**Controller code backcend**

**import { Controller, Get, Post, Put, Delete, Body, Param } from '@nestjs/common';**

**import { ProductsService } from './products.service'; // Correct import based on your service name**

**import { Product } from './products.entity';**

**@Controller('products')**

**export class ProductsController {**

**constructor(private readonly productService: ProductsService) {} // Ensure it's ProductsService**

**@Get()**

**async getAllProducts(): Promise<Product[]> {**

**return this.productService.getAllProducts();**

**}**

**// Add this new endpoint for fetching a product by ID**

**@Get(':id')**

**async getProductById(@Param('id') id: number): Promise<Product> {**

**return this.productService.getProductById(id); // Ensure this service method is defined**

**}**

**@Post()**

**async createProduct(@Body() productData: Partial<Product>): Promise<Product> {**

**return this.productService.createProduct(productData);**

**}**

**@Put(':id')**

**async updateProduct(@Param('id') id: number, @Body() productData: Partial<Product>): Promise<Product> {**

**return this.productService.updateProduct(id, productData);**

**}**

**@Delete(':id')**

**async deleteProduct(@Param('id') id: number): Promise<void> {**

**return this.productService.deleteProduct(id);**

**}**

**}**

**Entity code**

**import { Entity, PrimaryGeneratedColumn, Column } from 'typeorm';**

**@Entity('product')**

**export class Product {**

**@PrimaryGeneratedColumn()**

**id: number;**

**@Column()**

**name: string;**

**@Column()**

**description: string;**

**@Column()**

**category: string;**

**@Column('decimal')**

**price: number;**

**@Column()**

**image\_url: string;**

**}**

**Module code**

**import { Module } from '@nestjs/common';**

**import { TypeOrmModule } from '@nestjs/typeorm';**

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

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

**import { Product } from './products.entity';**

**@Module({**

**imports: [TypeOrmModule.forFeature([Product])],**

**controllers: [ProductsController],**

**providers: [ProductsService]**

**})**

**export class ProductsModule {}**

**Service code**

**import { Injectable } from '@nestjs/common';**

**import { InjectRepository } from '@nestjs/typeorm';**

**import { Repository } from 'typeorm';**

**import { Product } from './products.entity';**

**@Injectable()**

**export class ProductsService {**

**constructor(**

**@InjectRepository(Product)**

**private readonly productRepository: Repository<Product>,**

**) {}**

**// Get all products**

**async getAllProducts(): Promise<Product[]> {**

**try {**

**return await this.productRepository.find();**

**} catch (error) {**

**throw new Error('Error fetching products');**

**}**

**}**

**async getProductById(id: number): Promise<Product> {**

**const product = await this.productRepository.findOne({ where: { id } });**

**if (!product) {**

**throw new Error('Product not found'); // Handle not found error**

**}**

**return product;**

**}**

**// Create a new product**

**async createProduct(productData: Partial<Product>): Promise<Product> {**

**try {**

**const product = this.productRepository.create(productData); // Create a product object**

**return await this.productRepository.save(product); // Save the new product to DB**

**} catch (error) {**

**throw new Error('Error creating product');**

**}**

**}**

**async updateProduct(id: number, productData: Partial<Product>): Promise<Product> {**

**try {**

**const product = await this.productRepository.findOne({ where: { id } });**

**if (!product) {**

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

**}**

**await this.productRepository.update(id, productData);**

**return await this.productRepository.findOne({ where: { id } }); // Return updated product**

**} catch (error) {**

**console.error("Error in updating product:", error);**

**throw new Error('Error updating product');**

**}**

**}**

**// Delete a product**

**async deleteProduct(id: number): Promise<void> {**

**try {**

**await this.productRepository.delete(id); // Delete the product by ID**

**} catch (error) {**

**throw new Error('Error deleting product');**

**}**

**}**

**}**

**App.module**

**import { Module } from '@nestjs/common';**

**import { TypeOrmModule } from '@nestjs/typeorm';**

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

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

**import { Product } from './products/products.entity';**

**import { Order } from './order/order.entity';**

**@Module({**

**imports: [**

**TypeOrmModule.forRoot({**

**type: 'mysql',**

**host: 'localhost', // Database host (adjust as needed)**

**port: 3306, // MySQL default port**

**username: 'root', // MySQL username**

**password: '', // MySQL password**

**database: 'flower', // Name of your database**

**entities: [Product,Order], // Include all your entities here**

**synchronize: true, // Set to true for automatic table creation (only in development)**

**// Optional: Enables logging of queries**

**}),**

**TypeOrmModule.forFeature([Product]),**

**// Inject Product repository**

**],**

**controllers: [ProductsController],**

**providers: [ProductsService],**

**})**

**export class AppModule {}**