

Laporan Proyek Akhir

– **Praktikum Basis Data Lanjut**

MEMBUAT API MENGGUNAKAN NODE.JS DENGAN FRAMEWORK EXPRESS

Oleh :

Muhammad Arief Satria Wibawa (3122600015)



— Inisialisasi dengan npm init

```
PS D:\matkul\sem 3\Basdat Lanjut\Project\car-data> npm init
This utility will walk you through creating a package.json file.
It only covers the most common items, and tries to guess sensible defaults.
```

See 'npm help init' for definitive documentation on these fields and exactly what they do.

Use 'npm install <pkg>' afterwards to install a package and save it as a dependency in the package.json file.

Press ^C at any time to quit.

package name: (car-data)

version: (1.0.0)

description: Express API

entry point: (index.js) server.js

test command:

git repository: github.com/illufoxKusanagi/example-app-API

keywords:

author: muhammadArief

license: (ISC)

About to write to D:\matkul\sem 3\Basdat Lanjut\Project\car-data\package.json:

```
{
  "name": "car-data",
  "version": "1.0.0",
  "description": "Express API",
  "main": "server.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "repository": {
    "type": "git",
    "url": "github.com/illufoxKusanagi/example-app-API"
  },
  "author": "muhammadArief",
  "license": "ISC"
}
```

Is this OK? (yes) yes

```
npm notice
npm notice New major version of npm available! 8.19.2 -> 10.2.4
npm notice Changelog: https://github.com/npm/cli/releases/tag/v10.2.4
npm notice Run npm install -g npm@10.2.4 to update!
npm notice
```



Menginstall nodemon supaya tidak perlu mematikan server ketika ingin melakukan perubahan

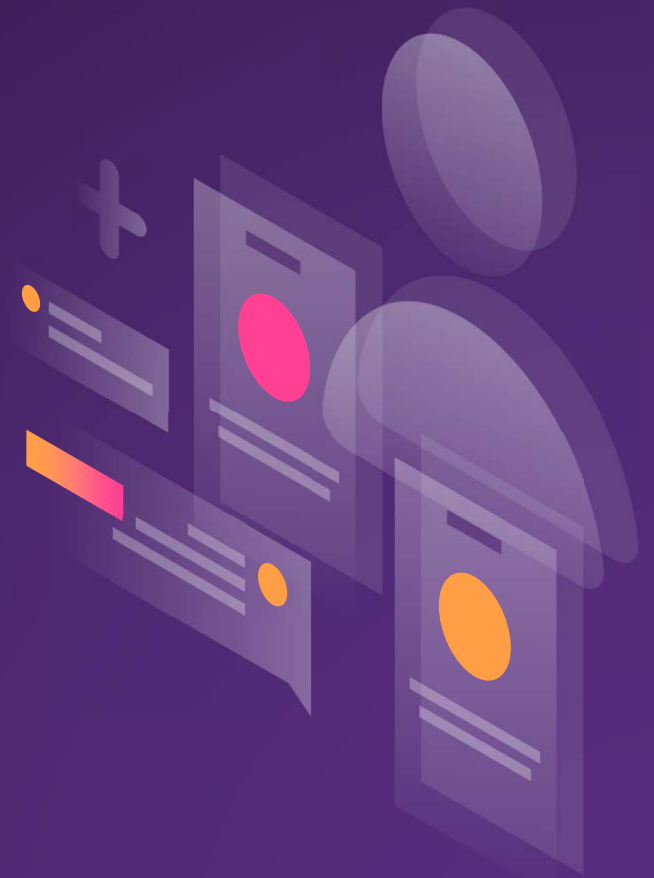
```
PS D:\matkul\sem 3\Basdat Lanjut\Project\car-data> npm install express mongoose cors --save
added 86 packages, and audited 87 packages in 13s

12 packages are looking for funding
  run `npm fund` for details

found 0 vulnerabilities
```

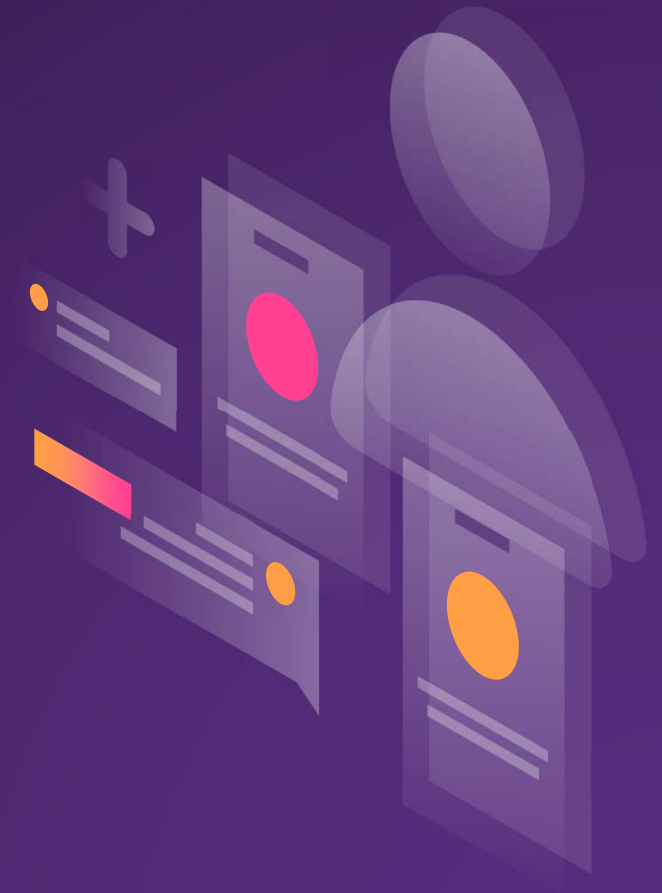
Dan menambahkan scriptbaru di package.js

```
1 {
2   "name": "example-app",
3   "version": "1.0.0",
4   "description": "Express API",
5   "main": "server.js",
6   "scripts": {
7     "dev": "nodemon server.js",
8     "test": "echo \"Error: no test specified\" && exit 1"
9   },
```



Membuat folder baru

```
▼ EXAMPLE-APP
  ▼ app
    > config
    > controllers
    > models
    > routes
  > node_modules
  {} package-lock.json
  {} package.json
```



Menambahkan file server.js, dan menuliskan beberapa kode untuk konfigurasi

```
JS server.js > app.get("/") callback > message
1  const express = require("express")
2  const cors = require("cors")
3  const app = express();
4
5  const corsOption = {
6    |   origin: "*"
7  };
8
9  //register cors middleware
10
11 app.use(cors(corsOption));
12 app.use(express.json());
13
14 //routes
15 app.get("/", (req, res) => {
16   |   res.json({ message: "Hello" });
17 });
18 const PORT = process.env.PORT || 8000;
19
20 app.listen(PORT, () => console.log(`server started on port ${PORT}`));
```

