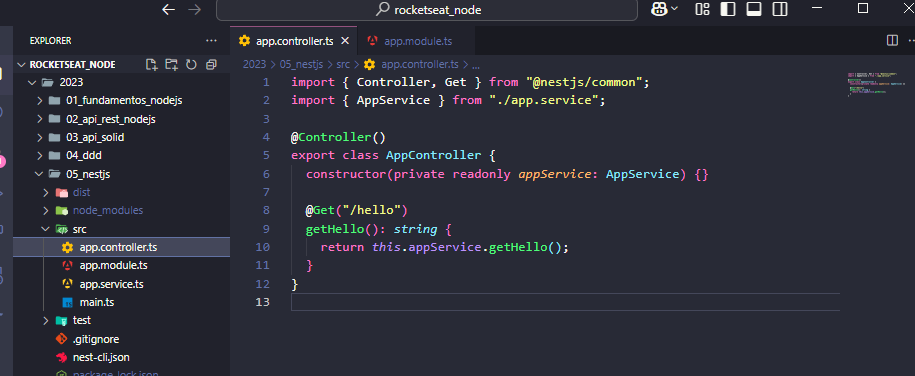
# 01 – Criando projeto com nest

Para iniciar o projeto em nest se vc tem o global na maquina nest new nome-projeto caso não tenha rode o comando npm i -g @nestjs/cli



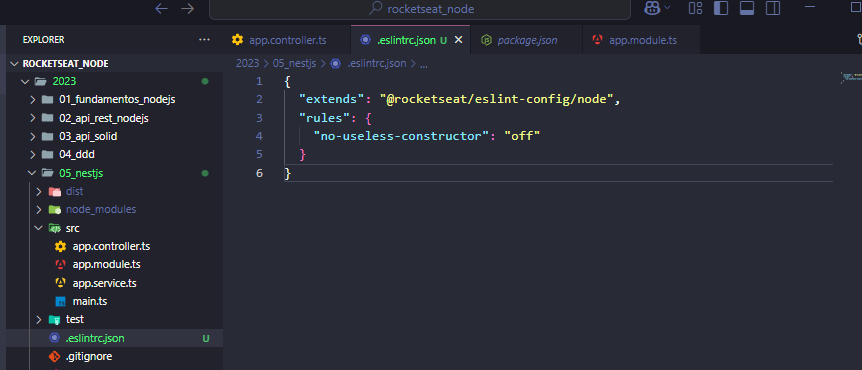
02 – Módulos serviços e controllers

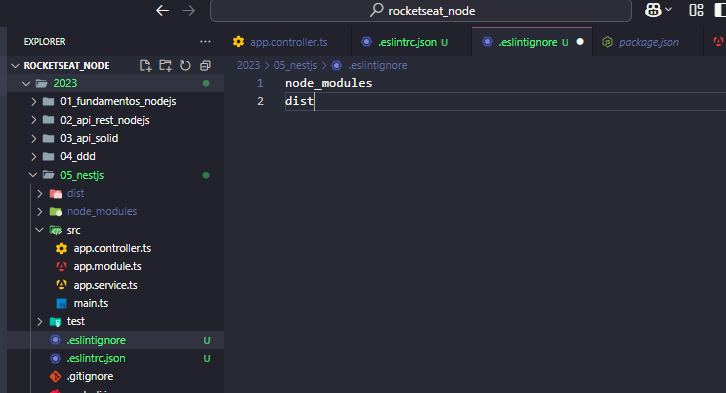
Controler é tudo que recebe requisição http o resto é provider



03 - Configurando Eslint e prettier

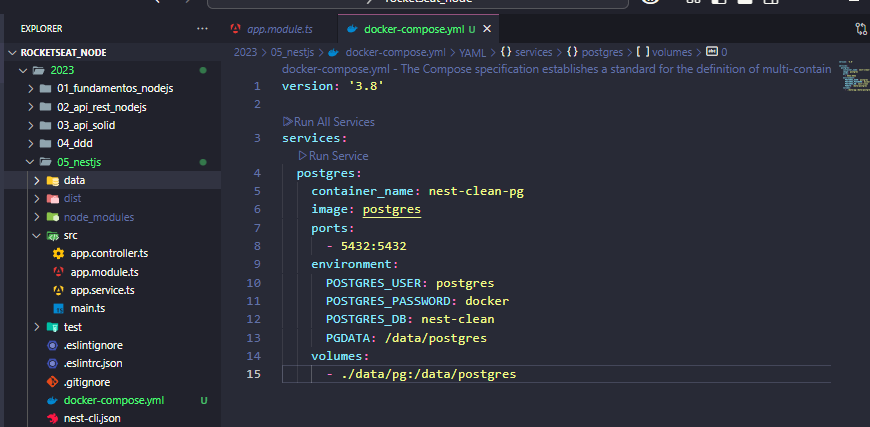
Instalar npm i eslint @rocketseat/eslint-config -D





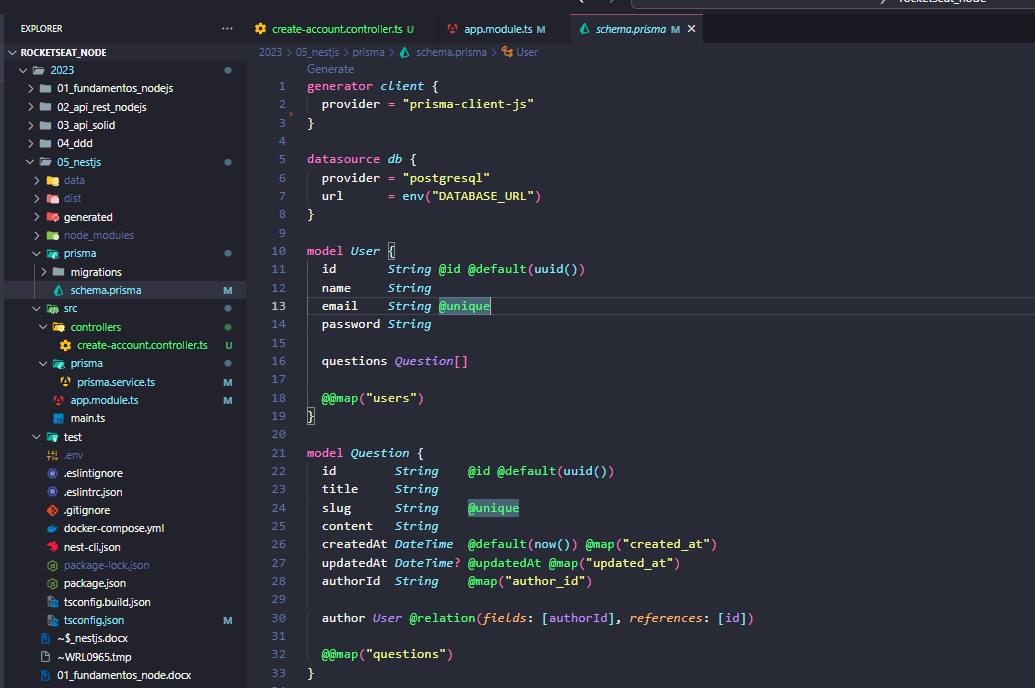
04 – Setup Docker compose

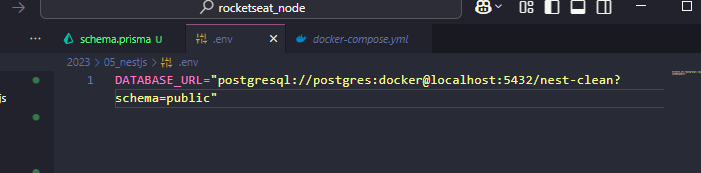
Depois rodar docker-compose up -d



05 - setup do prisma

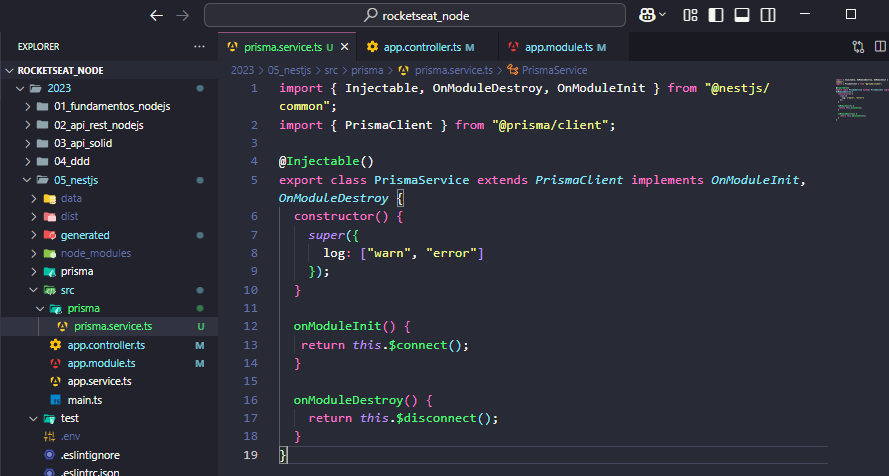
Instalar npm i prisma -D e depois npm i @prisma/cliente depois executar npx prisma init

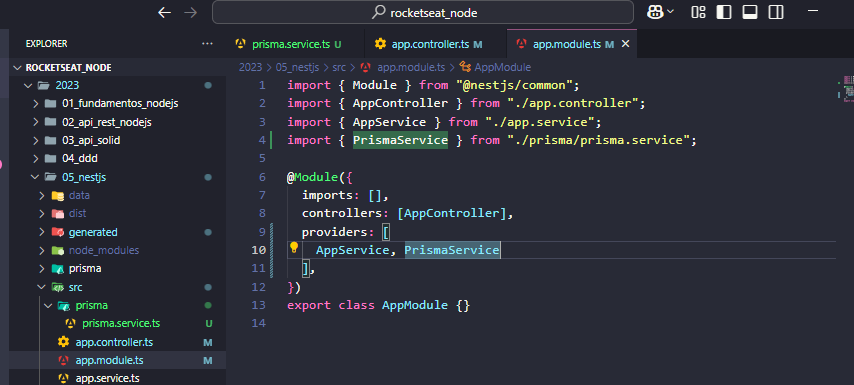




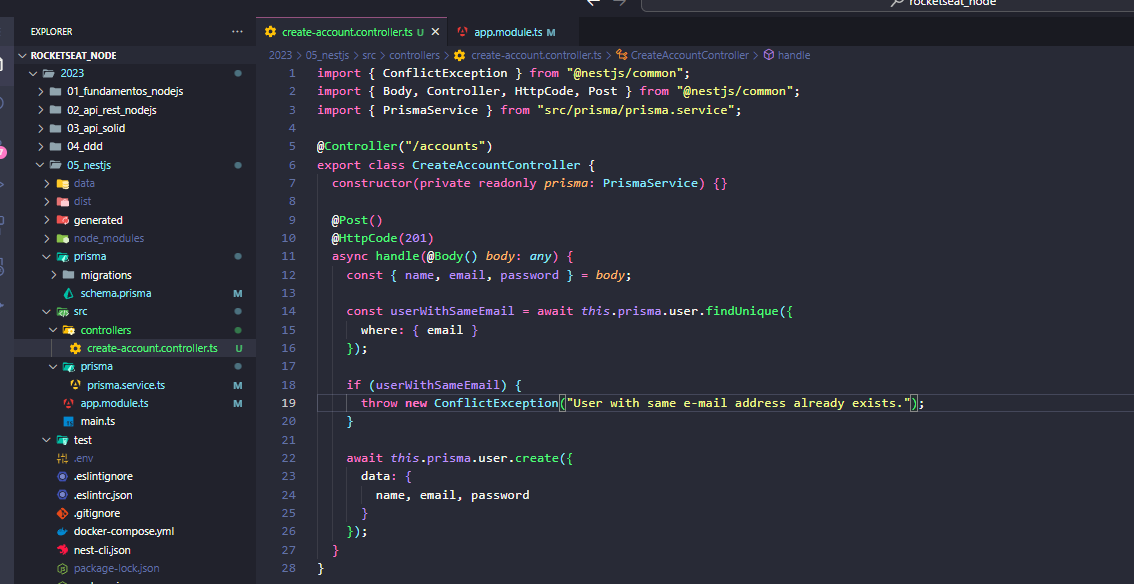
Depois rodar as migrations npx prisma migrate dev depois da um nome para migração e para ver a interface gráfica rodar npx prisma studio e para gerar as tipagem rodar npx prisma generate

