Les **controller** sont chargés de traiter les demandes entrantes et de renvoyer les réponses au client.

```ts
import { Body, Controller, Delete, Get, Param, Post, Put, Query } from '@nestjs/common';
import { CarService } from './car.service';
import { CarDto } from './car.dto';
import { query } from 'express';


@Controller('car')
export class CarController {
    constructor (private carService: CarService) {}

    @Get()
    async getCars() {
        return this.carService.getCars();
    }

    @Post()
    public postCar(@Body() car: CarDto) {
        return this.carService.postCar(car);
    }

    @Get(':id')
    public async getCarById(@Param('id') id: number) {
        const result = await this.carService.getCarById(id);
        return result;
    }

    @Delete(':id')
    public async deleteCarById(@Param('id') id: number) {
        return this.carService.deleteCarById(id);
    }

    @Put(':id')
    public async putCarById(@Param('id') id: number, @Query() query) {
        const propertyName = query.property_name;
        const propertyValue = query.property_value;
        return this.carService.putCarById(id, propertyName, propertyValue);
    }
}
```

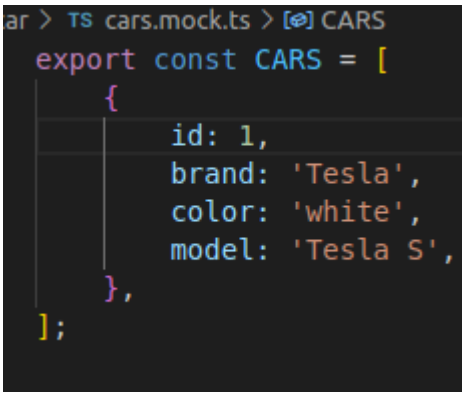
Les **controller** sont chargés d'accepter les requêtes HTTP du client et de fournir une réponse.
Ils appellent du code situé dans la partie service. La partie **service** est chargée de fournir certaines données, qui peuvent être réutilisées dans l'ensemble de l'application.

```
@Injectable()
export class CarService {
    private cars = CARS;

    public async getCars() {
        return this.cars;
    }

    public async postCar(car) {
        return this.cars.push(car);
    }

    public async getCarById(id: number): Promise<any> {
        const carId = Number(id);
        return new Promise((resolve) => {
            const car = this.cars.find((car) => car.id === carId);
            if (!car) {
                throw new HttpException('Not found', 404);
            }
            return resolve(car);
        });
    }

    public async deleteCarById(id): Promise<any> {
        const carId = Number(id);
        return new Promise((resolve) => {
            const index = this.cars.findIndex(car => car.id === carId);
            if (index === -1) {
                throw new HttpException('Not found', 404);
            }
            this.cars.splice(index, 1);
            return resolve(this.cars);
        });
    }

    public async putCarById(id: number, propertyName: string, propertyValue: string): Promise<any> {
        const carId = Number(id);
        return new Promise((resolve) => {
            const index = this.cars.findIndex(car => car.id === carId);
            if (index === -1) {
                throw new HttpException('Not found', 404);
            }
            this.cars[index][propertyName] = propertyValue;
```

Le **Module** fournit des métadonnées que Nest utilise pour organiser la structure de l'application.

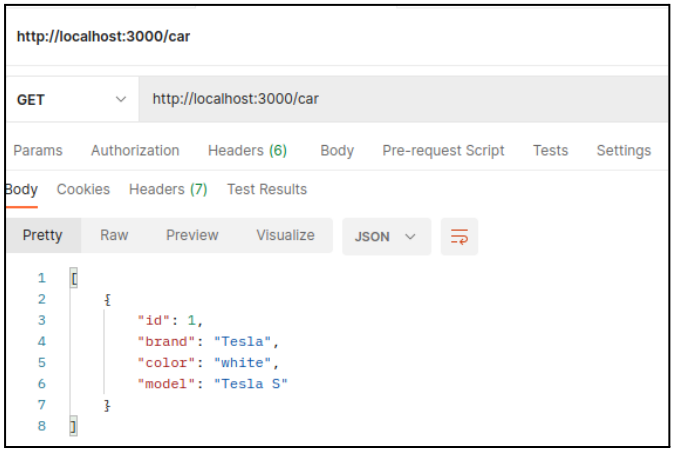
```ts
import { Module } from '@nestjs/common';
import { MongooseModule } from '@nestjs/mongoose'
import { CarController } from './car.controller';
import { CarService } from './car.service';
import { CarSchema } from './schemas/car.schema'

@Module({
  imports: [
    MongooseModule.forFeature([
      {
        name: 'Car',
        schema: CarSchema,
      },
    ]),
  ],
  controllers: [CarController],
  providers: [CarService],
})
export class CarModule {}
```

Un premier objet est initialisé dans la base de données.

