nest generate controller –dry-run

nest g co

nest g module coffees

@Params() params

@Params(‘id’) id:string

@Body() body

@Body(‘name’) name

@HttpCode(HttpStatus.GONE)

As we are using express, we can get the response object as follows @Res() response; using this to send data, we loose the nestjs features like interceptors and http code decorators; Use nest standard response instead of this

PUT (SUPPOSE TO) replaces the entire resource and PATCH (SUPPOSE TO) replace only partially of the resource

Normally we use path parameters to identify specific resource and query parameters for filtration and sort

@Query() paginationQuery

Throw new HttpException(`Not Found`, HttpsStatus.NOT\_FOUND)

Throw new NotFoundException(‘Not found’); // automatically handles the response code and error like {statusCode: 404, “message”: “Coffee not found”, “error”: “Not Found”}

Use DTO’s for transferring data, nest g class coffees/sto/create-coffee.dto –no-spec

Its good practice to mark all properties in DTO readonly

Make different dtos for create, update, patch etc; reuse the properties if needed from already made dtos

To use validators and transofrmers, install class-validators and class-transformer package and register the validators in main.ts as app.useGlobalPiper(new ValidationPipe()); After adding this you can directly import the validators in DTO’s and use them there

We can reuse already made DTO’s using mappedTypes package. Install the mapped Type package and use already made DTO’s as export class UpdateCoffeeDTO extends PartialType(CreateCoffeeDTO){}; This inherits all the validation rules of CreateCoffeeDTO and also applied additional rule of optional (@IsOptional()) on all the inherited fields

With parameters to ValidationPipe({whitelist:true}) we can strip all the extra properties that were sent in the request, we can also add ValidationPipe({whitelist:true, forbidWhitelisted:true}) to return error if extra properties are sent in requests

If we pass forbidNonWhitelisted: true in validation pipe, it would give an error if extra properties are sent in the request

Transform:true passing in the validation pipe help transform the objects that are sent in the request. By Default all data is sent in a request as JSON, passing transform flag will enable validation pipe to transform JSON data into required type (or primitive type) mentioned in the controller. This helps us to transform the body, params etc into actual types that are mentioned at the controller level.