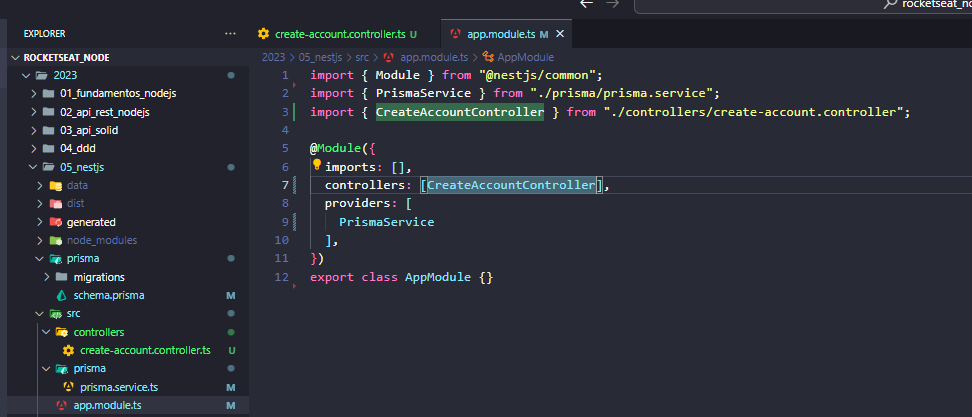
06 – Criando serviço do prisma





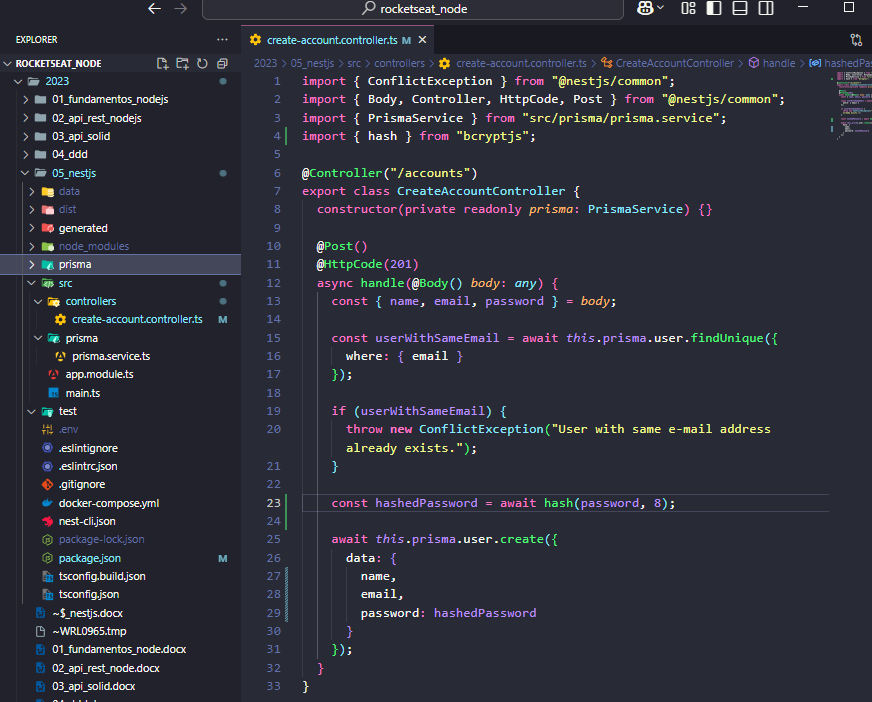
07 - Controller de criação de conta





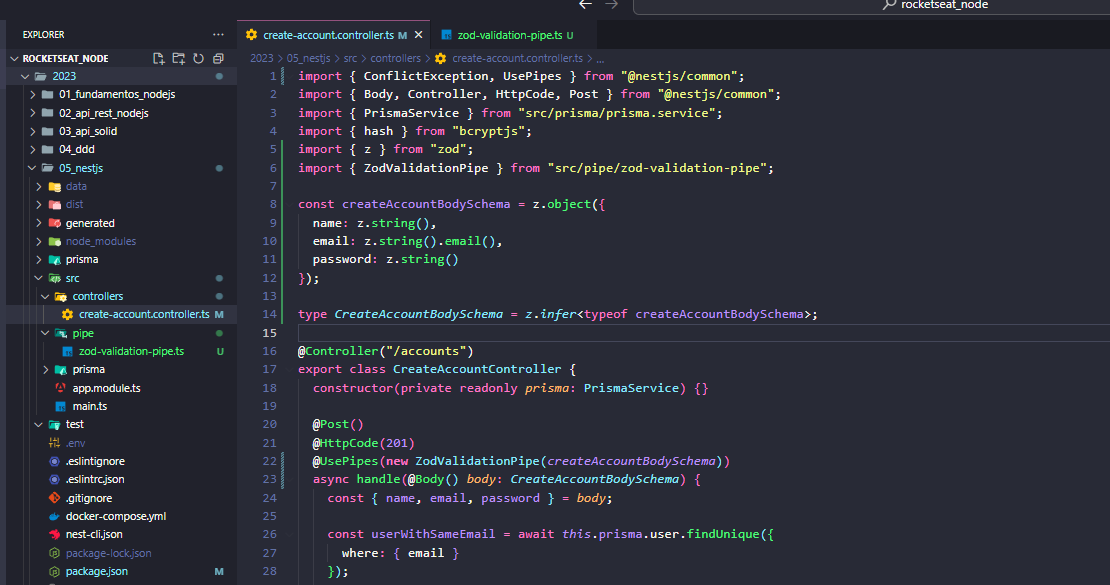
08 - Gerando hash da senha

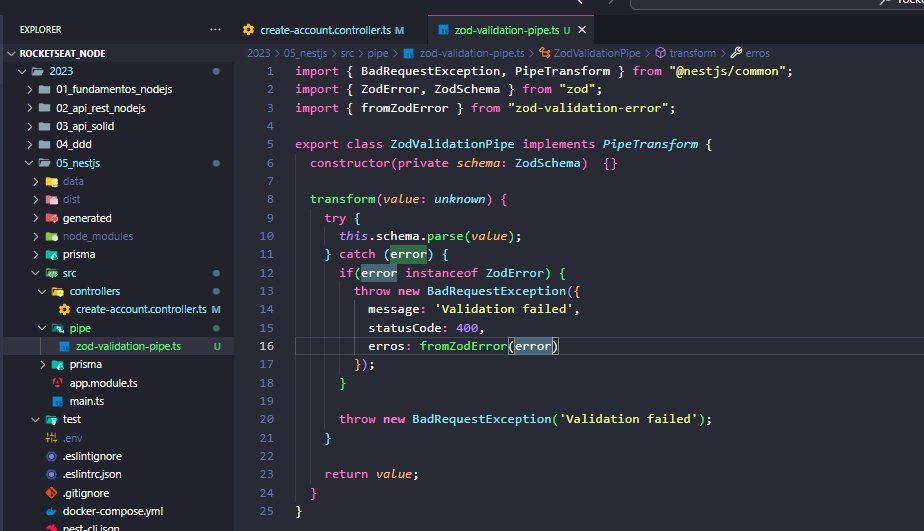
Instalar npm i bcryptjs e os tipes npm i @types/bcryptjs -D



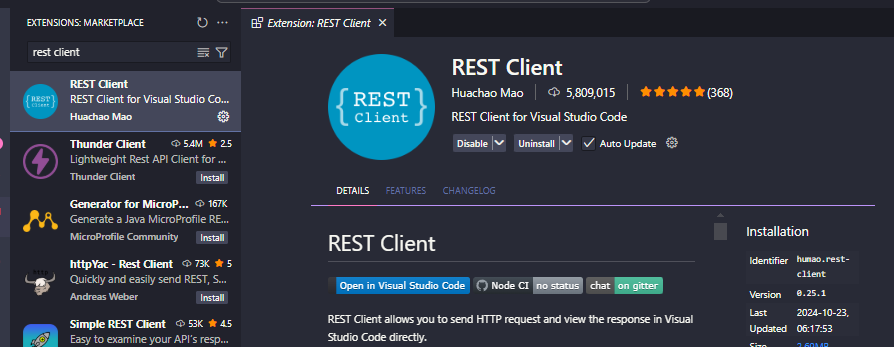
09 - Criando pipe de validação do zod

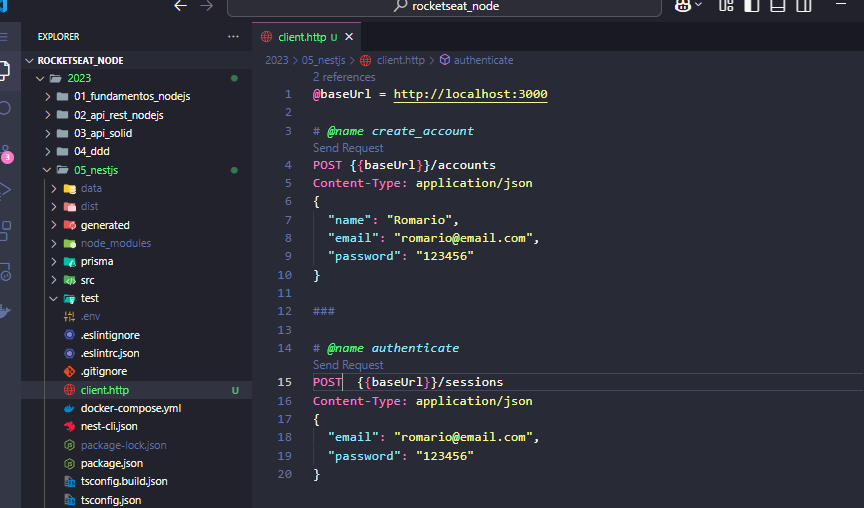
Instalar o npm i zod e também instalar npm i zod-validation-error