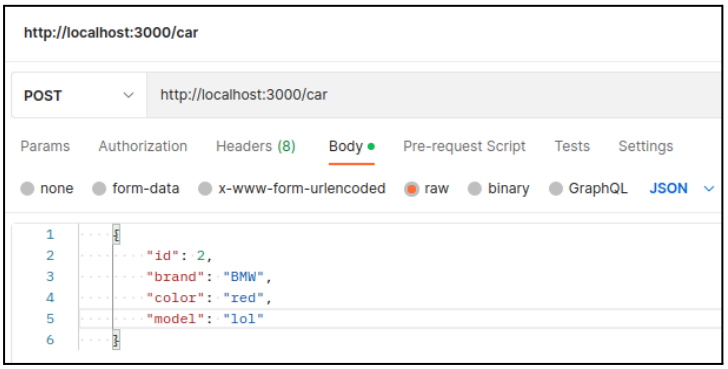


Voici des requêtes réalisées avec Postman:

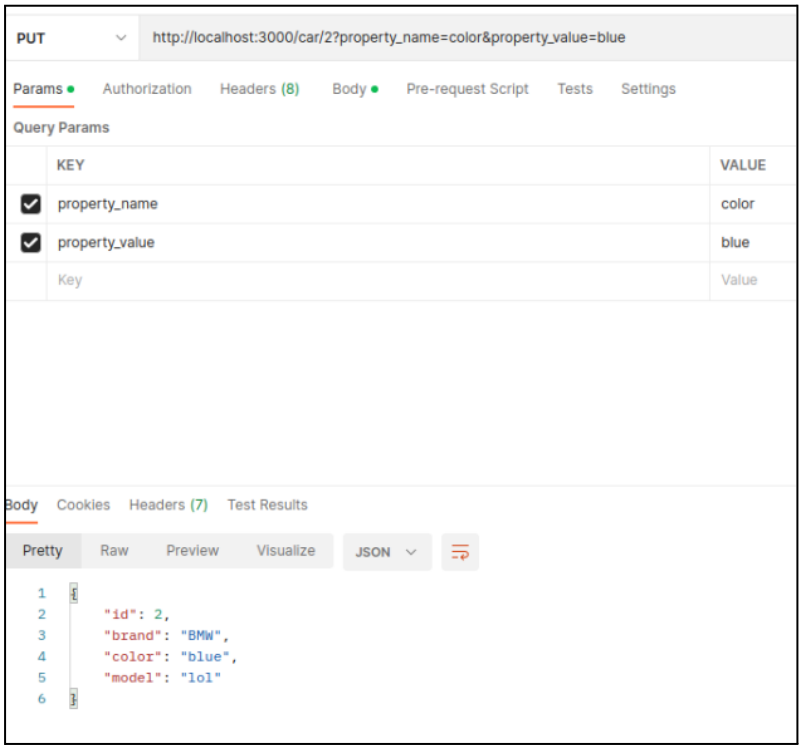
get



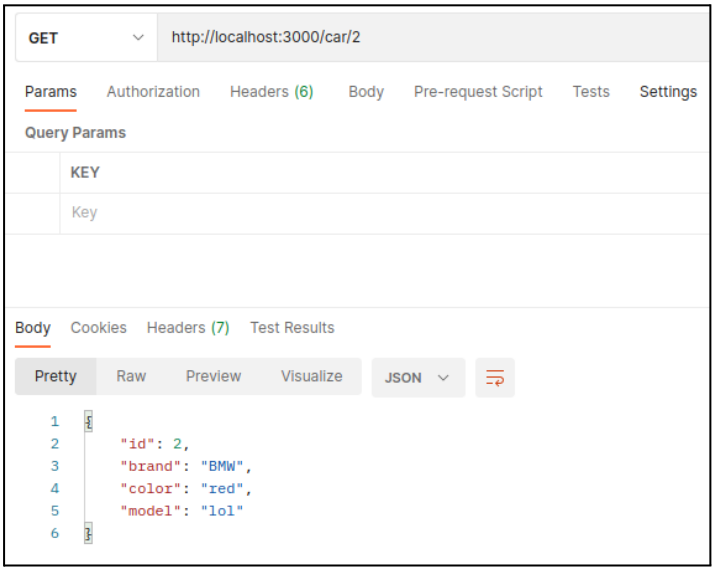
post



put



get sur un élément précis



delete