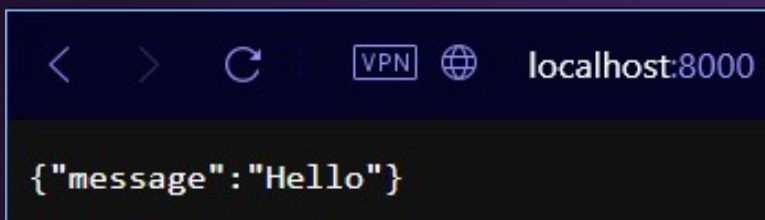


Mencoba untuk menjalankan server menggunakan terminal

```
PS D:\matkul\sem 3\Basdat Lanjut\Project\example-app> node server.js  
server started on port 8000
```

Kemudian buka browser dan ketikkan
<http://localhost:8000>

Jika berhasil, akan muncul tampilan berikut



The screenshot shows a web browser window with the address bar displaying "localhost:8000". The page content shows a JSON object: {"message": "Hello"}.

```
< > ↻ VPN localhost:8000  
{ "message": "Hello" }
```



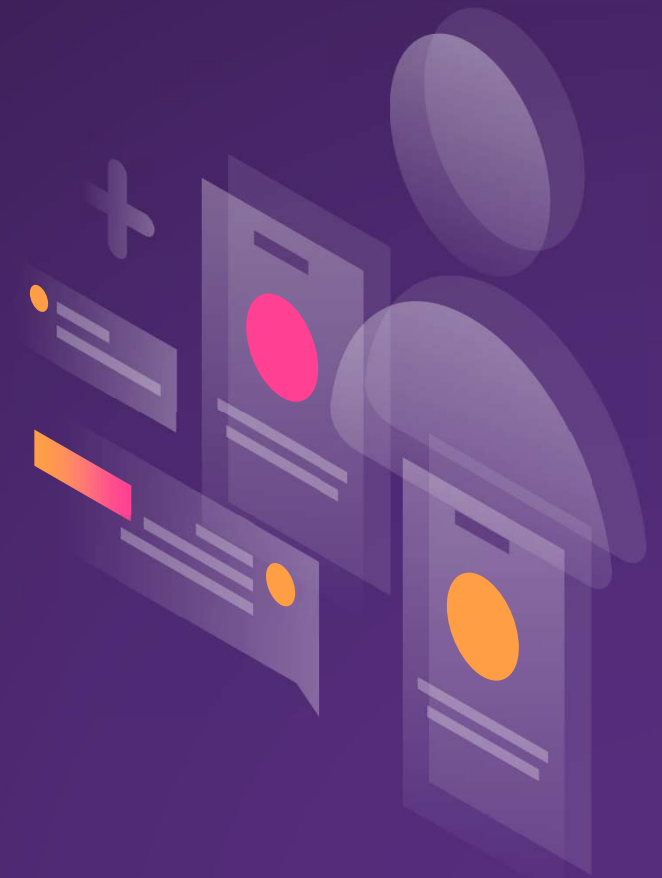
Menambahkan database.js untuk koneksi ke database

Menambahkan file database.js pada direktori berikut

```
▼ app
  ▼ config
    JS database.js
```

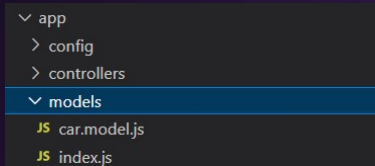
Serta menulis beberapa kode pada database.js

```
app > config > JS database.js > [?] <unknown> > 🔑 url
1  module.exports = {
2    💡 url: "mongodb://localhost:27017/express_api"
3  };
```



Menambahkan model

Menambahkan file mahasiswaModel.js dan index.js pada direktori berikut



Isi dari mahasiswaModel.js dan index.js

```

app > models > JS car.model.js > ...
1  module.exports = mongoose => {
2
3      const schema = mongoose.Schema(
4          {
5              car_brand: String,
6              car_model: String,
7              car_price: String,
8              year_produced: String,
9          }, {
10             timestamps: true
11         }
12     );
13     return mongoose.model("car", schema);
14 }
```

```

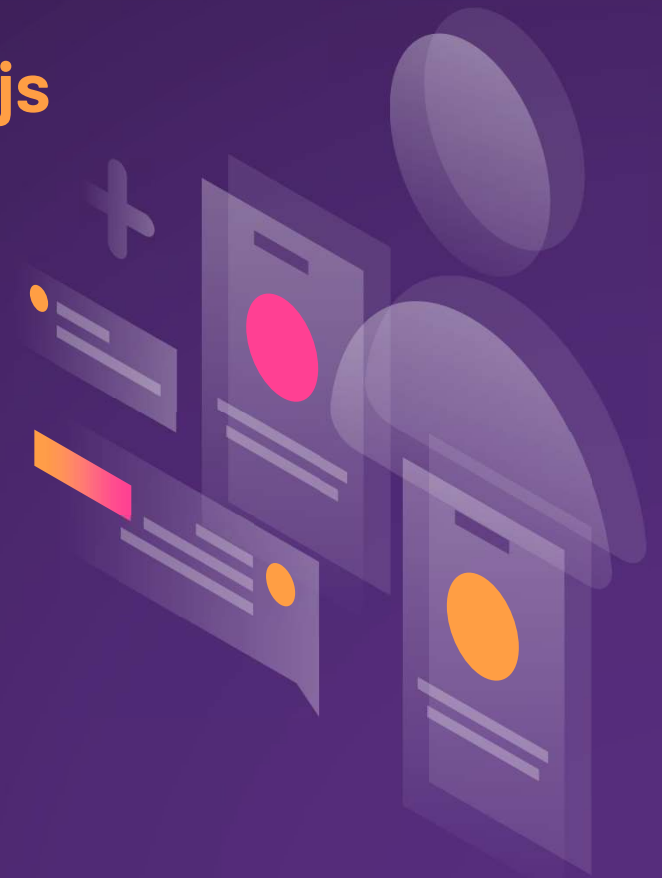
app > models > JS index.js > ...
1  const dbConfig = require("../config/database");
2  const mongoose = require("mongoose");
3
4  module.exports = {
5      mongoose,
6      url: dbConfig.url,
7      car: require("../car.model.js")(mongoose)
8  }
```



Setting koneksi database pada server.js

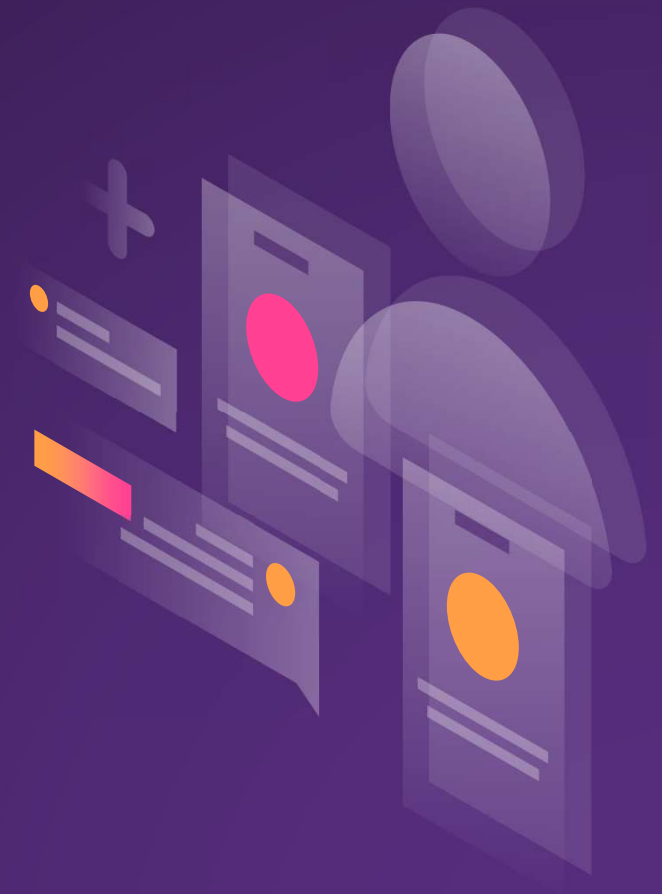
Pada server.js tambahkan kode berikut

```
1  const express = require("express")
2  const cors = require("cors")
3  const db = require("../app/models")
4  const app = express();
5
6  const corsOption = {
7    origin: "*"
8  };
9
10 //register cors middleware
11 app.use(cors(corsOption));
12 app.use(express.json());
13
14 const mongooseConfig = {
15   useNewUrlParser: true,
16   useUnifiedTopology: true
17 };
18
19 //connect ke database
20 db.mongoose.connect(db.url, mongooseConfig)
21   .then(() => console.log("Database.connected"))
22   .catch(err => console.log(`Connection failed${err.message}`));
```



