Contesta tranquilo sin olvidarte nada. A partir del nest g res manifestaciones  
y  
*-- CreateTable*

CREATE TABLE `Manifestaciones` (

    `id` INTEGER NOT NULL AUTO\_INCREMENT,

    `e` VARCHAR(255) NULL,

    `m\_u` VARCHAR(255) NULL,

    `p` VARCHAR(255) NULL,

    `s` VARCHAR(255) NULL,

    `f\_h` DATETIME(3) NULL,

    `x` VARCHAR(255) NULL,

    `f\_i` VARCHAR(255) NULL,

    `r` VARCHAR(255) NULL,

    `i\_r` VARCHAR(191) NULL,

    `d` VARCHAR(191) NULL,

    `personalinvolucrado` VARCHAR(191) NULL,

    `email` VARCHAR(255) NULL,

    `internosinvolucrado` VARCHAR(191) NULL,

    `ims` VARCHAR(191) NULL,

    `createdAt` DATETIME(3) NOT NULL DEFAULT CURRENT\_TIMESTAMP(3),

    `updatedAt` DATETIME(3) NOT NULL,

    PRIMARY KEY (`id`)

) DEFAULT CHARACTER SET utf8mb4 COLLATE utf8mb4\_unicode\_ci;

model Manifestaciones {

  id                  Int       **@id** **@default**(autoincrement()) *// Clave primaria autoincremental*

  e                   String?   **@db.VarChar**(255) *// Campo opcional (VARCHAR(255))*

  m\_u                 String?   **@db.VarChar**(255) *// Campo opcional (VARCHAR(255))*

  p                   String?   **@db.VarChar**(255) *// Campo opcional (VARCHAR(255))*

  s                   String?   **@db.VarChar**(255) *// Campo opcional (VARCHAR(255))*

  f\_h                 DateTime? *// Campo opcional (DATETIME)*

  x                   String?   **@db.VarChar**(255) *// Campo opcional (VARCHAR(255))*

  f\_i                 String?   **@db.VarChar**(255) *// Campo opcional (VARCHAR(255))*

  r                   String?   **@db.VarChar**(255) *// Campo opcional (VARCHAR(255))*

  i\_r                 String?   *// Campo opcional (TEXT)*

  d                   String?   *// Campo opcional (TEXT)*

  personalinvolucrado String?   *// Campo opcional (LONGTEXT)*

  email               String?   **@db.VarChar**(255) *// Campo opcional (VARCHAR(255))*

  internosinvolucrado String?   *// Campo opcional (LONGTEXT)*

  ims                 String?   *// Campo opcional (LONGTEXT)*

  createdAt           DateTime  **@default**(now())  *// TIMESTAMP con valor por defecto*

  updatedAt           DateTime  **@updatedAt**      *// TIMESTAMP con actualización automática*

}

Necesito que uses toda la misma lógica de aquí de bajo que se usó en products, para agregar una nueva “nest g res manifestaciones”, no tienen que modificar la lógica, solo los nombre, porque ahora en vez de products usaremos manifestaciones, en vez de Products usaremos Manifestaciones, en vez de product usaremos manifestacion, en vez de products usaremos manifestaciones

De tal modo que todo el proyecto quedara con 2 nest generate resource   
//src\app.module.ts  
import { Module } from '@nestjs/common';

import { ProductsModule } from './products/products.module';

import { PrismaService } from './prisma/prisma.service';

import { ImpactosModule } from './impactos/impactos.module';

@**Module**({

  imports: [ProductsModule, ImpactosModule],

  controllers: [],

  providers: [PrismaService],

})

export class AppModule {}

//src\main.ts  
import { NestFactory } from '@nestjs/core';

import { AppModule } from './app.module';

import { SwaggerModule, DocumentBuilder } from '@nestjs/swagger';

async function **bootstrap**() {

  const app = await NestFactory.**create**(AppModule);

  app.**setGlobalPrefix**('api');

  const config = new **DocumentBuilder**()

    .**setTitle**('Cats example')

    .**setDescription**('The cats API description')

    .**setVersion**('1.0')

    .**addTag**('cats')

    .**build**();

  const document = SwaggerModule.**createDocument**(app, config);

  SwaggerModule.**setup**('api', app, document);

  app.**enableCors**();

  await app.**listen**(4000);

}

**bootstrap**();

//tsconfig.build.json  
{

  "extends": "./tsconfig.json",

  "exclude": ["node\_modules", "test", "dist", "\*\*/\*spec.ts"]

}

//tsconfig.json  
{

  "compilerOptions": {

    "module": "commonjs",

    "declaration": true,

    "removeComments": true,

    "emitDecoratorMetadata": true,

    "experimentalDecorators": true,

    "allowSyntheticDefaultImports": true,

    "target": "ES2021",

    "sourceMap": true,

    "outDir": "./dist",

    "baseUrl": "./",

    "incremental": true,

    "skipLibCheck": true,

    "strictNullChecks": false,

    "noImplicitAny": false,

    "strictBindCallApply": false,

    "forceConsistentCasingInFileNames": false,

    "noFallthroughCasesInSwitch": false

  }

}

nest g res products

//prisma\schema.prisma

model Product {

  id          Int      **@id** **@default**(autoincrement())

  name        String   **@unique**

  description String?