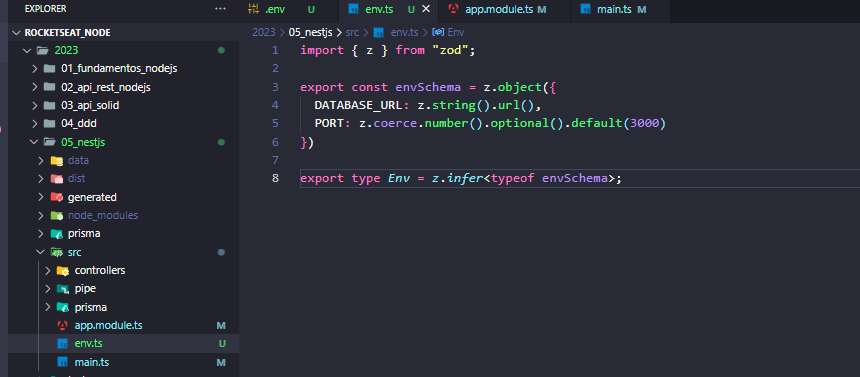
10 – Extensão rest cliente no vscode

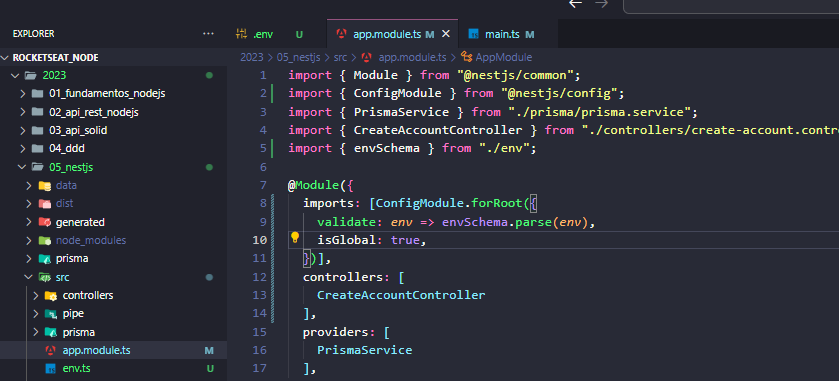


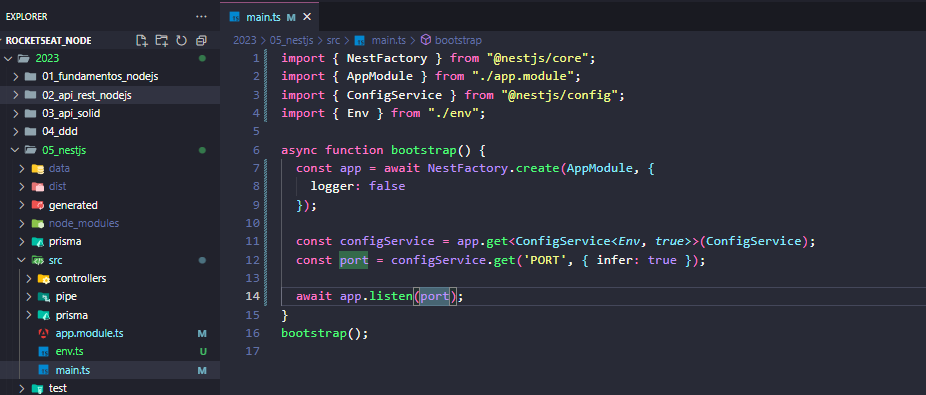


11 – Usando configmodule no nest.js

Instalar npm i @nestjs/config

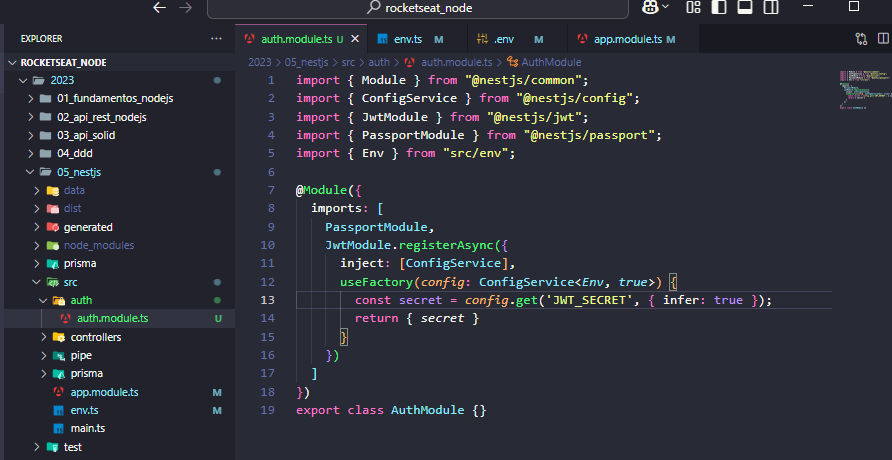


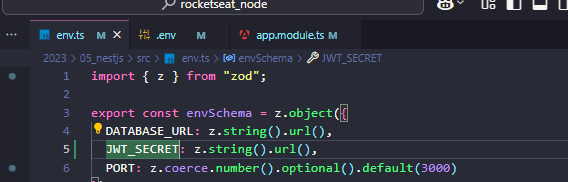


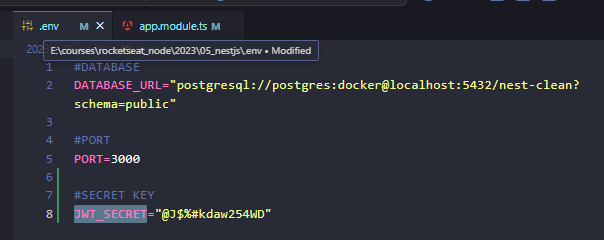


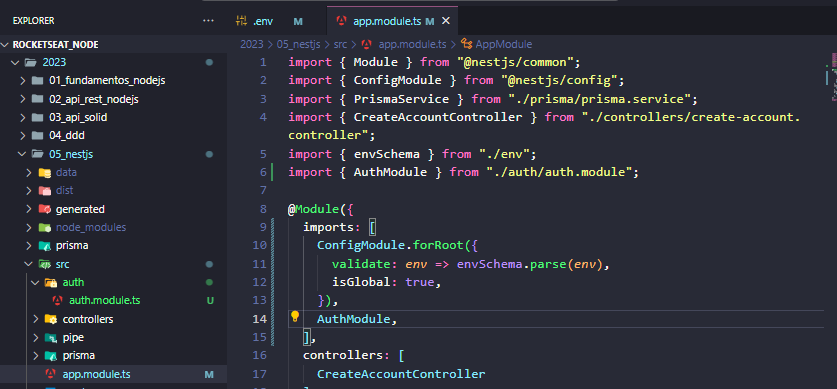
12 – Configurando autenticação jwt

Instalar npm i @nestjs/passport @nestjs/jwt









13 – Gerando token jwt

Gerar chave privada

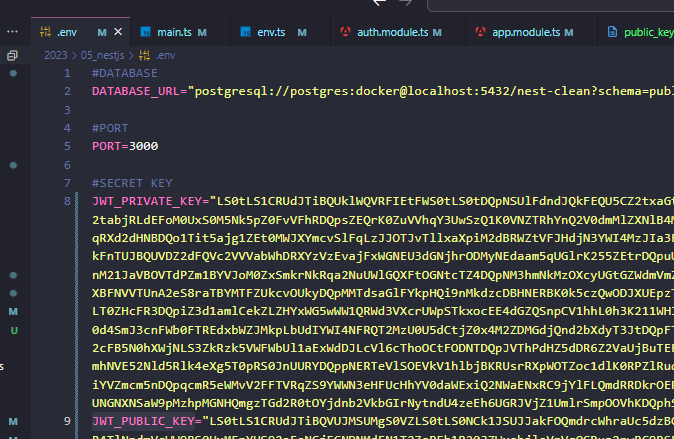
openssl genpkey -algorithm RSA -out private\_key.pem -pkeyopt rsa\_keygen\_bits:2048

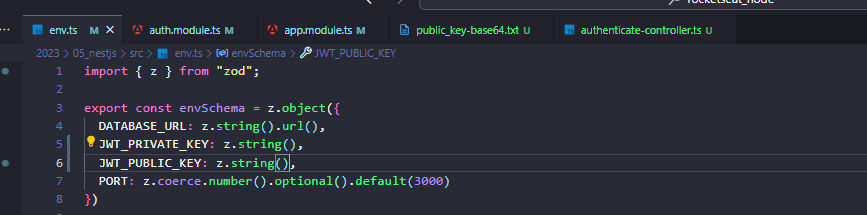
gerar chave publica

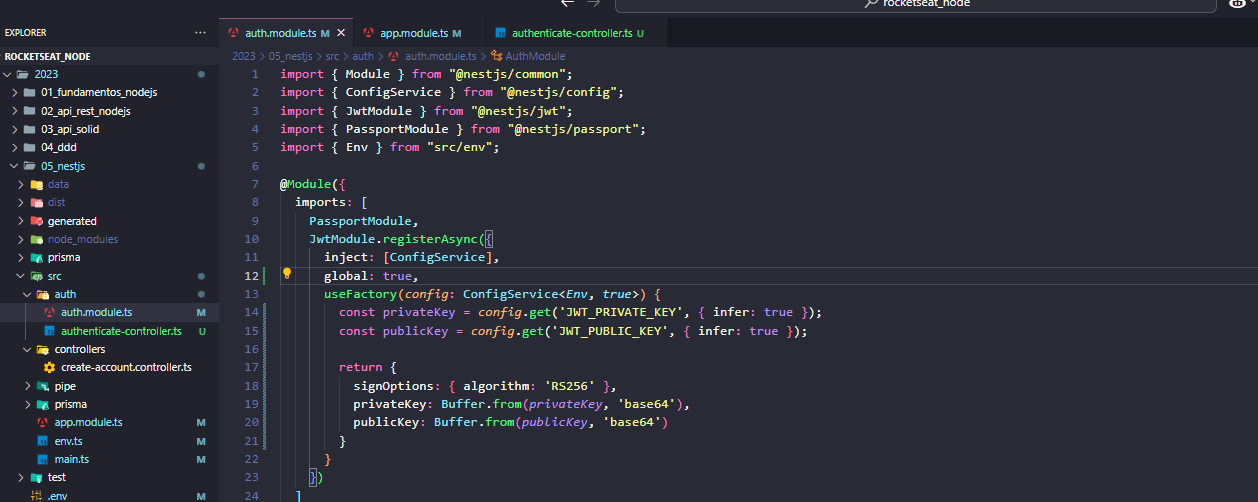
openssl rsa -pubout -in private\_key.pem -out public\_key.pem

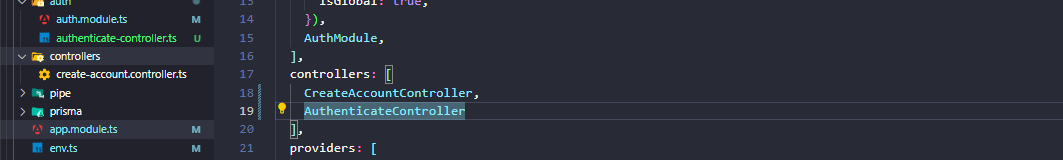
e para converter em base 64  
base64 private\_key.pem > private\_key-base64.txt

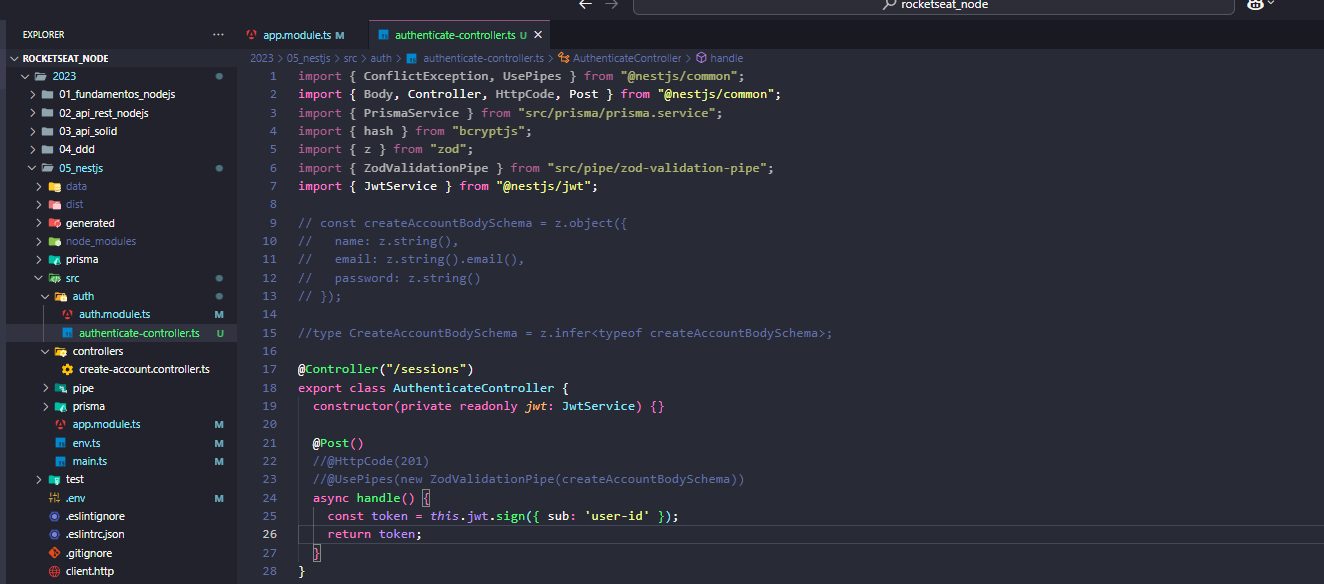
base64 public\_key.pem > public\_key-base64.txt