Membuat controller

```
app > controllers > JS car.controller.js > ...
1  const db = require("../models");
2  const Car = db.car;
3
4  exports.create = (req, res) => {
5
6  }
7
8  exports.findAll = (req, res) => {
9    res.send({ message: "it works" });
10 }
11
12 exports.show = (req, res) => {
13
14 }
15
16 exports.update = (req, res) => {
17
18 }
19
20 exports.delete = (req, res) => {
21
22 }
```



Menambahkan route

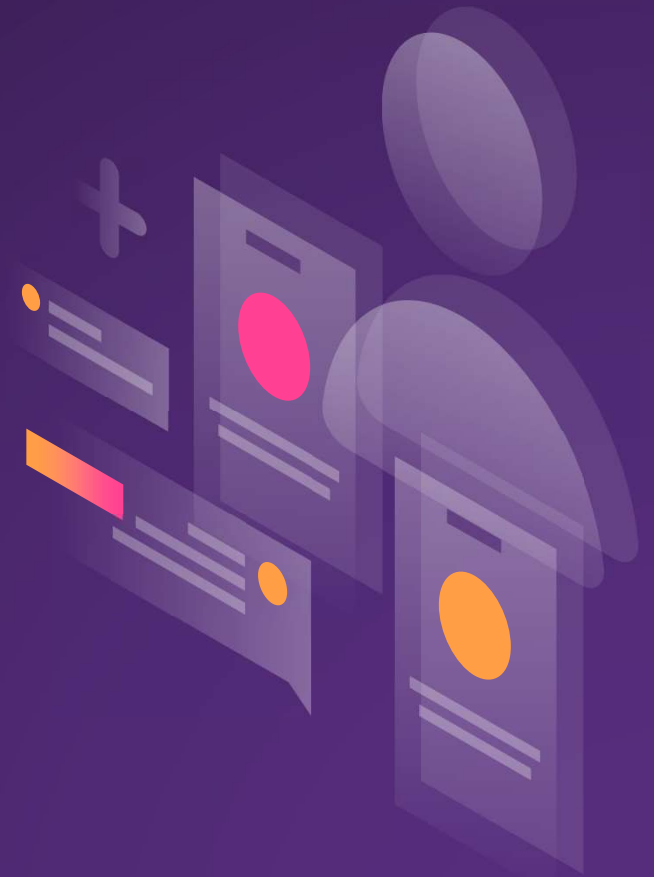
```
app > routes > JS car.route.js > <unknown> > exports
1  module.exports = app => {
2    const car = require("../controllers/car.controller")
3    const routes = require("express").Router();
4
5    routes.get("/", car.findAll);
6    routes.get("/:id", car.show);
7    routes.post("/", car.create);
8    routes.put("/:id", car.update);
9    routes.delete("/:id", car.delete);
10
11    app.use("/car", routes)
12  }
```



Mengubah server.js untuk pemanggilan route

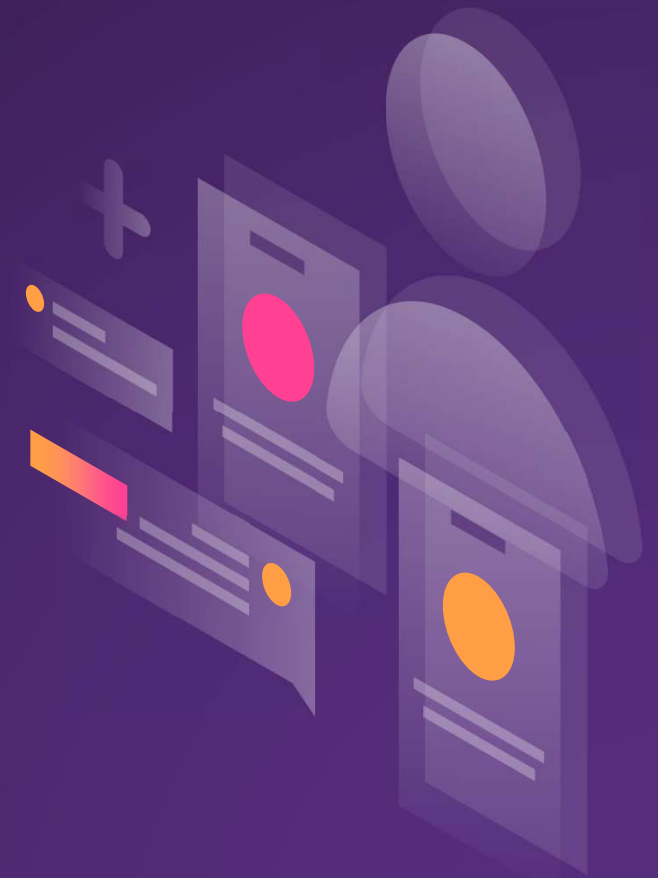
```
JS server.js > ...
1  const express = require("express")
2  const cors = require("cors")
3  const db = require("../app/models")
4  const app = express();
5
6  const corsOption = {
7    origin: "*"
8  };
9
10
11 //connect to database
12 const mongooseConfig = {
13   useNewUrlParser: true,
14   useUnifiedTopology: true
15 };
16
17 db.mongoose.connect(db.url, mongooseConfig)
18   .then(() => console.log("database connected"))
19   .catch(err => {
20     console.log(`connection failed ${err.message}`);
21     process.exit(1);
22   });
23
24 //register cors middleware
25
26 app.use(cors(corsOption));
27 app.use(express.json());
28
29 //routes
30 require("../app/routes/car.route")(app);
```

Pada server.js tambahkan kode berikut yang digunakan untuk memanggil route yang nantinya akan digunakan

