# Primer paso:

Instalar @nestjs

npm install -g @nestjs/cli

Crear Proyecto ex02

nest new ex02

Objetivo crear los endpoints para realizar las operaciones CRUD de los datos de gatos:

|  |  |
| --- | --- |
| GET | api/v0/cat/read/:id |
| DELETE | api/v0/cat /delete/:id |
| GET | api/v0/cat /readAll |
| POST | api/v0/cat /create |

Datos que se almacenan de un gato:

* Name
* Age
* Breed

cd ex02

npm install --save mongoose

npm install --save-dev @types/mongoose

La aplicación se organiza en 2 módulos

## Database module

nest g mo database

nest g class database/databaseProviders

## Cats module

nest g mo cats

El módulo cats tiene un controlador, un servicio, una interface y tres clases

nest g service cats/services/cats

nest g controller cats/api/cats

nest g class cats/schema/catSchema

nest g class cats/cats.provider

nest g class cats/interfaces/createCatDto

nest g interface cats/interfaces/cat

# Código

**Contenido del fichero src\database\database-providers.ts**

import \* as mongoose from 'mongoose';

export const DatabaseProviders = [

  {

    provide: 'DATABASE\_CONNECTION',

    useFactory: (): Promise<typeof mongoose> =>

      mongoose.connect("mongodb://localhost/animals")

  },

];

**Fichero src\database\database.module.ts**

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

import { DatabaseProviders } from './database.providers';

@Module({

  providers: [...DatabaseProviders],

  exports: [...DatabaseProviders],

})

export class DatabaseModule {}

**Fichero src\cats\interfaces\cat.interface.ts**

import { Document } from 'mongoose';

export interface Cat extends Document {

  readonly name: string;

  readonly age: number;

  readonly breed: string;

}

**Fichero src\cats\schema\cat-schema**

import \* as mongoose from 'mongoose';

export const CatSchema = new mongoose.Schema({

    name: String,

    age: Number,

    breed: String,

  });

**Fichero con el objeto DTO (Data transfer Object) src\cats\interfaces\create-cat-dto.ts**

export class CreateCatDto {

    name: string;

    age: number;

    breed: string;

}

**Fichero con el servicio: src\cats\services\cats\cats.service.ts**

import { Injectable, Inject } from '@nestjs/common';

import { Model } from 'mongoose';

import { Cat } from 'src/cats/interfaces/cat.interface';

import { CreateCatDto } from 'src/cats/interfaces/create-cat-dto';

@Injectable()

export class CatsService {

    constructor(

        @Inject('CAT\_MODEL')

        private readonly catModel: Model<Cat>,

      ) {}

      async create(createCatDto: CreateCatDto): Promise<Cat> {

        const createdCat = new this.catModel(createCatDto);

        return createdCat.save();

      }

      async findAll(): Promise<Cat[]> {

        return this.catModel.find().exec();

      }

      async delete(id: string){

        return this.catModel.deleteOne({ \_id: id})

      }

}

**Fichero src\cats\cats.provider.ts**

import { Connection } from 'mongoose';

import { CatSchema } from './schema/cat-schema';

export const catsProviders = [

  {

    provide: 'CAT\_MODEL',

    useFactory: (connection: Connection) => connection.model('Cat', CatSchema),

    inject: ['DATABASE\_CONNECTION'],

  },

];

**Fichero src\cats\cats.module.ts**

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

import { CatsController } from './api/cats/cats.controller';

import { CatsService } from './services/cats/cats.service';

import { DatabaseModule } from 'src/database/database.module';

import { catsProviders } from './cats.provider';

@Module({

  imports: [DatabaseModule],

  controllers: [CatsController],

  providers: [

    CatsService,

    ...catsProviders

  ]

})

export class CatsModule {}

**Fichero src\cats\api\cats\cats.controller.ts**

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

import { CatsService } from 'src/cats/services/cats/cats.service';

import { CreateCatDto } from '../../types/create-cat-dto'

@Controller('api/cats/v0')

export class CatsController {

    constructor( private catsService: CatsService ) {}

    @Post('cat/create')

    catCreate(@Body()catDetails){

        this.catsService.create(catDetails);

        return { msg: 'hello' }

    }

    @Get('cat/readAll')

    readAll(){

        return this.catsService.findAll();

    }

    @Delete('cat/delete/:id')

    deleteCat(@Param('id') id: string){

        console.log('Borrar a '+ id);

        return this.catsService.delete(id);

    }

}