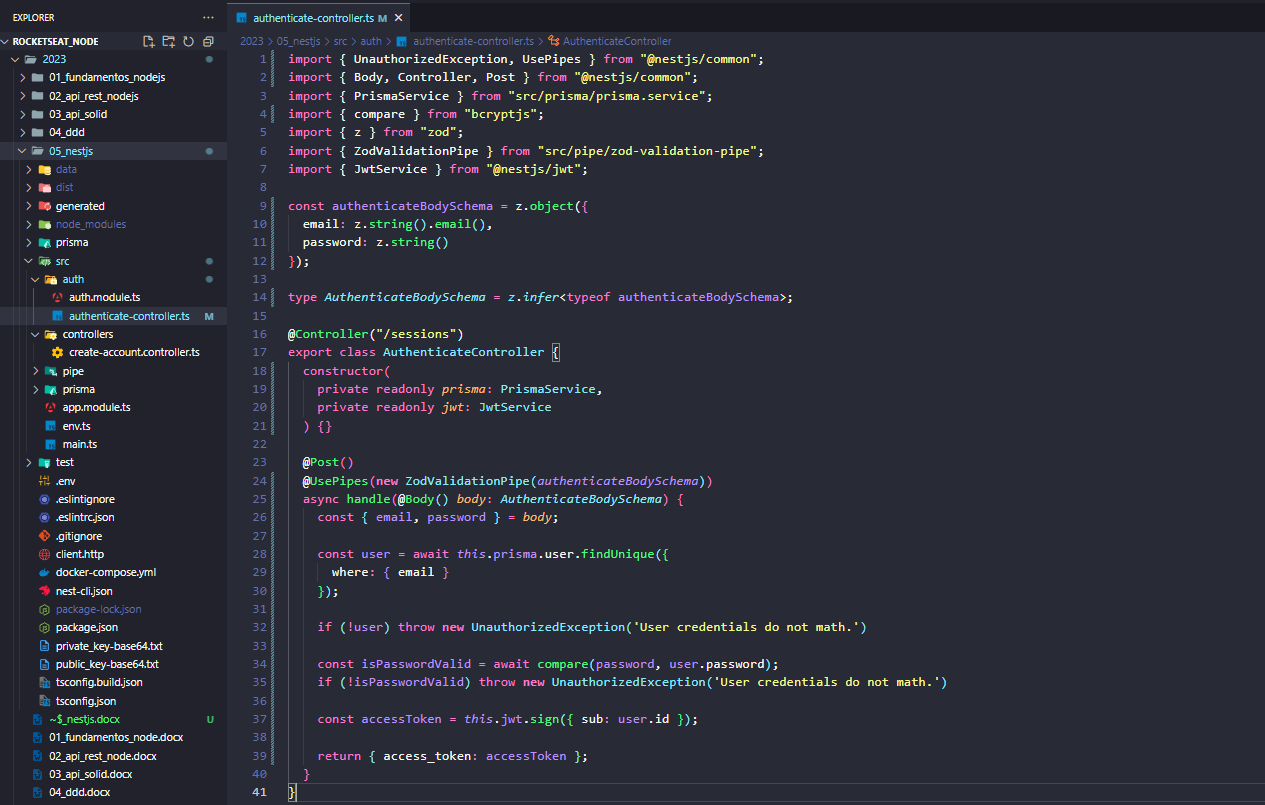






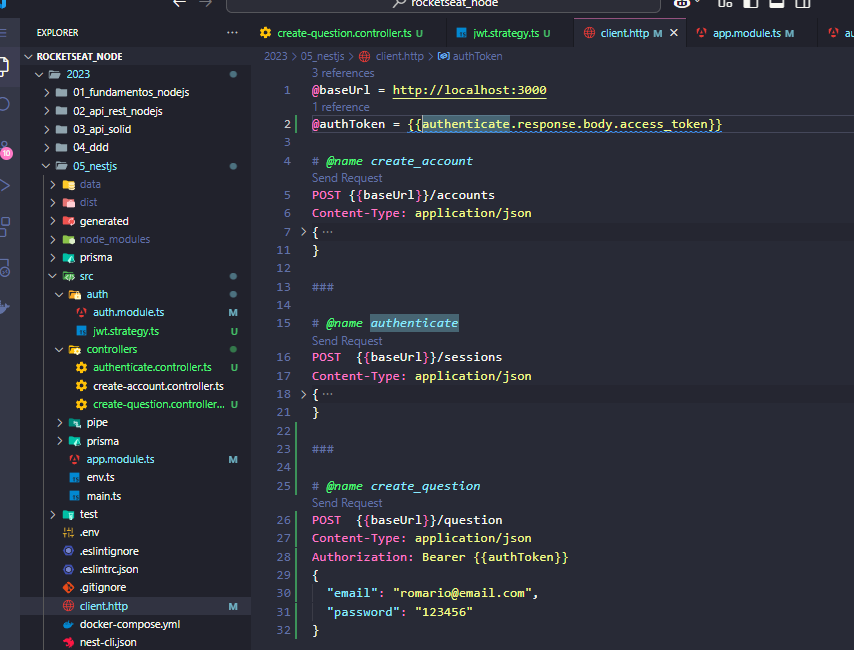


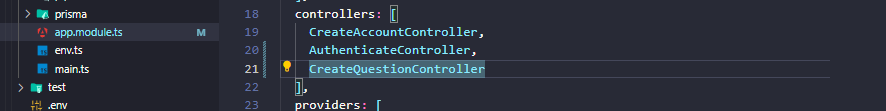
14 – Controller de autenticação

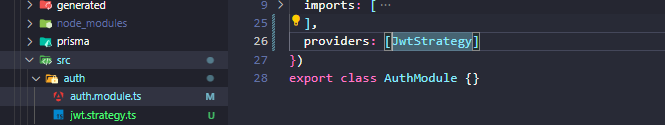


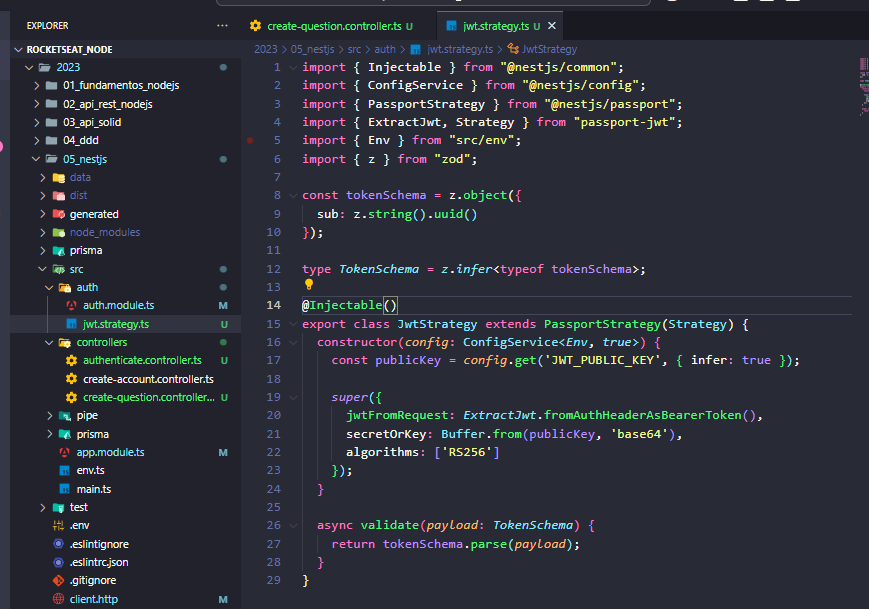
15 – Protegendo rotas com guards

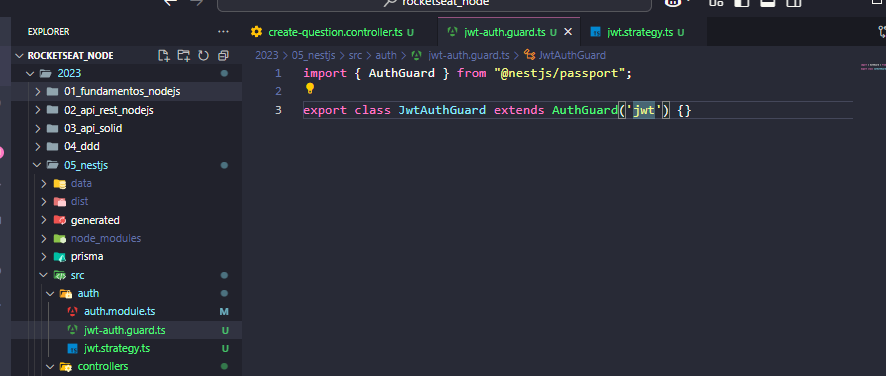
Instalar npm i passsport-jwt e o type npm i @types/passport-jwt -D