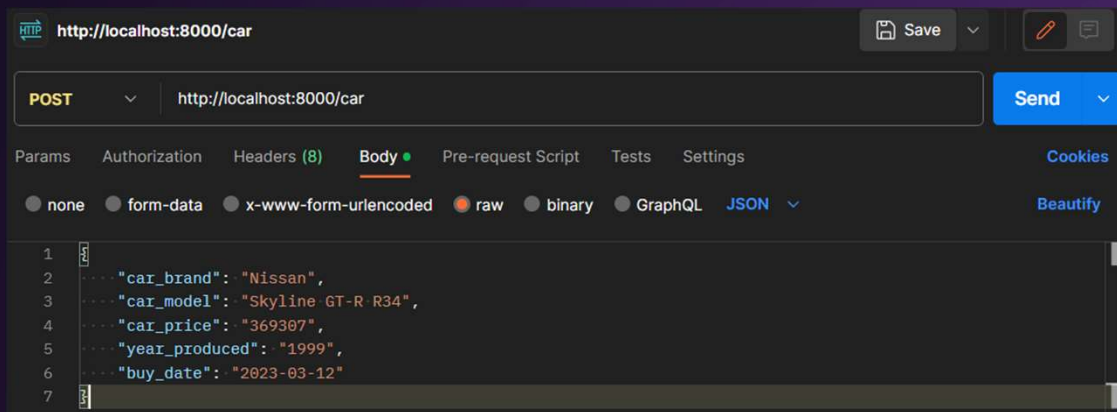


Menambahkan controller create pada car.controller

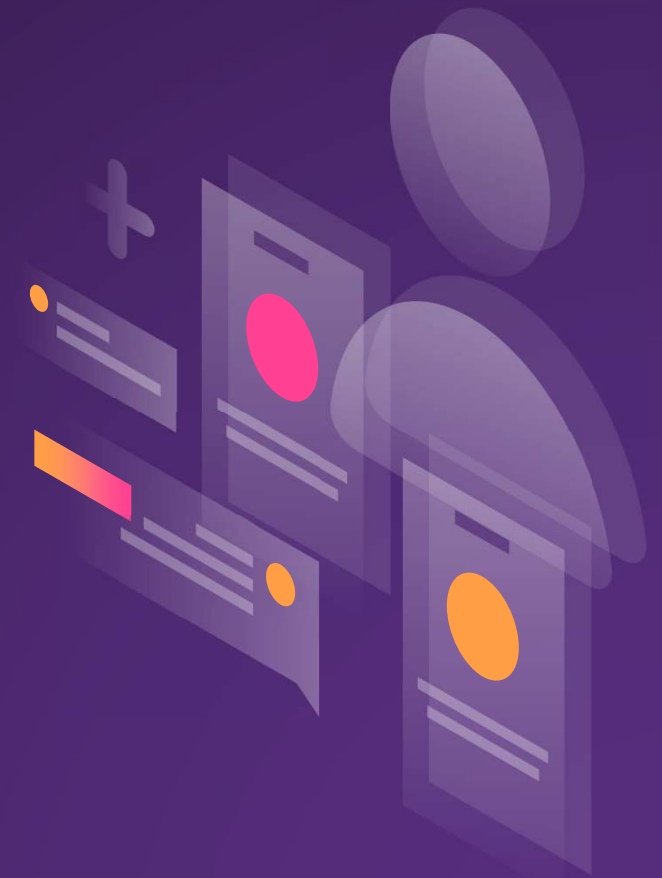
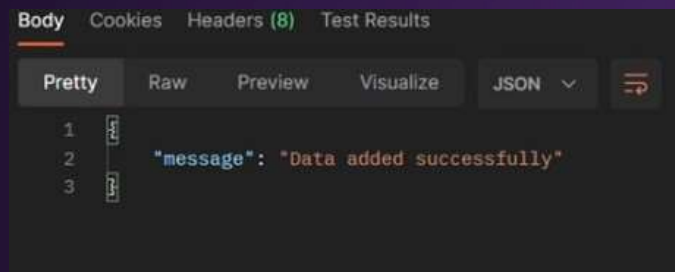
```
app > controllers > JS car.controller.js > create > create
1  const db = require("../models");
2  const Car = db.car;
3
4  exports.create = (req, res) => {
5
6      Car.create(req.body)
7          .then(() => res.send({ message: "Data added successfully" }))
8          .catch(err => res.status(500).send({ message: err.message }));
9  }
```



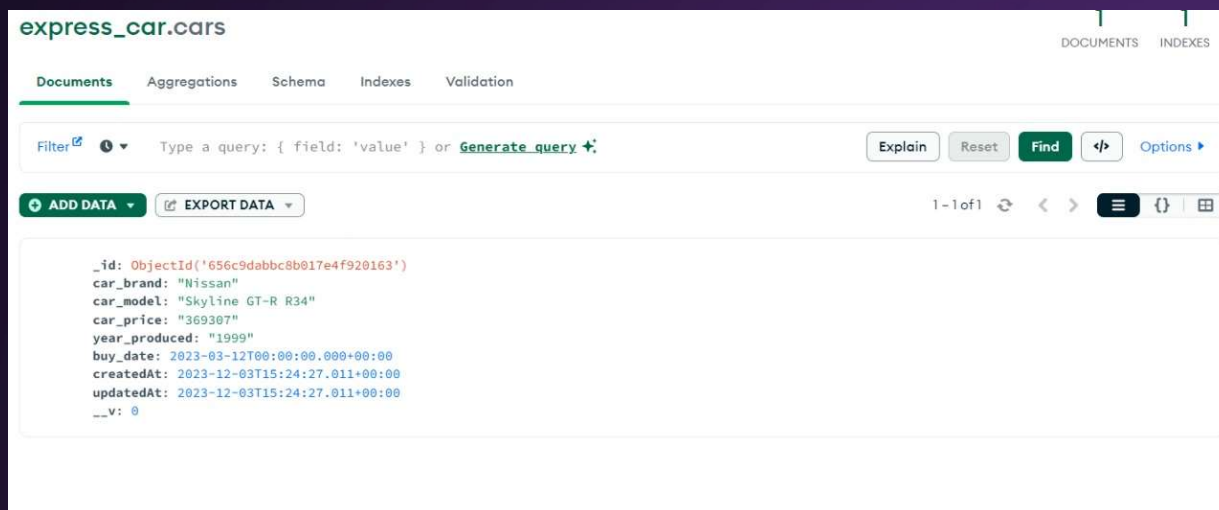
Melakukan percobaan di postman



Kemudian klik “send” dan tunggu prosesnya hingga muncul pesan berikut :



Cek di MongoDB Compass untuk update datanya



The screenshot shows the MongoDB Compass interface for the 'express_car.cars' collection. The 'Documents' tab is active, displaying a single document. The document contains the following fields:

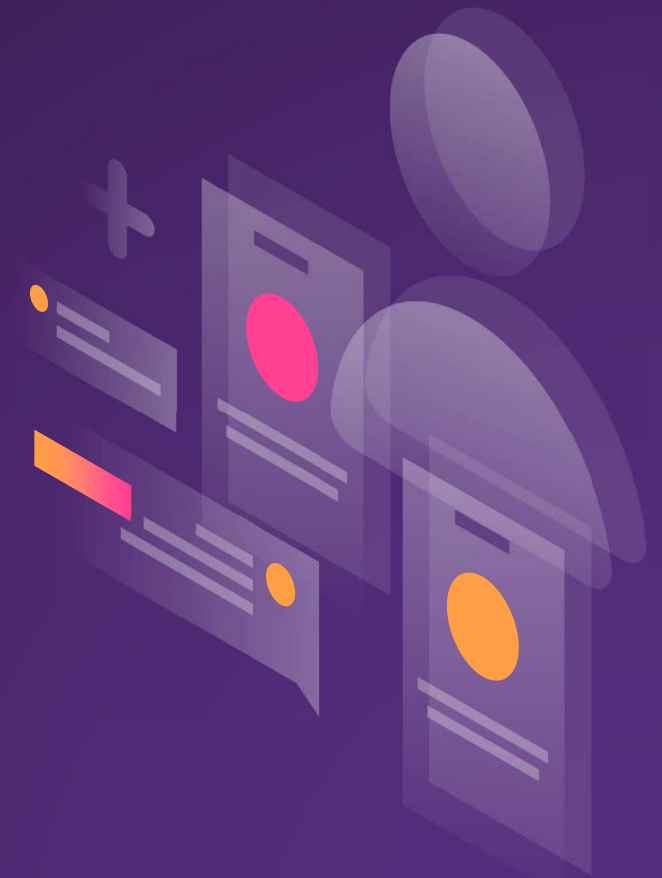
```
{
  "_id": ObjectId("656c9dabbc8b017e4f920163"),
  "car_brand": "Nissan",
  "car_model": "Skyline GT-R R34",
  "car_price": "369307",
  "year_produced": "1999",
  "buy_date": "2023-03-12T00:00:00.000+00:00",
  "createdAt": "2023-12-03T15:24:27.011+00:00",
  "updatedAt": "2023-12-03T15:24:27.011+00:00",
  "__v": 0
}
```

The interface includes a search bar with a filter icon, a query input field, and buttons for 'Explain', 'Reset', 'Find', and 'Options'. Below the search bar are buttons for 'ADD DATA' and 'EXPORT DATA'. The document is displayed in a JSON format, and the status bar at the bottom indicates '1 - 1 of 1'.

— Mengubah route untuk show data

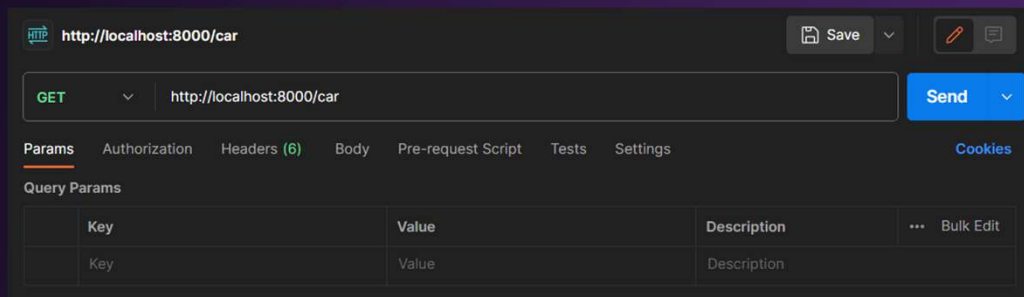
Pada `car.controller.js` tambahkan beberapa kode untuk menampilkan data

```
app > controllers > JS car.controller.js > ...
1  const db = require("../models");
2  const Car = db.car;
3
4  exports.create = (req, res) => {
5
6      req.body.buy_date = new Date(req.body.buy_date)
7
8      Car.create(req.body)
9          .then(() => res.send({ message: "Data added successfully" }))
10         .catch(err => res.status(500).send({ message: err.message }));
11 }
12
13 exports.findAll = (req, res) => {
14     Car.find()
15         .then(data => res.send(data))
16         .catch(err => res.status(500).send({ message: err.message }));
17 }
```

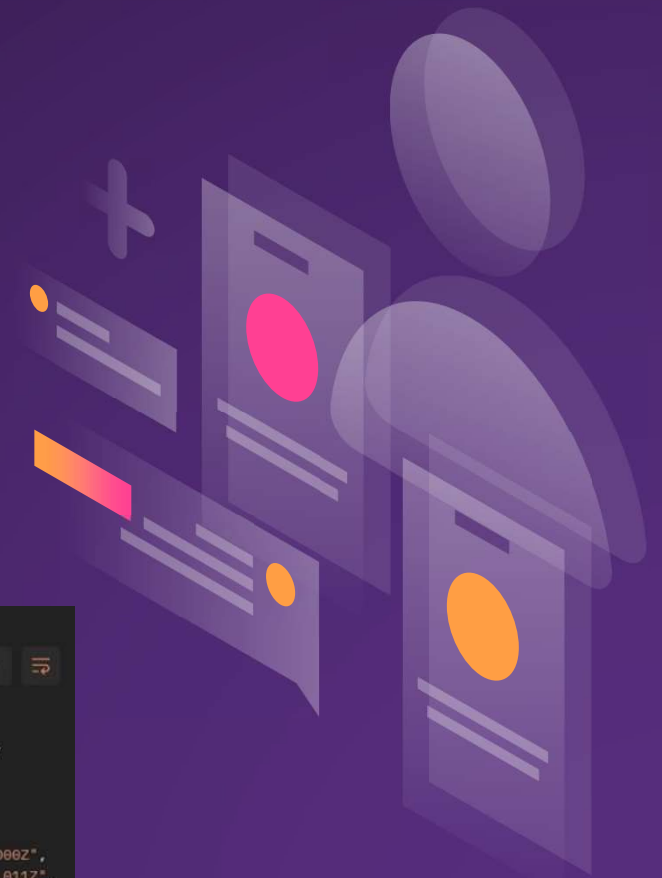
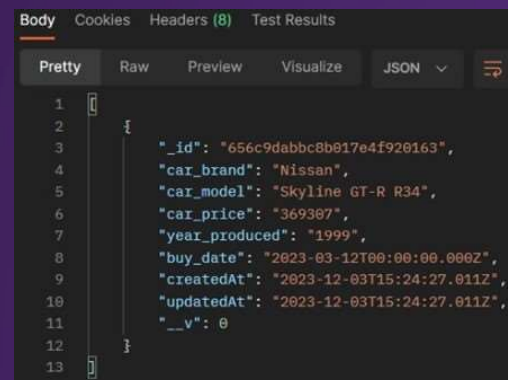


