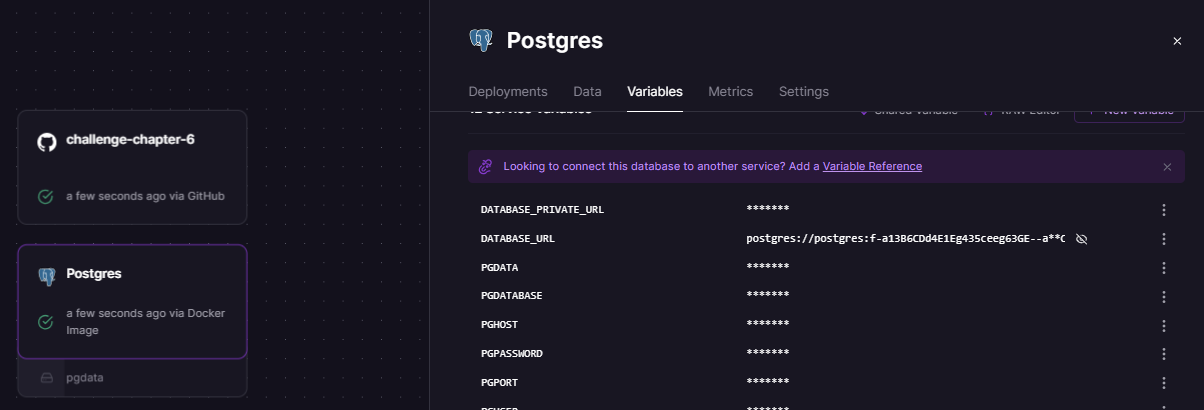
1. Di bagian pojok kiri atas, klik **stiff-zephyr**



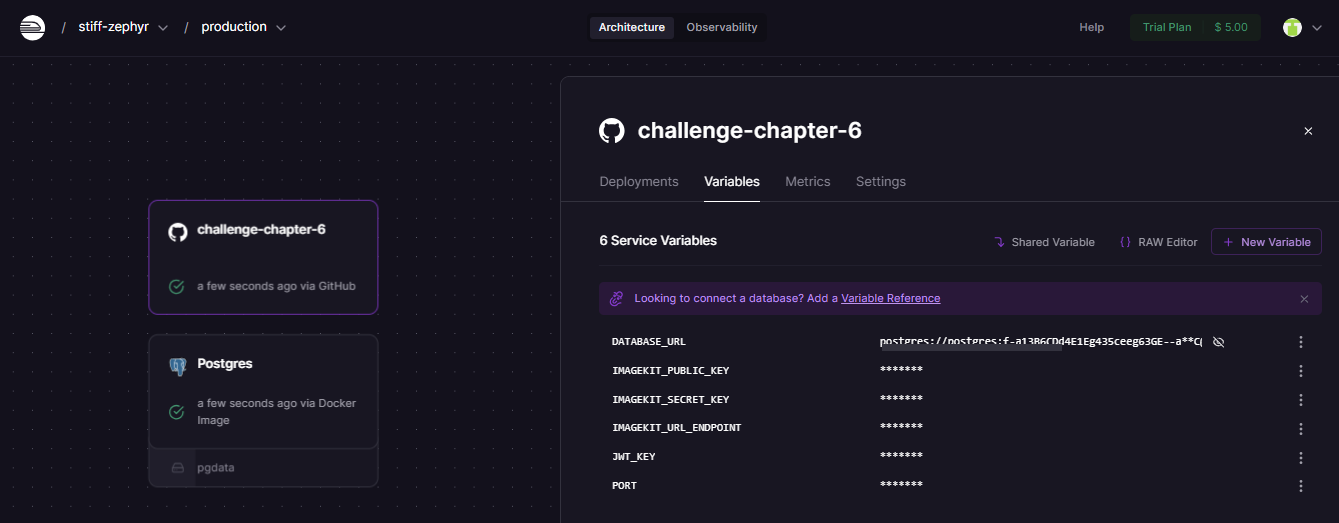
1. Klik **New** di bagian pojok kanan atas
2. Pilih Database Postgres dan Klik Database Postgres