  price       Float

  image       String?

  createdAt   DateTime **@default**(now())

  updatedAt   DateTime **@updatedAt**

}

//src\products\dto\create-product.dto.ts  
import { Product } from '@prisma/client';

export type CreateProductDto = Omit<Product, 'id' | 'createdAt' | 'updatedAt'>;

//src\products\dto\update-product.dto.ts  
import { CreateProductDto } from './create-product.dto';

export type UpdateProductDto = Partial<CreateProductDto>;

//src\products\entities\product.entity.ts  
export class Product {}

//src\products\products.controller.spec.ts  
import { Test, TestingModule } from '@nestjs/testing';

import { ProductsController } from './products.controller';

import { ProductsService } from './products.service';

**describe**('ProductsController', () => {

  let controller: ProductsController;

**beforeEach**(async () => {

    const module: TestingModule = await Test.**createTestingModule**({

      controllers: [ProductsController],

      providers: [ProductsService],

    }).**compile**();

    controller = module.**get**<ProductsController>(ProductsController);

  });

**it**('should be defined', () => {

**expect**(controller).**toBeDefined**();

  });

});

//src\products\products.controller.ts  
import {

  Controller,

  Get,

  Post,

  Body,

  Patch,

  Param,

  Delete,

} from '@nestjs/common';

import { ProductsService } from './products.service';

import { CreateProductDto } from './dto/create-product.dto';

import { UpdateProductDto } from './dto/update-product.dto';

import { ApiOperation, ApiResponse, ApiTags } from '@nestjs/swagger';

@**ApiTags**('products')

@**Controller**('products')

export class ProductsController {

  constructor(private readonly *productsService*: ProductsService) {}

  @**Post**()

  @**ApiOperation**({ summary: 'Create a product' })

**create**(@**Body**() *createProductDto*: CreateProductDto) {

    return this.productsService.**create**(*createProductDto*);

  }

  @**Get**()

  @**ApiResponse**({ status: 200, description: 'Return all products' })

**findAll**() {

    return this.productsService.**findAll**();

  }

  @**Get**(':id')

**findOne**(@**Param**('id') *id*: string) {

    return this.productsService.**findOne**(+*id*);

  }

  @**Patch**(':id')

**update**(@**Param**('id') *id*: string, @**Body**() *updateProductDto*: UpdateProductDto) {

    return this.productsService.**update**(+*id*, *updateProductDto*);

  }

  @**Delete**(':id')

**remove**(@**Param**('id') *id*: string) {

    return this.productsService.**remove**(+*id*);

  }

}

//src\products\products.module.ts  
import { Module } from '@nestjs/common';

import { ProductsService } from './products.service';

import { ProductsController } from './products.controller';

import { PrismaService } from 'src/prisma/prisma.service';

@**Module**({

  controllers: [ProductsController],

  providers: [ProductsService, PrismaService],

})

export class ProductsModule {}

//src\products\products.service.spec.ts  
import { Test, TestingModule } from '@nestjs/testing';

import { ProductsService } from './products.service';

**describe**('ProductsService', () => {

  let service: ProductsService;

**beforeEach**(async () => {

    const module: TestingModule = await Test.**createTestingModule**({

      providers: [ProductsService],

    }).**compile**();

    service = module.**get**<ProductsService>(ProductsService);

  });

**it**('should be defined', () => {

**expect**(service).**toBeDefined**();

  });

});

//src\products\products.service.ts  
import {

  Injectable,

  NotFoundException,

  ConflictException,

  InternalServerErrorException,

} from '@nestjs/common';

import { CreateProductDto } from './dto/create-product.dto';

import { UpdateProductDto } from './dto/update-product.dto';

import { PrismaService } from 'src/prisma/prisma.service';

import { Prisma } from '@prisma/client';

@**Injectable**()

export class ProductsService {

  constructor(private *prismaService*: PrismaService) {}

  async **create**(*createProductDto*: CreateProductDto) {

    try {

      return await this.prismaService.product.**create**({

        data: *createProductDto*,

      });

    } catch (error) {

      if (error instanceof Prisma.PrismaClientKnownRequestError) {

        if (error.code === 'P2002') {

          throw new **ConflictException**(

            `Product with name ${*createProductDto*.name} already exists`,

          );

        }

      }

      throw new **InternalServerErrorException**();

    }

  }

**findAll**() {

    return this.prismaService.product.**findMany**();

  }

  async **findOne**(*id*: number) {

    const productFound = await this.prismaService.product.**findUnique**({

      where: {

        id: *id*,

      },

    });

    if (!productFound) {

      throw new **NotFoundException**(`Product with id ${*id*} not found`);

    }

    return productFound;

  }

  async **update**(*id*: number, *updateProductDto*: UpdateProductDto) {

    const productFound = await this.prismaService.product.**update**({

      where: {

        id,

      },

      data: *updateProductDto*,

    });

    if (!productFound) {

      throw new **NotFoundException**(`Product with id ${*id*} not found`);

    }

    return productFound;

  }

  async **remove**(*id*: number) {

    const deletedProduct = await this.prismaService.product.**delete**({

      where: {

        id,

      },

    });

    if (!deletedProduct) {

      throw new **NotFoundException**(`Product with id ${*id*} not found`);

    }

    return deletedProduct;

  }

}