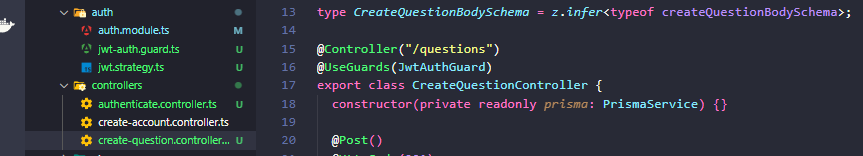




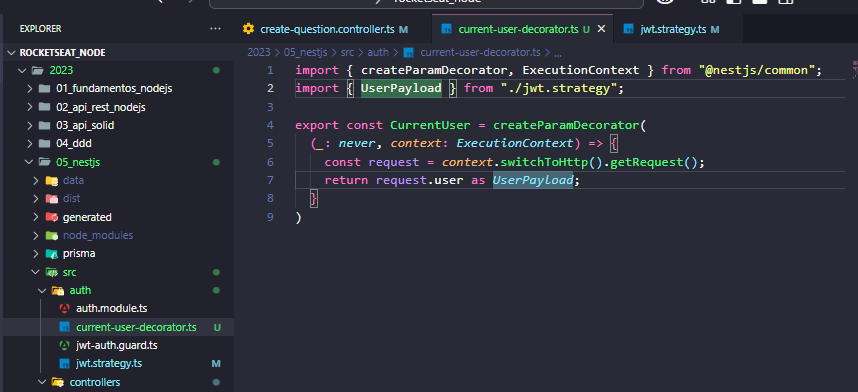


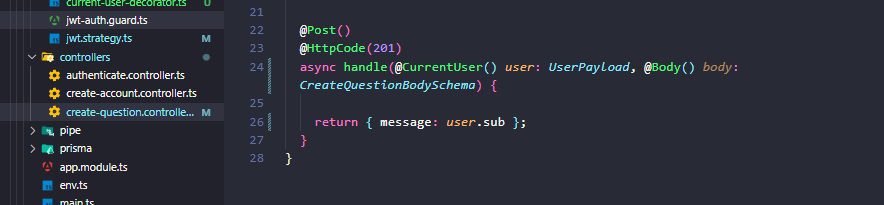


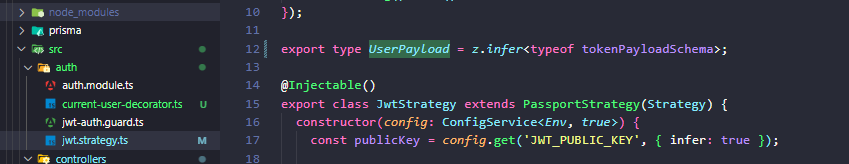




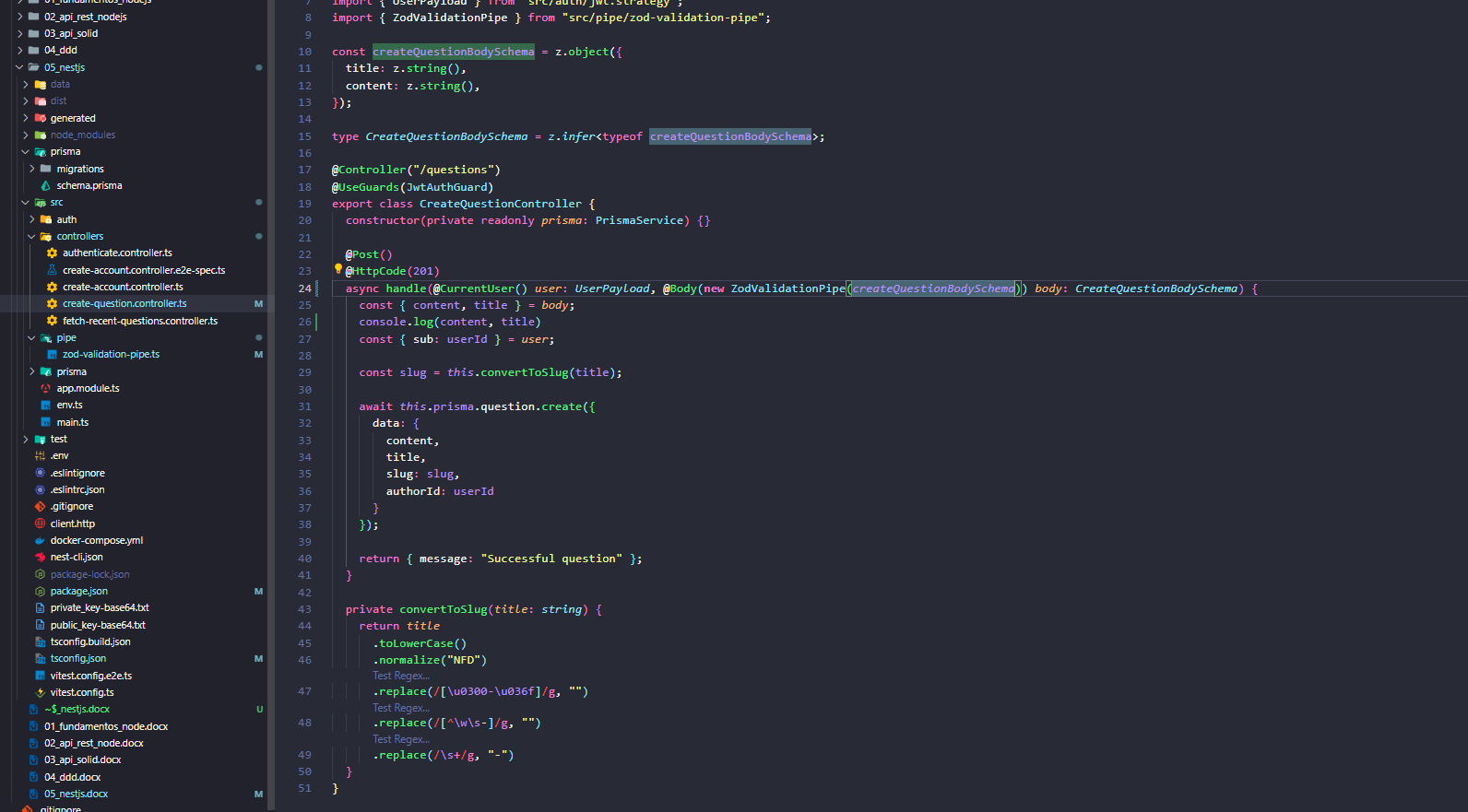
16 – criando decorator de autenticação



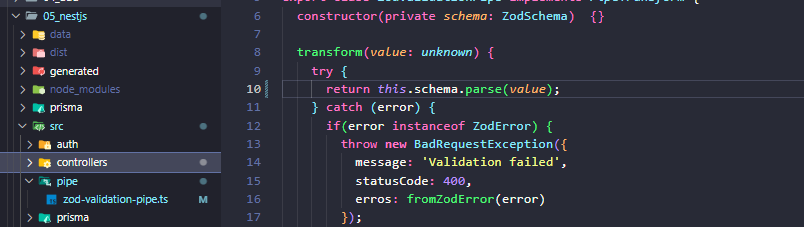


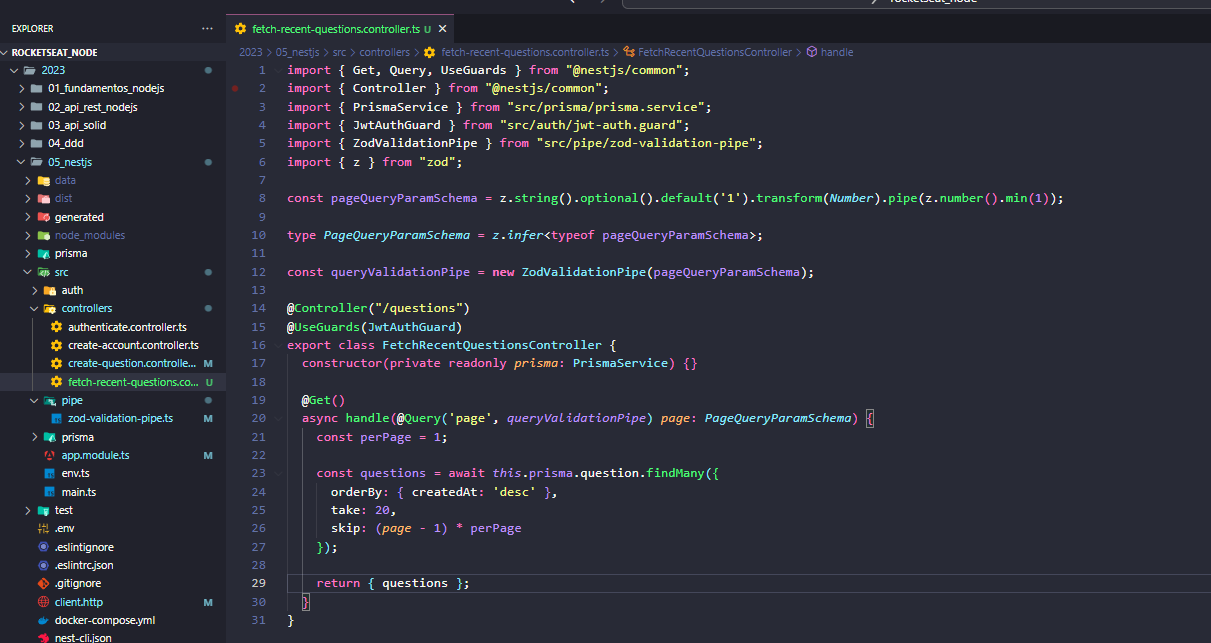


17 – controler de criação de pergunta



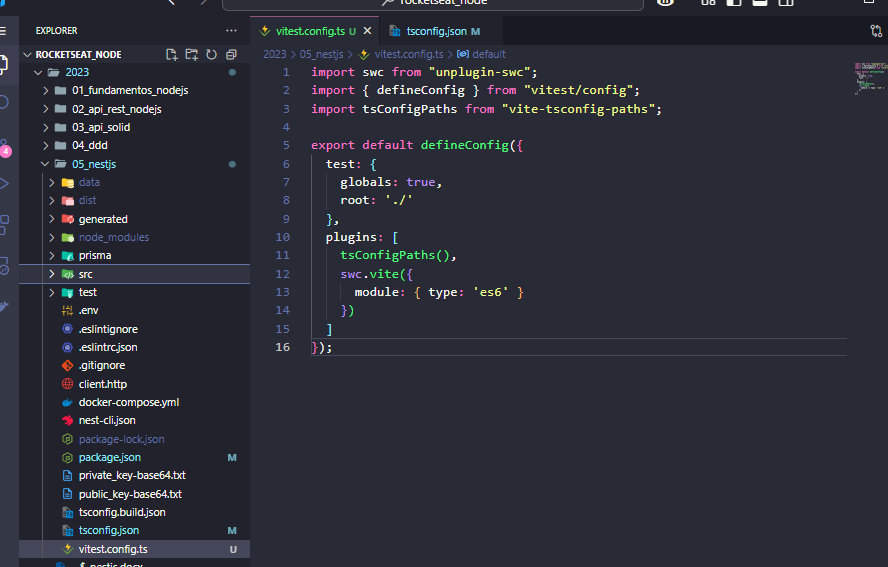
18 – controler de listagem de perguntas

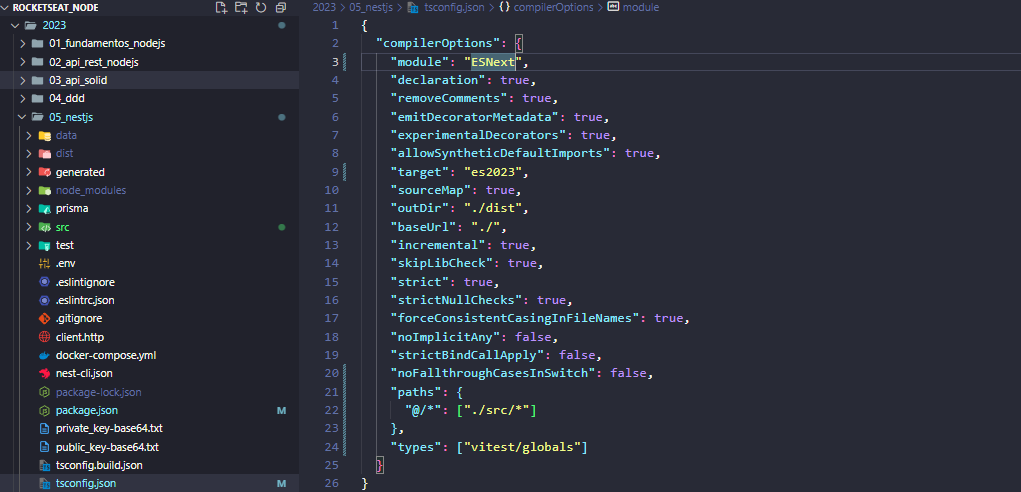


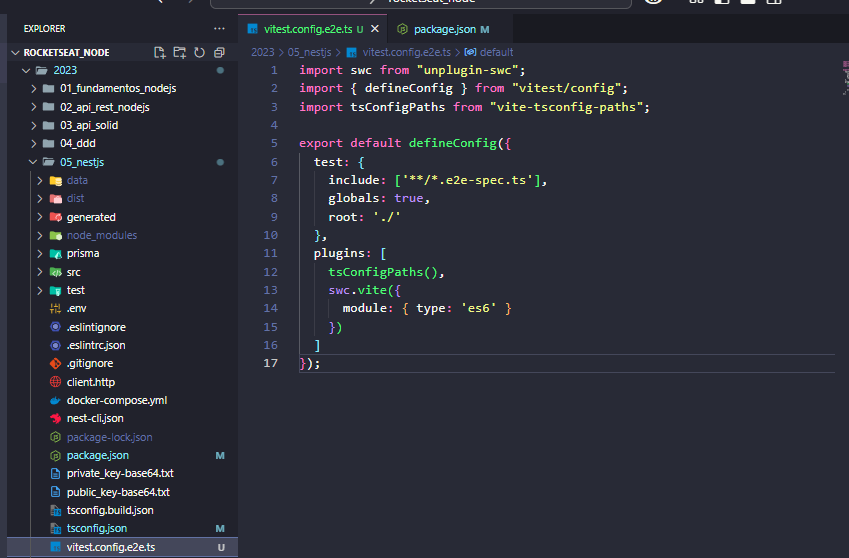


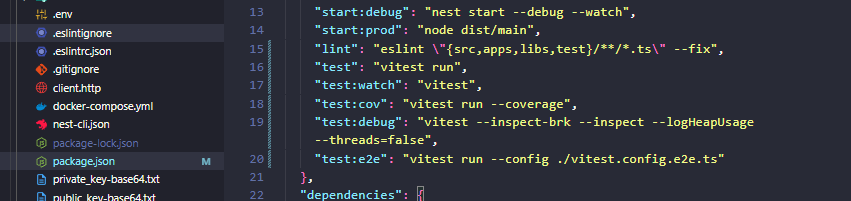
19 – Configurando vitest com swc

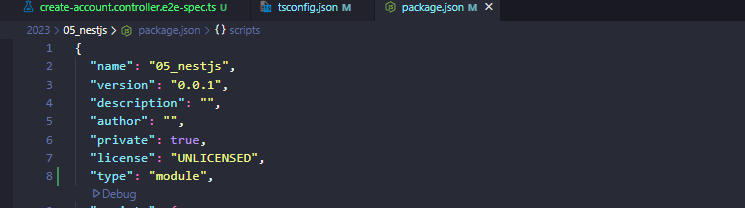
Instalar npm i vitest unplugin-swc @swc/core @vitest/coverage-v8 -D e também npm i vitest-tsconfig-paths -D