1. Klik variable dan copy database linknya

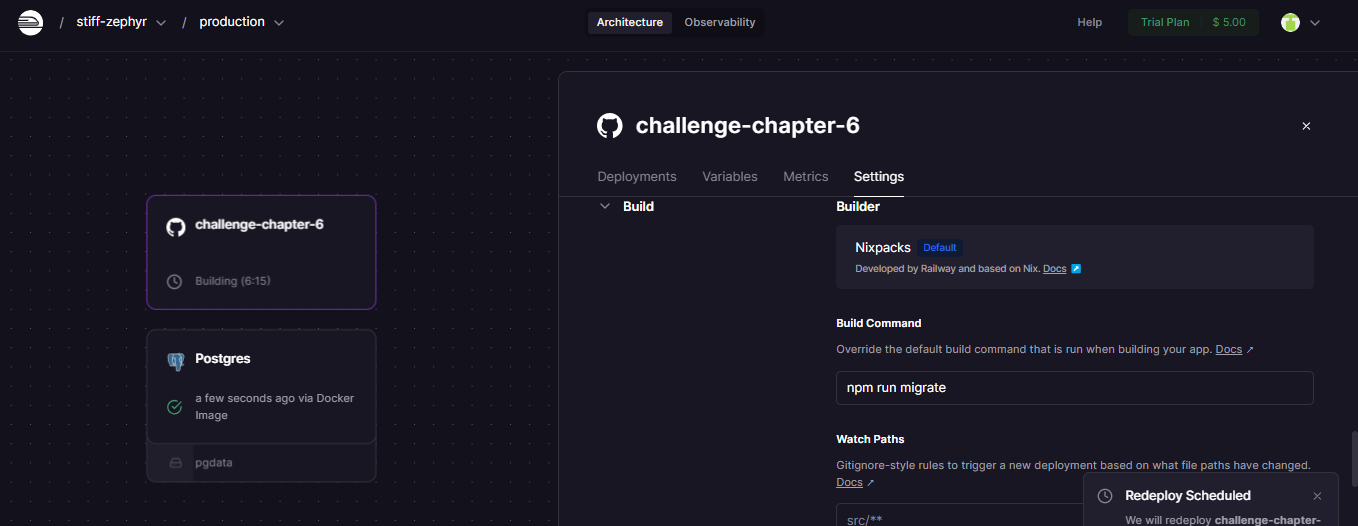


1. Paste link Database\_URL ke variable Projek kita

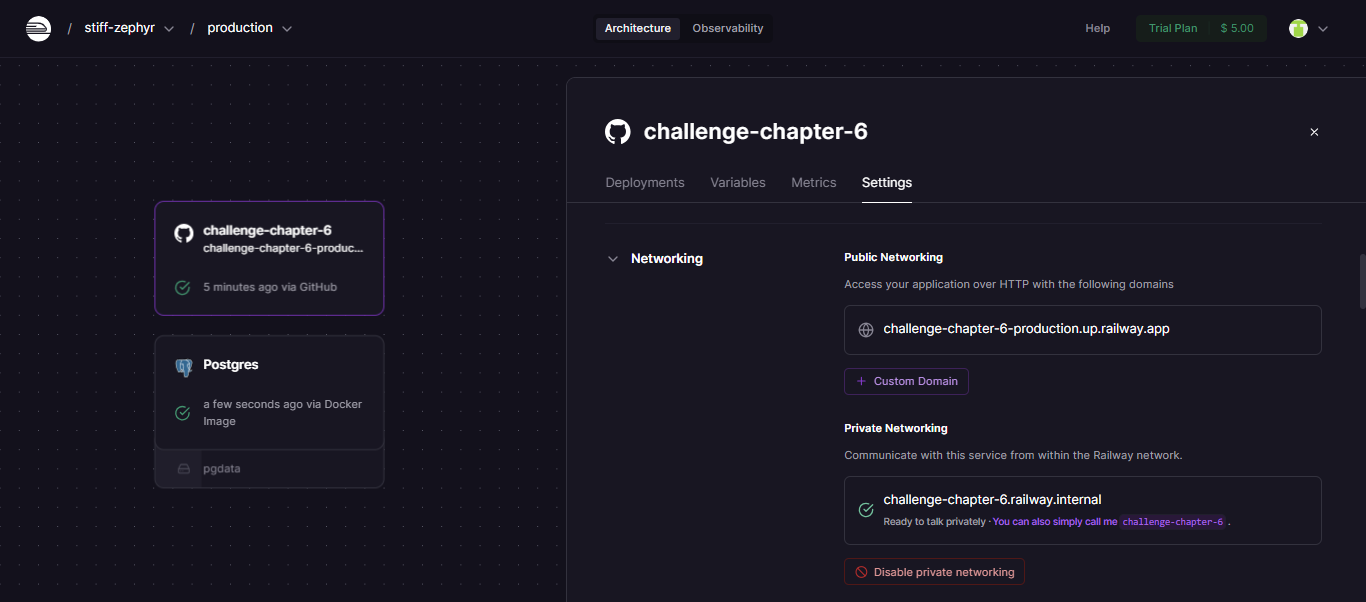


1. Klik Setting pada projek kita
2. Pada bagian Build Command, tuliskan:

**npm run migrate**



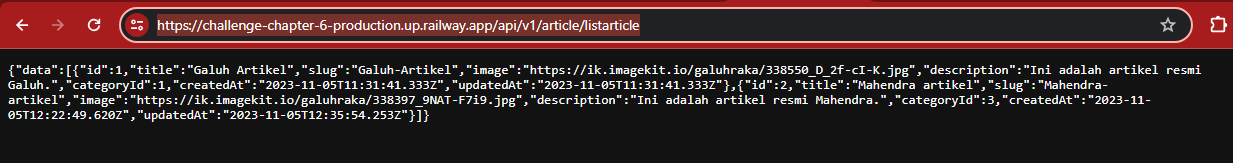
1. Untuk mengeceknya, copy pada bagian **public networking** nya