Mengecek perubahan menggunakan postman

Kirim sebagai get, dan tunggu prosesnya



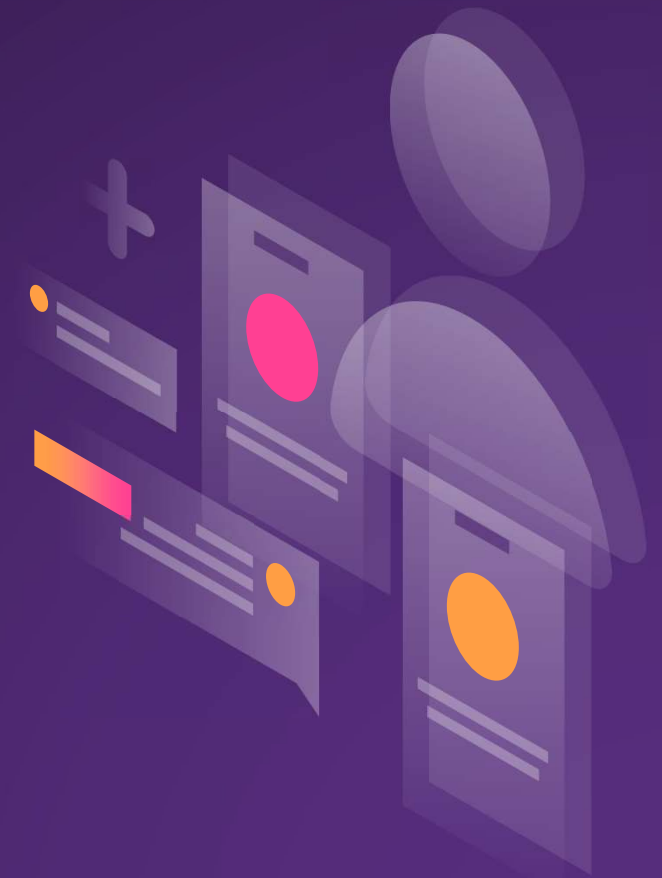
Jika berhasil, akan muncul tampilan berikut



— Menghilangkan `_id` dan `__v`

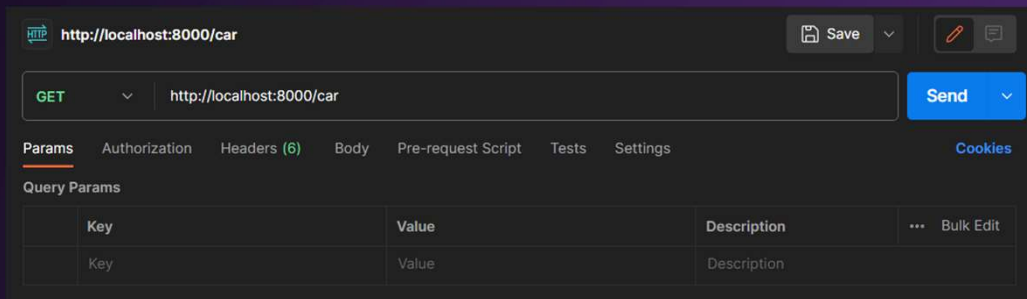
Pada `car.model` tambahkan beberapa kode berikut

```
app > models > JS car.model.js > <unknown> > exports > schema.method("toJSON") callback
1  module.exports = mongoose => {
2
3      const schema = mongoose.Schema(
4          {
5              car_brand: String,
6              car_model: String,
7              car_price: String,
8              year_produced: String,
9              buy_date: Date
10         }, {
11             timestamps: true
12         });
13
14         schema.method("toJSON", function () {
15             const { __v, _id, ...object } = this.toObject()
16             object.id = _id;
17
18             return object;
19         });
20
21         return mongoose.model("car", schema);
22     }
23
```



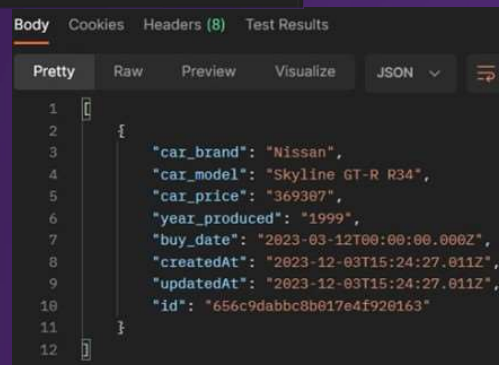
Mengecek data yang ada menggunakan postman

Kirim sebagai get, dan tunggu prosesnya



Jika berhasil, akan muncul tampilan berikut :

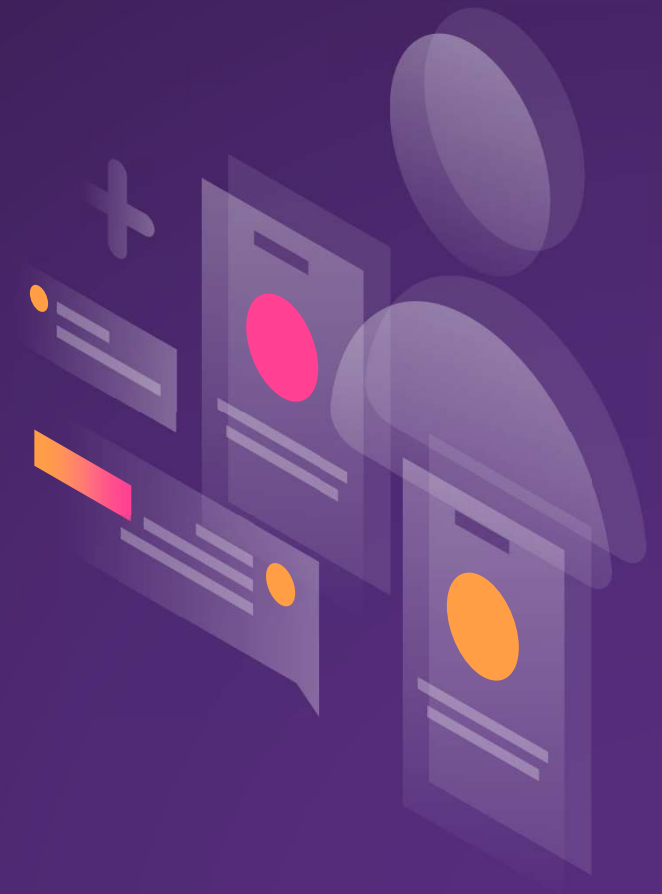
Dan `_id` berubah menjadi `id`, dan `__v` tidak muncul lagi



Membuat route untuk show data berdasarkan id

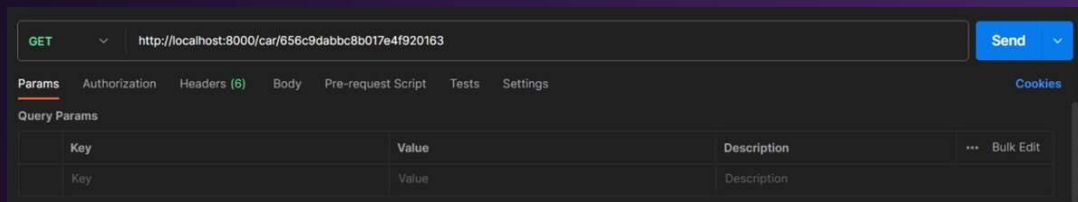
Pada `car.controller.js` tambahkan kode berikut

```
exports.show = (req, res) => {  
  const id = req.params.id;  
  
  Car.findById(id)  
    .then(data => res.send(data))  
    .catch(err => res.status(500).send({ message: err.message }));  
}
```



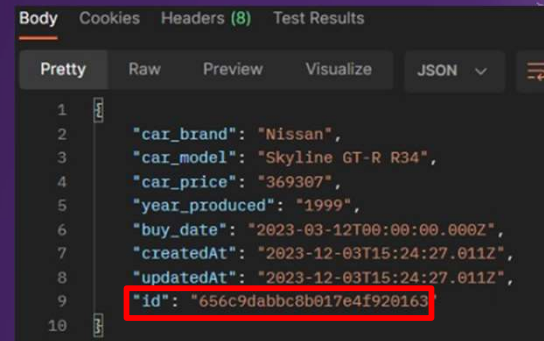
Cek perubahan menggunakan postman

Sertakan id dari data yang sudah ada sebelumnya



Jika berhasil, akan muncul tampilan berikut :