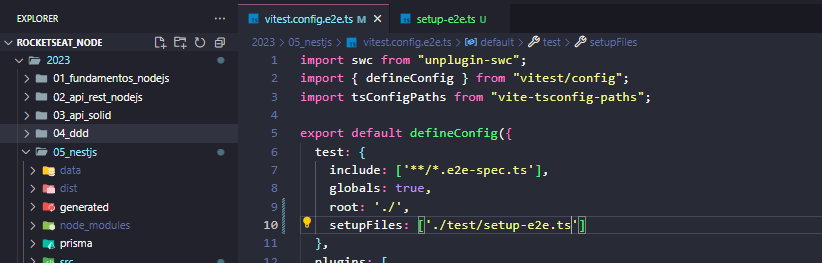


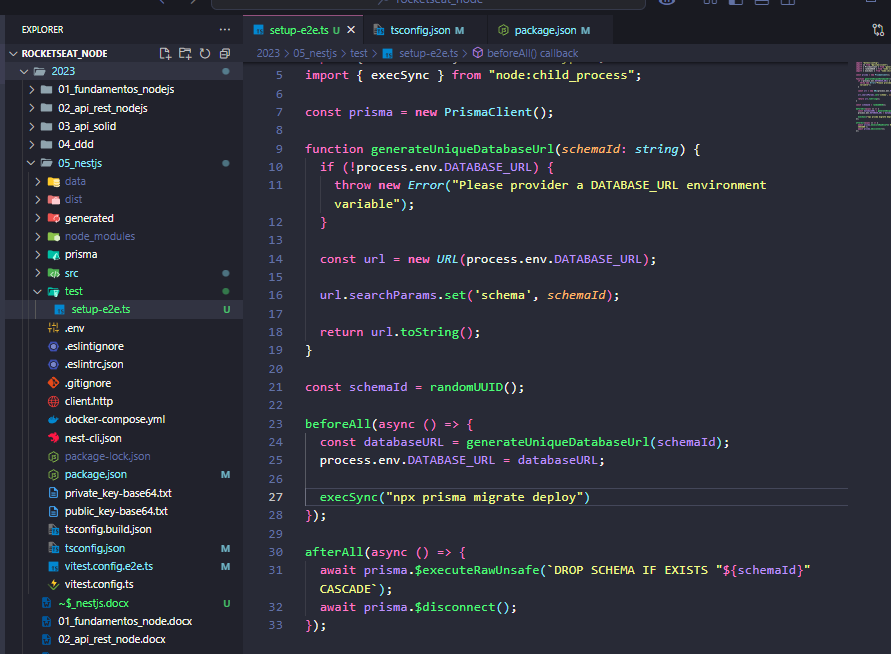




20 - Banco de dados isolado nos testes

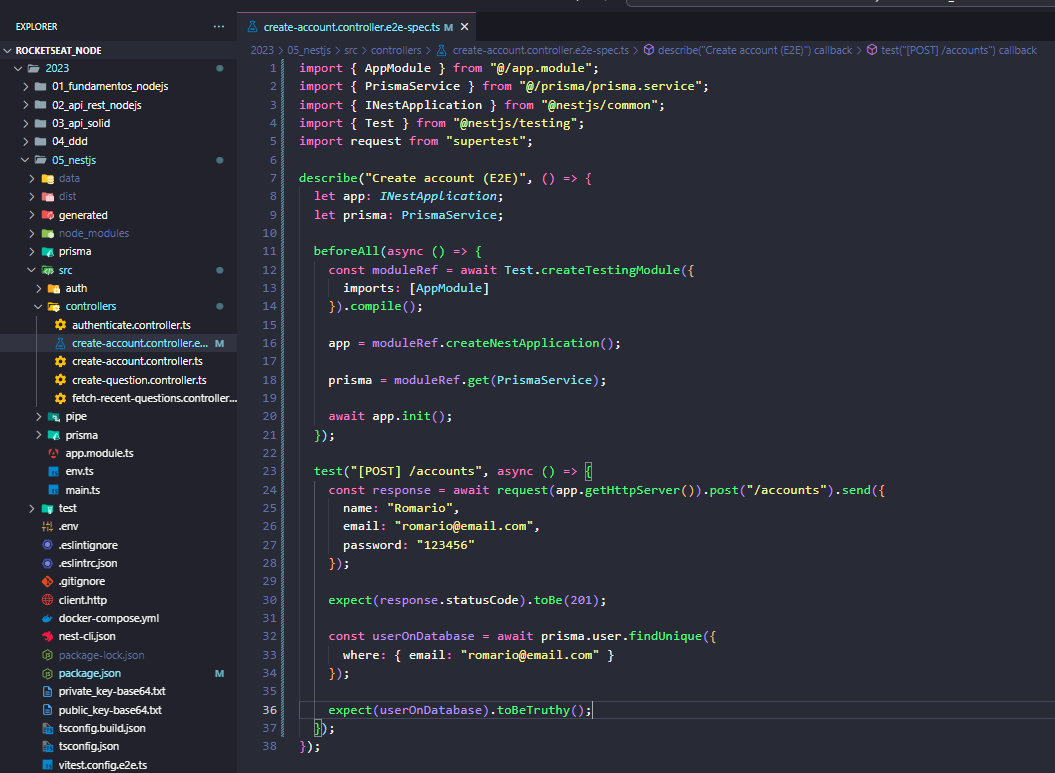
Instalar npm i dotenv -D

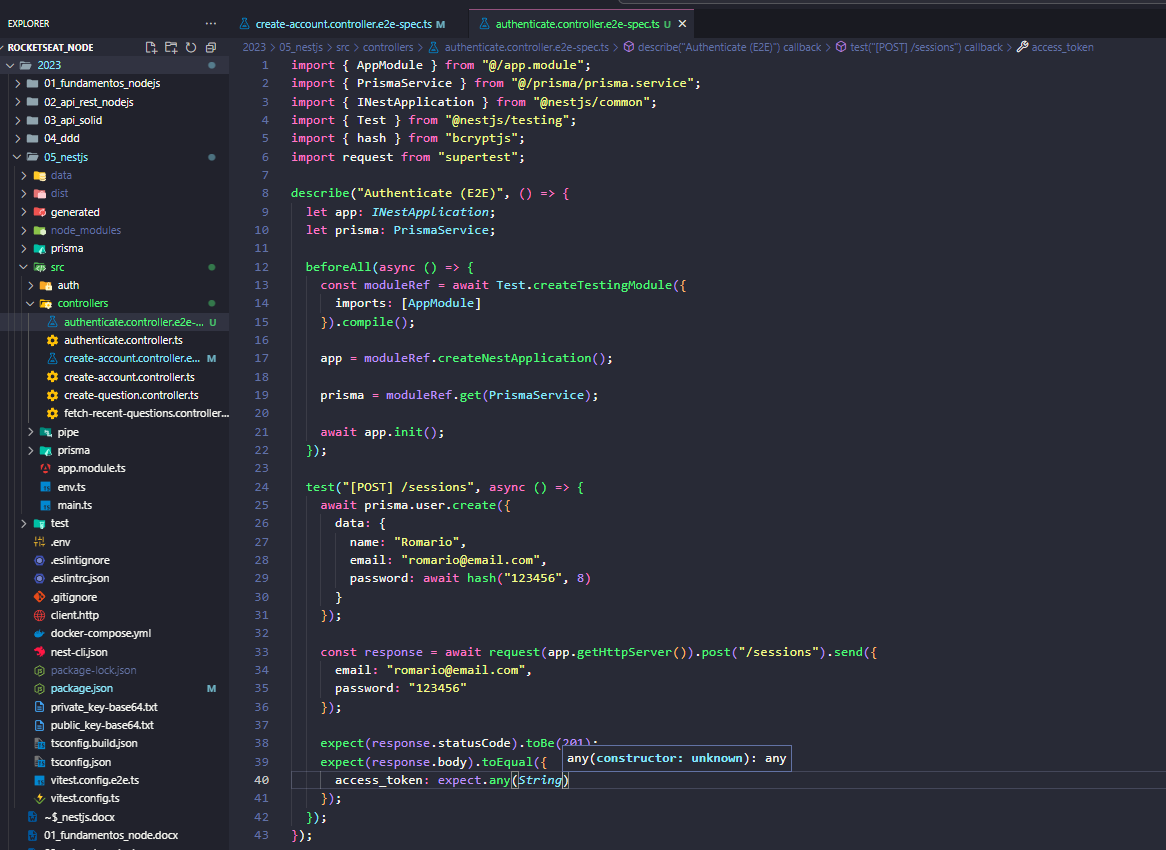




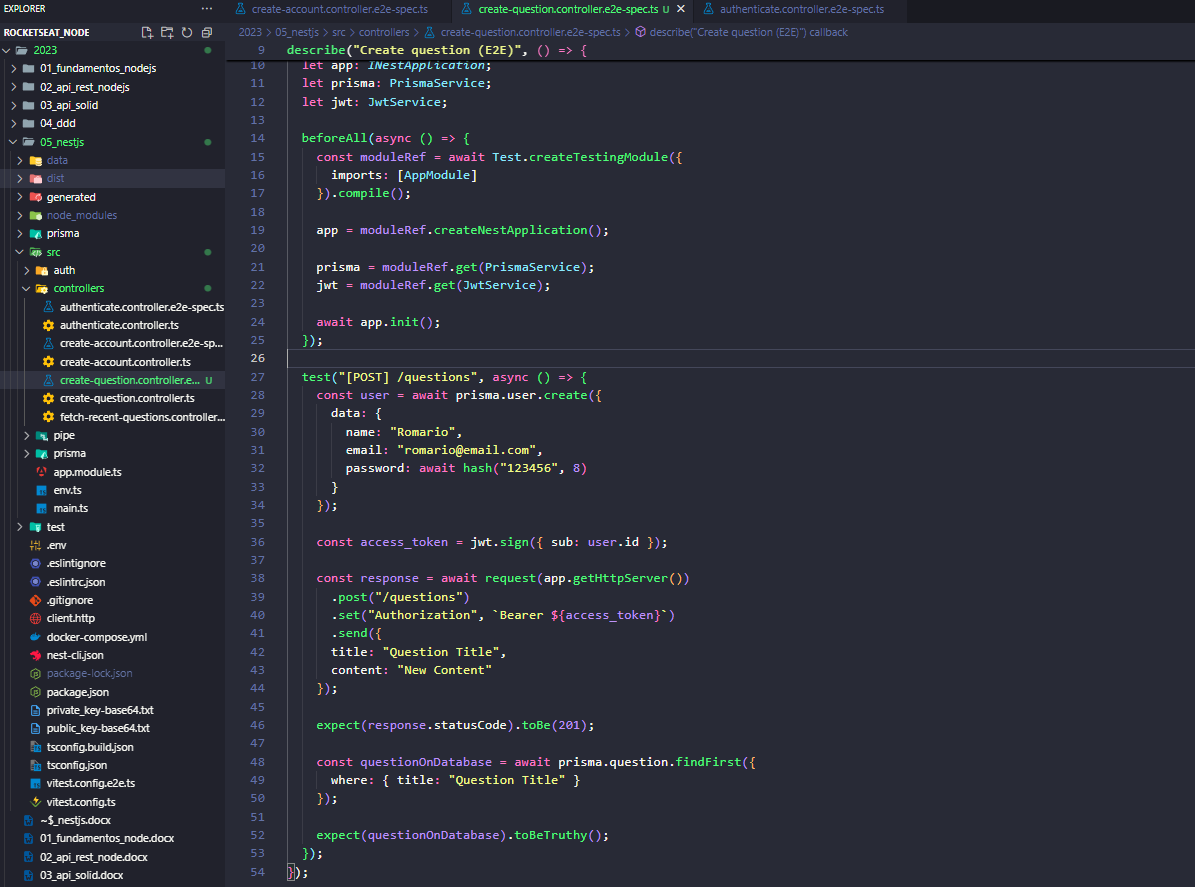
21 – Testes E2E de usuários

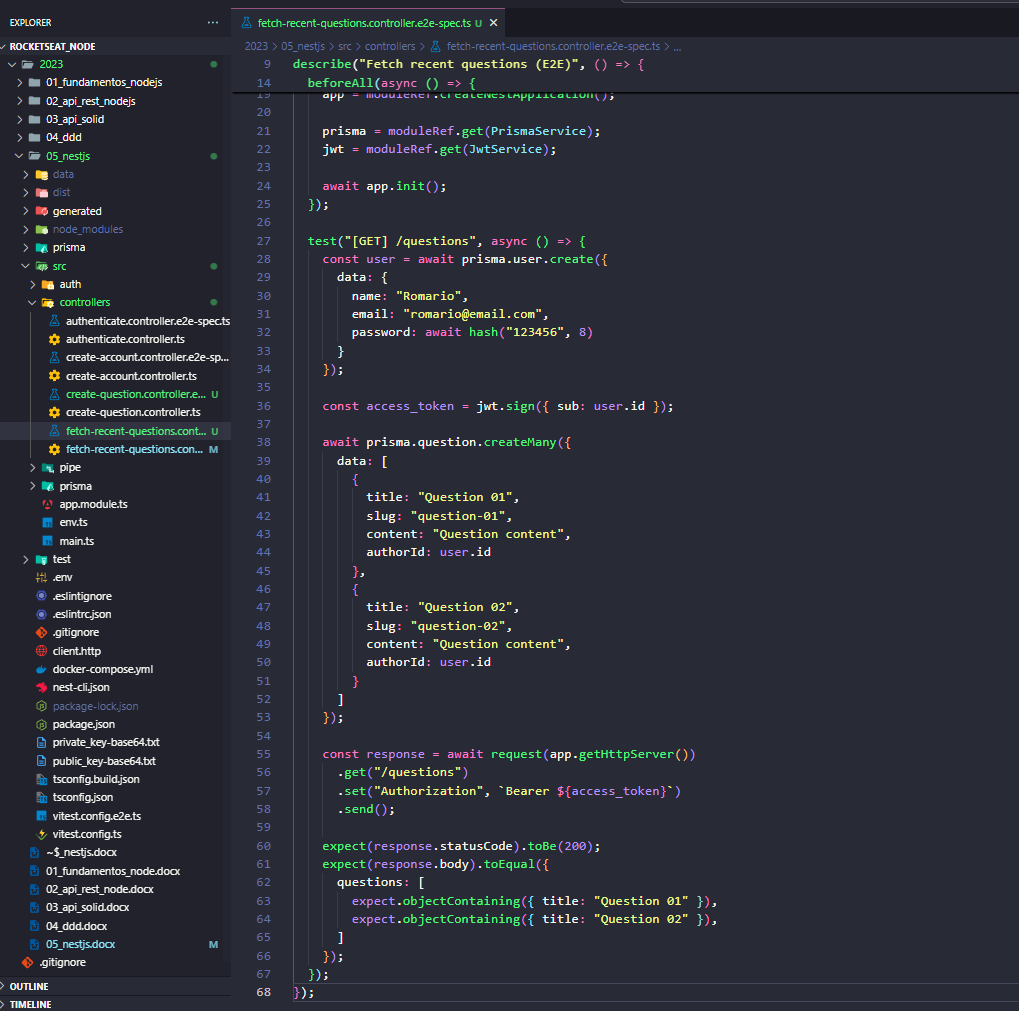
Instalar o npm i supertest -D e npm i @types/supertest -D