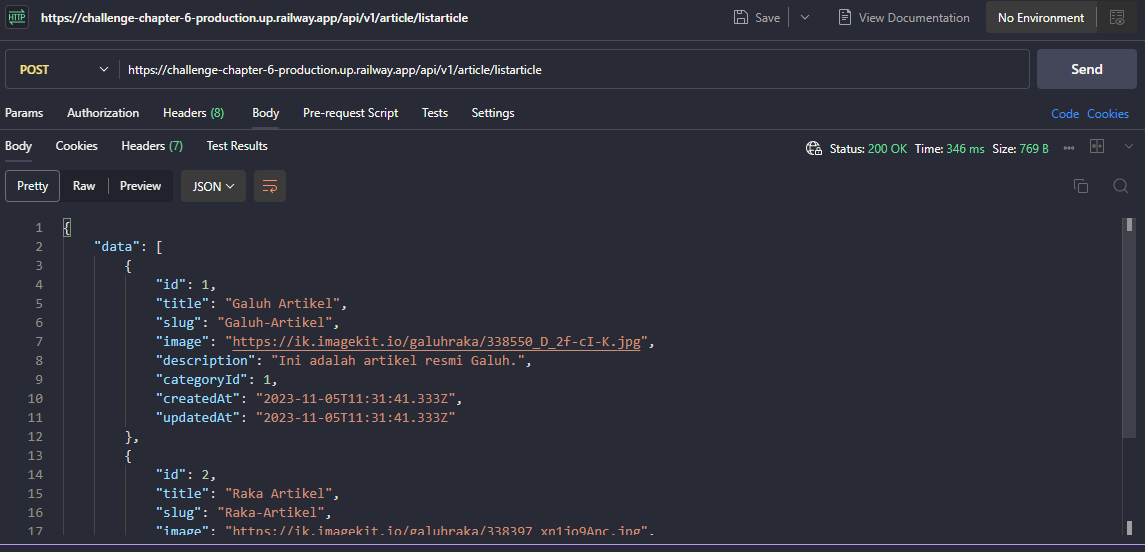
1. Ganti **DATABASE URL LOCAL** menjadi **DATABASE URL SERVER** dibagian file **.env**

Yg ada di variable Postgre di Railway

1. Lalu lakukan = **npx prisma migrate dev --name init**
2. Test di web

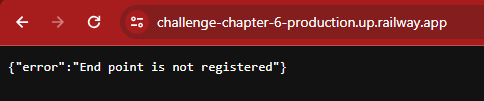


1. Test di postman



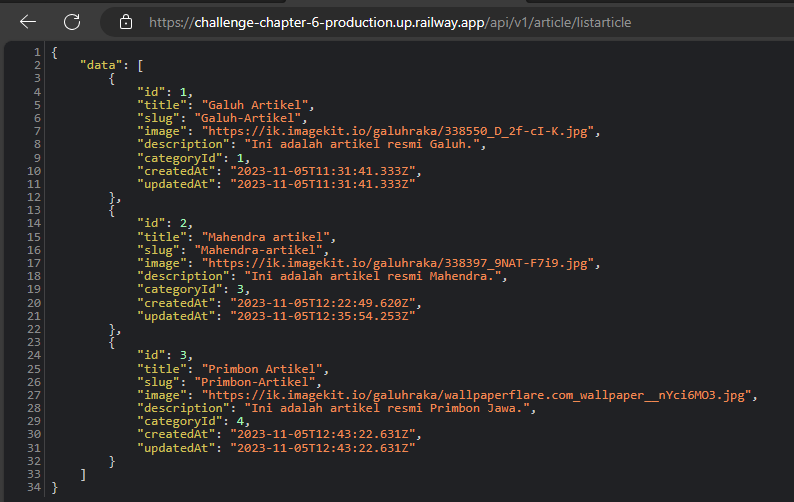
1. Link Railway =

[**https://challenge-chapter-6-production.up.railway.app**](https://challenge-chapter-6-production.up.railway.app)

****

1. Link untuk melihat data upload =

[**https://challenge-chapter-6-production.up.railway.app/api/v1/article/listarticle**](https://challenge-chapter-6-production.up.railway.app/api/v1/article/listarticle)

****