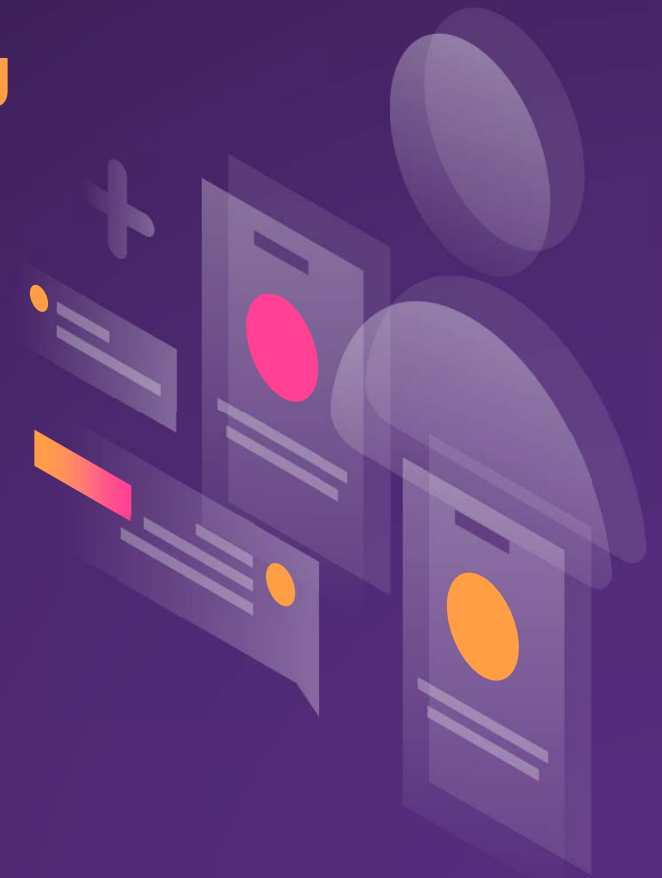
Sesuai dengan id yang dicari



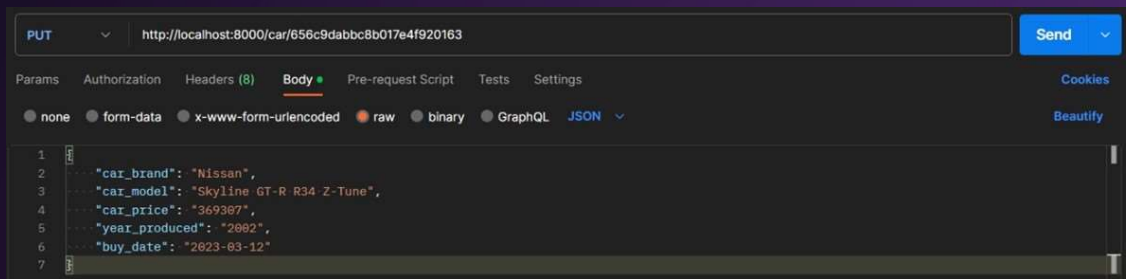
Membuat route untuk update data yang sudah ada

Pada car.controller.js tambahkan kode berikut

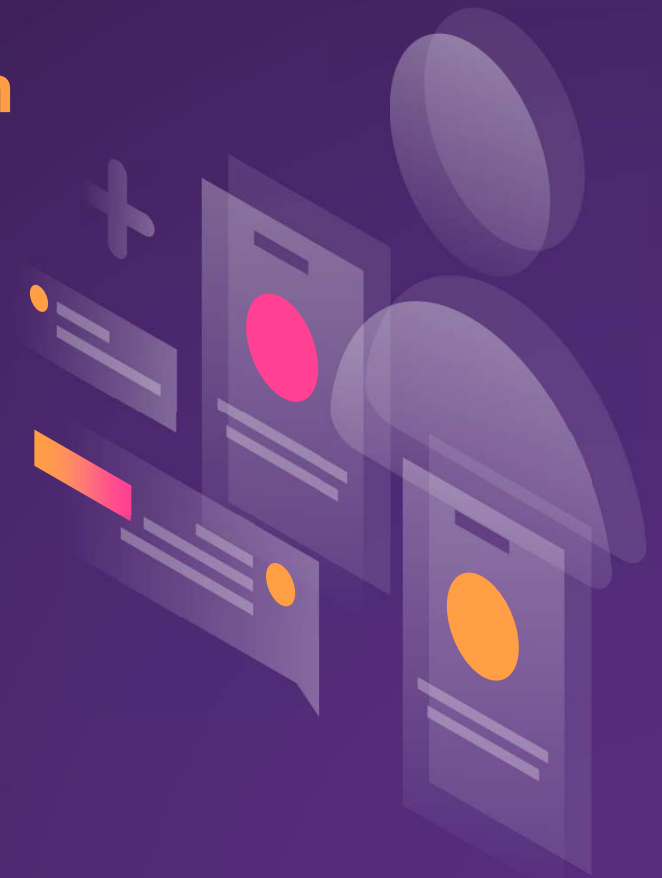
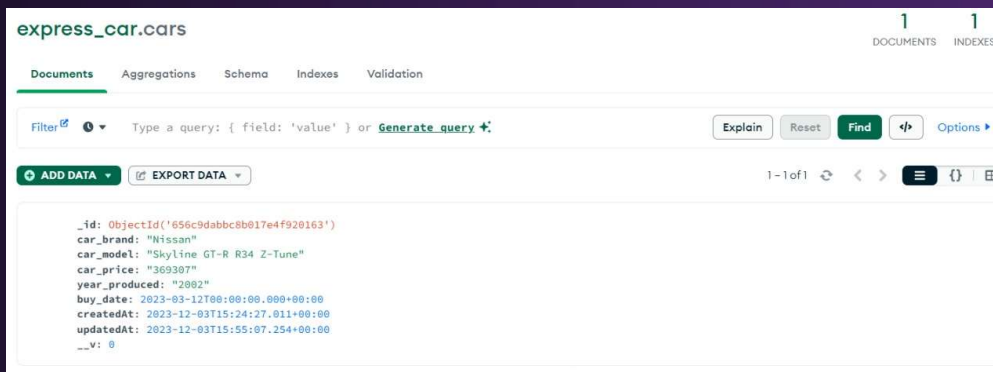
```
exports.update = (req, res) => {  
  const id = req.params.id;  
  
  req.body.buy_date = new Date(req.body.buy_date);  
  
  Car.findByIdAndUpdate(id, req.body, { useFindAndModify: false })  
    .then(data => {  
      if (!data) {  
        res.status(404).send({ message: "Cannot update data" })  
      }  
      res.send({ message: "Data updated successfully" })  
    })  
    .catch(err.status(500).send({ message: err.message })))  
}
```