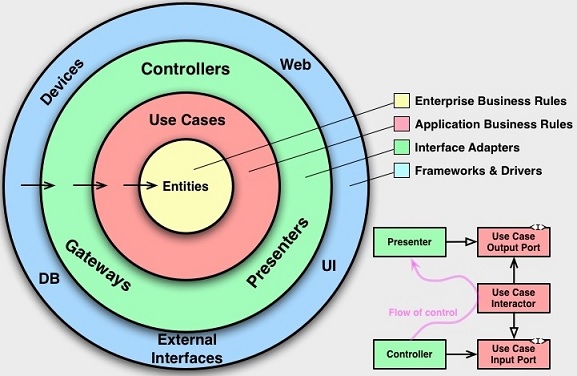


22 – Testes E2E de perguntas



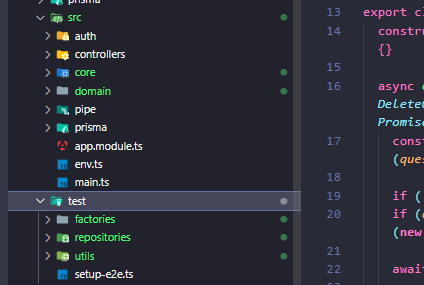


23 – Entendendo as camadas

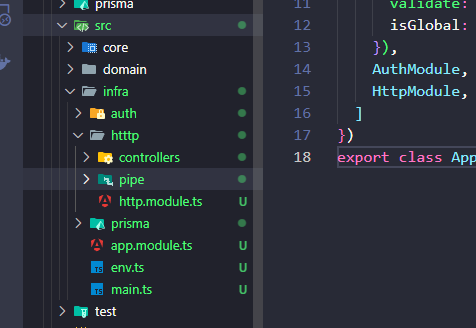


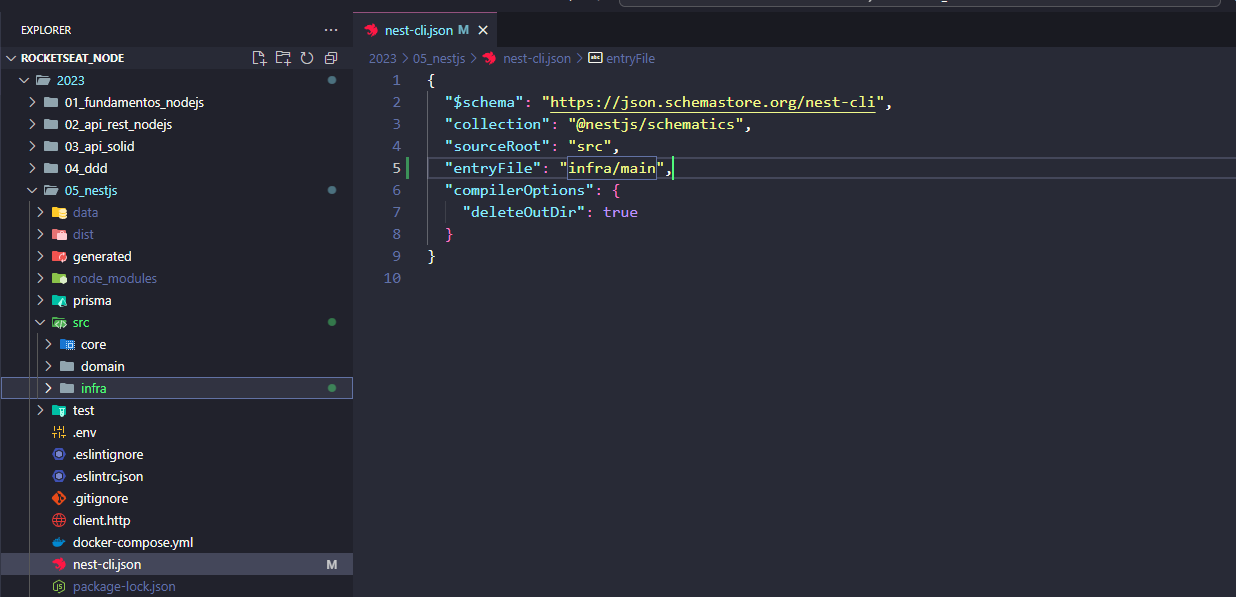
24 – Copiando camada de dompinio

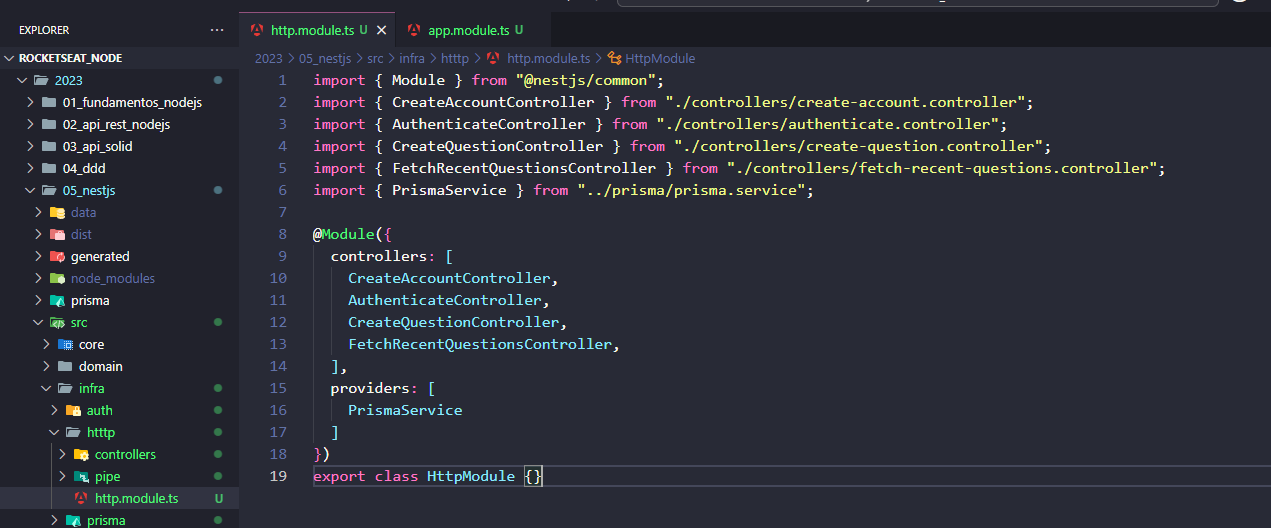
Copiar do módulo anterior os arquivos core, domain e do test factories, repositories e utils

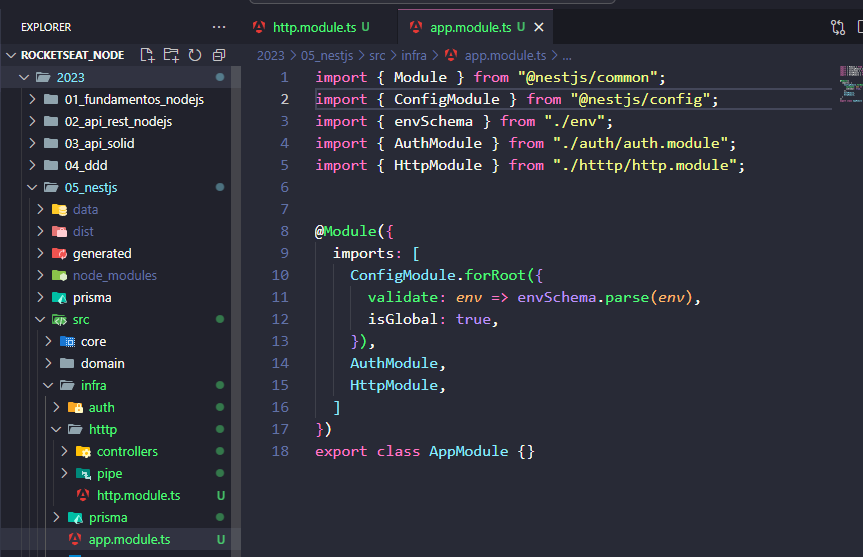


25 – Criando camada de infraestrutura

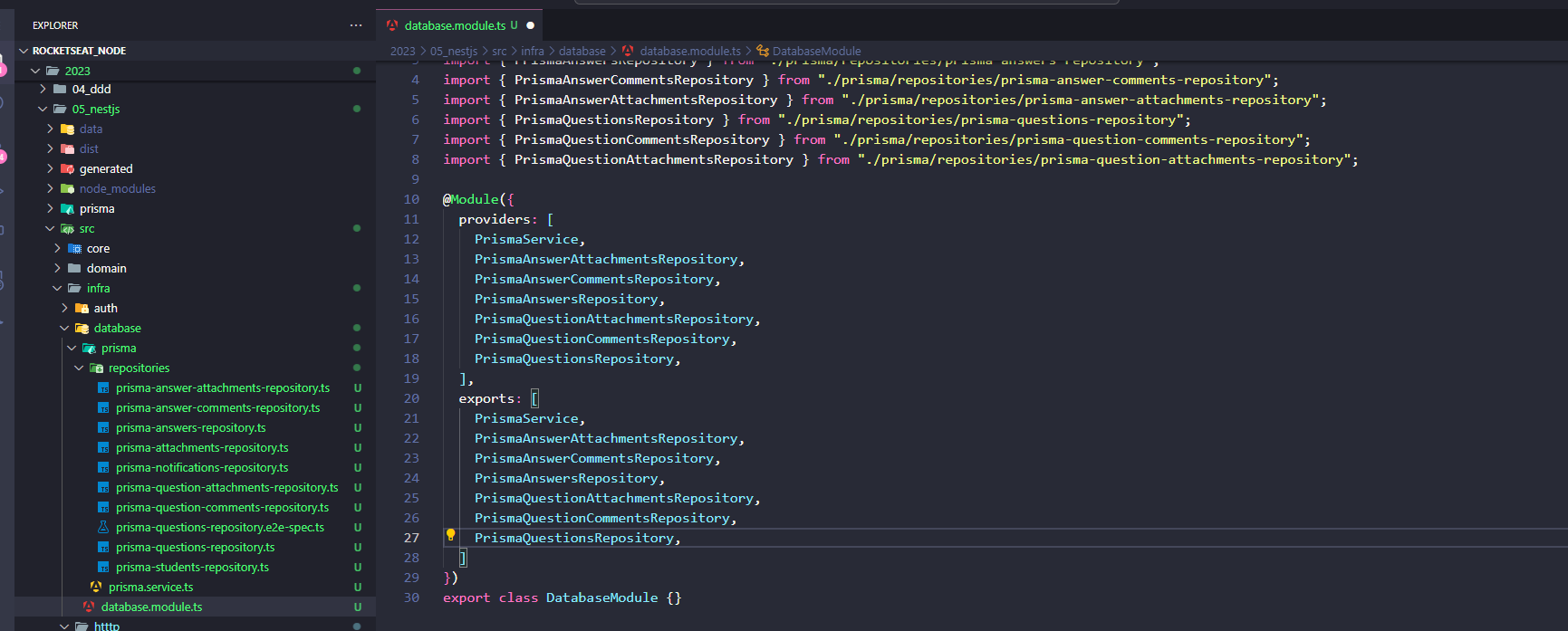


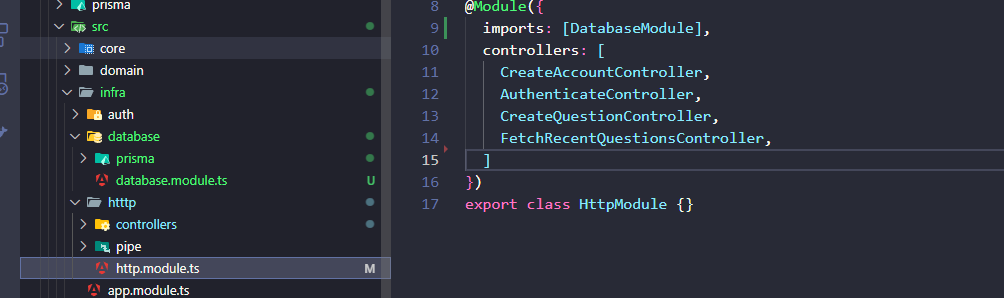


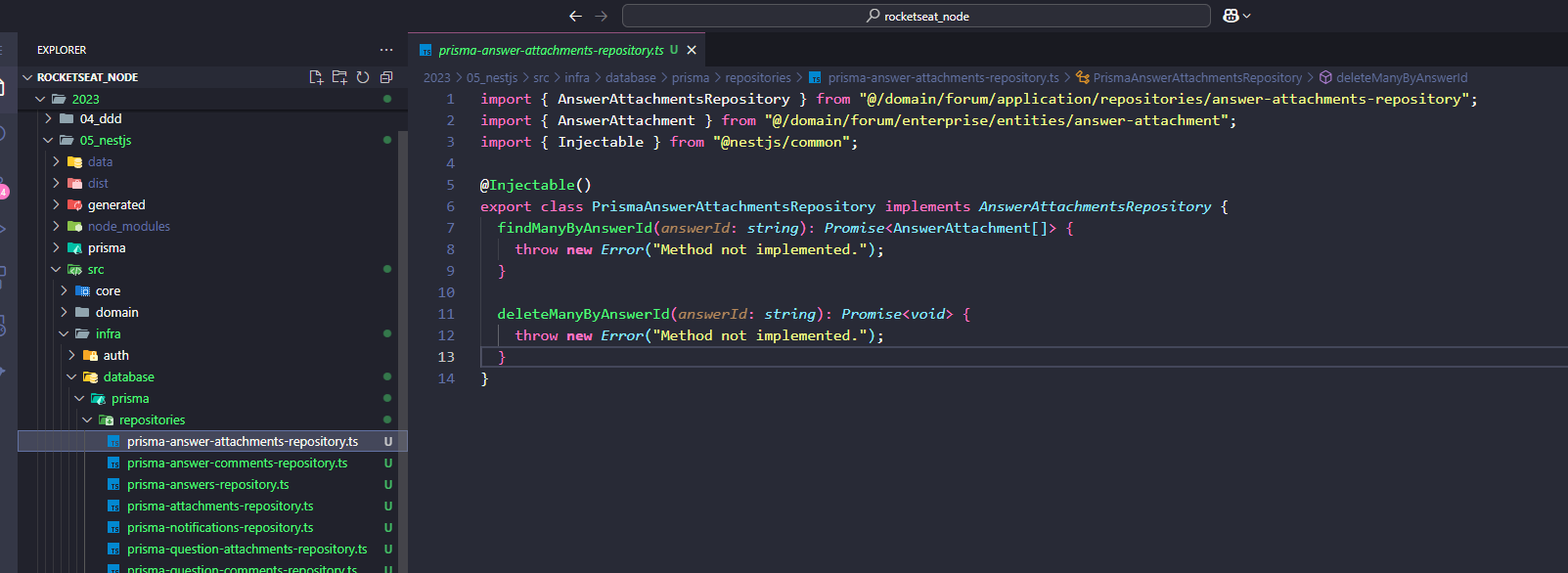




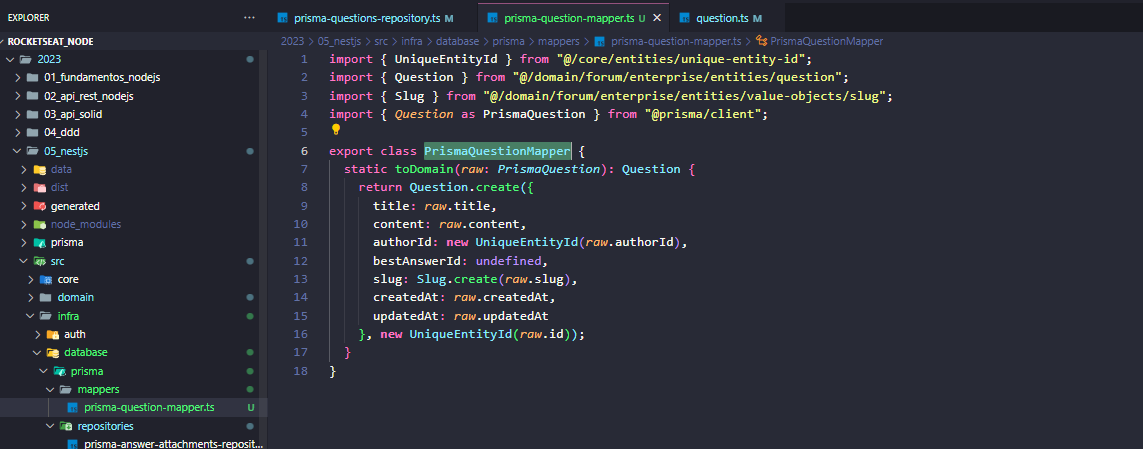
26 – Implementando repositórios do prisma

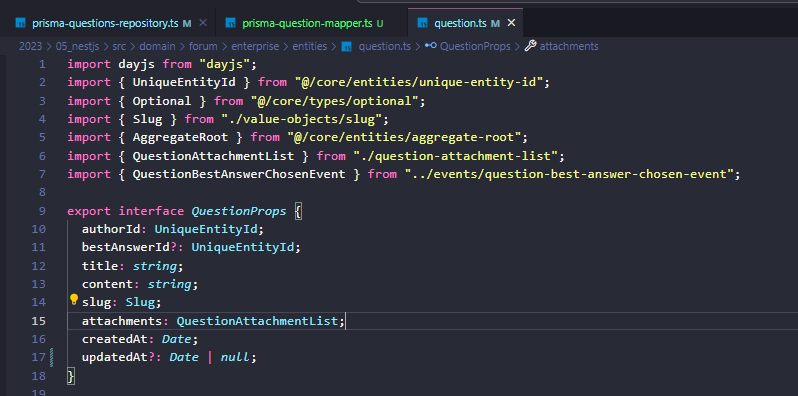


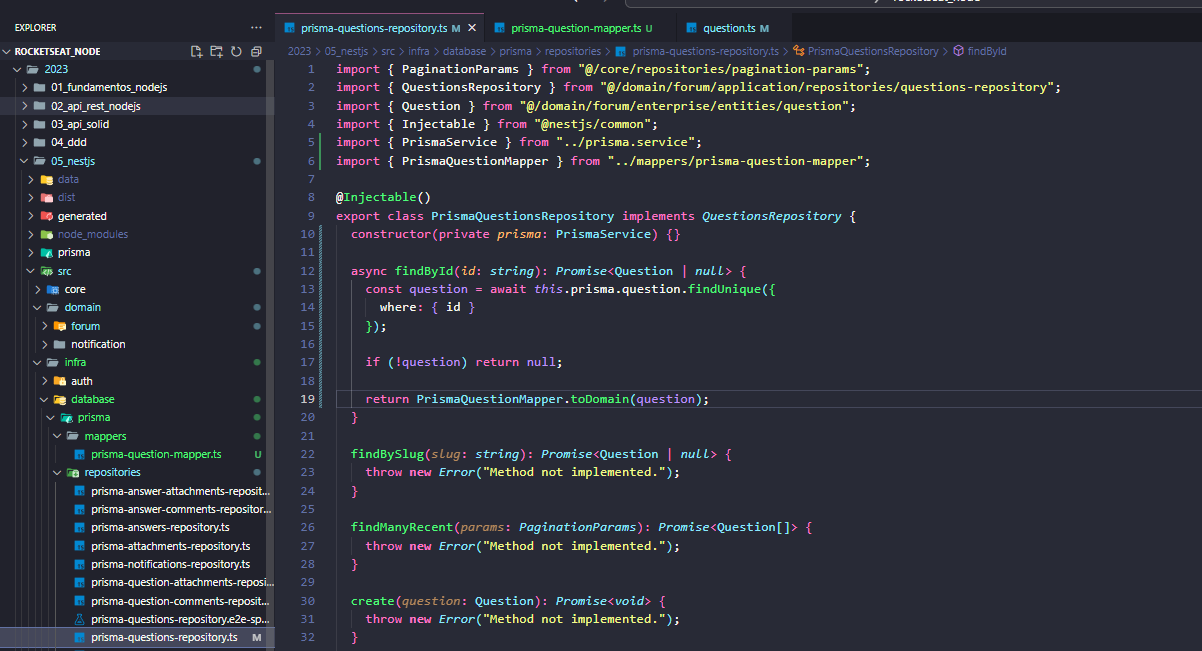




27 – Conversa entre camadas mappers







28 – Criando schema do prisma

29 – Implementando questionsrepository

30 – Comunicação entre camadas

31 –

32 –

33 –

34 –

35 –

36 –

37 –

38 –

39 –

40 –

41 -

42 -

43 -

44 -

45 -

46 -

47 -

48 -

49 -

50 -

51 -

52 -

53 -

54 -

55 -

56 -

57 -

58 -

59 -

60 -

61 -

62 -

63 -

64 -

65 -

66 -

67 -

68 -

69 -

70 -