— Cek perubahan menggunakan postman



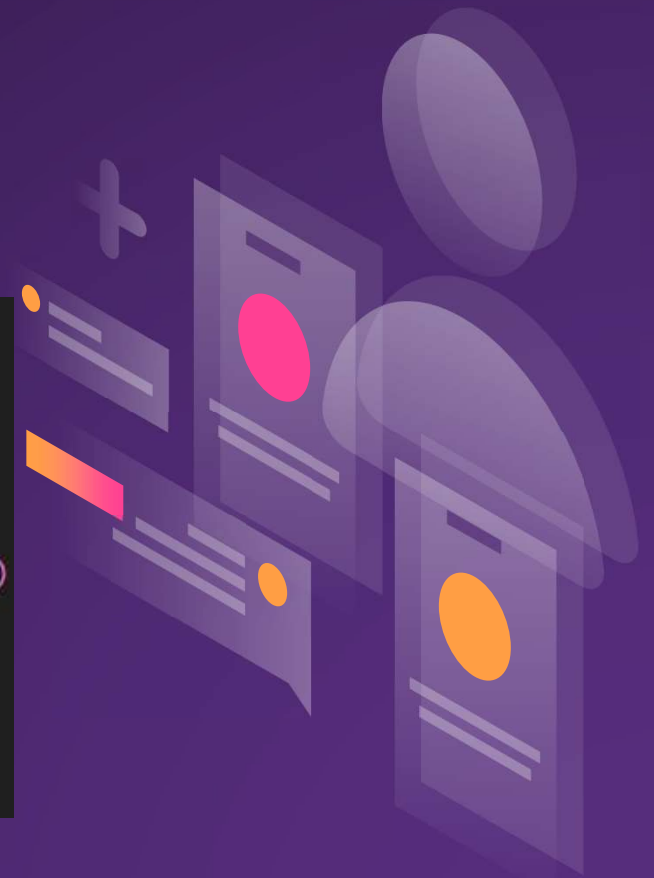
Jika berhasil, akan di update di database



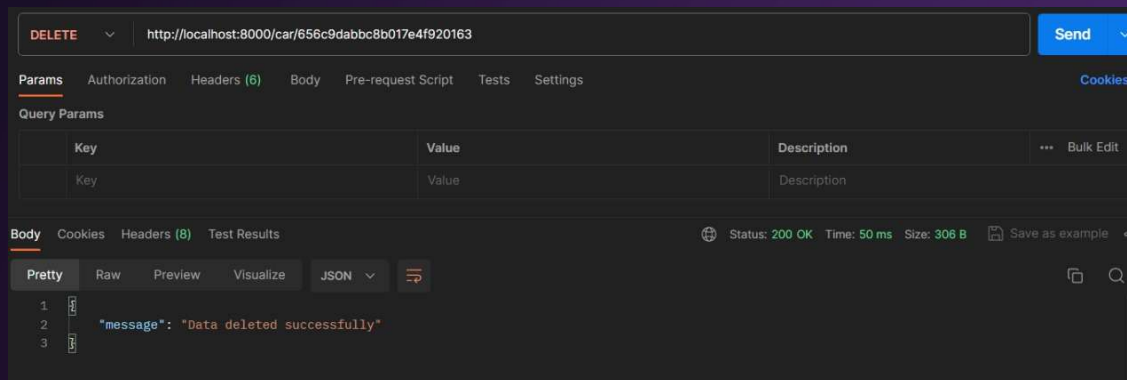
Menambahkan route untuk delete data yang sudah ada

Pada car.controller.js tambahkan kode berikut

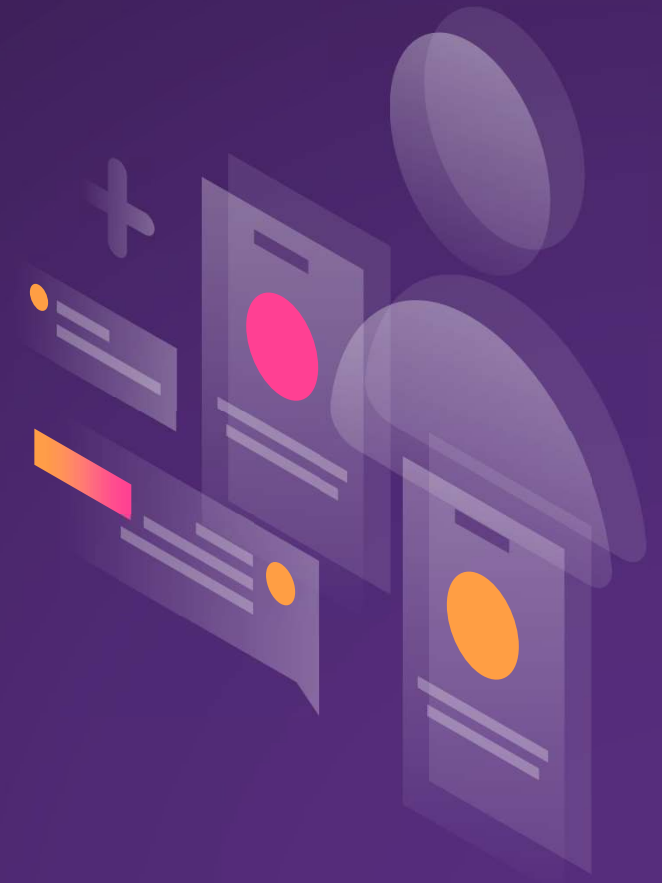
```
exports.delete = (req, res) => {  
  const id = req.params.id;  
  
  Car.findByIdAndDelete(id)  
    .then(data => {  
      if (!data) {  
        res.status(404).send({ message: "Cannot delete data" })  
      }  
      res.send({ message: "Data deleted successfully" })  
    })  
    .catch(err => res.status(500).send({ message: err.message })))  
}
```



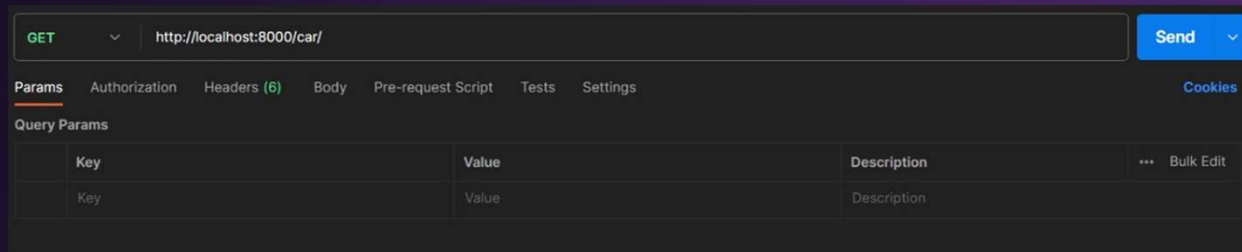
Melakukan percobaan delete menggunakan postman



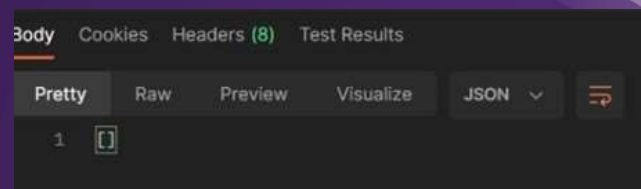
Jika berhasil, akan muncul pesan “Data deleted successfully”



Mengecek data tadi menggunakan postman



Dan data yang tadi sudah terhapus, sehingga tidak muncul jika di panggil



— Terima Kasih!!

Sekian presentasi dari saya

