Building RESTful APIs with Node.js and Express For Kafka Client

By : Abdeljalil BENETTALEB

Installation Prerequisites

- Install Postman
- Install Node and Express
- Install MongoDB & Mongoose
- Install Nodemon
- Install babel-cli & babel-preset-es2015 & babel-preset-stage-0
  Create new file called **.babelrc** :

```
{
  "presets": [
    "es2015",
"stage-0"
  ]
}
```

Initial Server Build

- Install Body-parser

Create new file called **index.js** :

```
import express from 'express';

const app = express();
const PORT = 3000;

app.get('/', (req, res) =>
    res.send(`Node and express server is running on port ${PORT}`)
);

app.listen(PORT, () =>
    console.log(`Your server is running on port ${PORT}`)
);
```

In Packge.json :

```
{
  "name": "crm",
  "version": "1.0.0",
  "description": "Rest API For Kafka Web Server",
  "main": "index.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1",
    "start": "nodemon ./index.js --exec babel-node -e js"
  },
```

```
  "repository": {
    "type": "git",
    "url": "git+https://github.com/ibenettaleb/restAPIKafka.git"
  },
  "keywords": [
    "Rest",
    "API",
    "NodeJS",
    "Kafka",
    "ExpressJS"
  ],
  "author": "Abdeljalil BENETTALEB",
  "license": "ISC",
  "bugs": {
    "url": "https://github.com/ibenettaleb/restAPIKafka/issues"
  },
  "homepage": "https://github.com/ibenettaleb/restAPIKafka#readme",
  "dependencies": {
    "body-parser": "^1.18.2",
    "express": "^4.16.3",
    "mongoose": "^5.0.11",
    "nodemon": "^1.17.2"
  },
  "devDependencies": {
    "babel-cli": "^6.26.0",
    "babel-preset-es2015": "^6.24.1",
    "babel-preset-stage-0": "^6.24.1"
  }
}
```

Create those folder :

1. Src
    a. Controllers
        i. crmController.js
    b. models
        i. crmModel.js
    c. routes
        i. crmRoute.js

Inside crmRoute.js :

```
const routes = (app) => {
app.route('/contact')
        .get((req, res) =>
        res.send('GET request successfull !'))
        .post((req, res) =>
        res.send('POST request successfull !'));

    app.route('/contact/:contactId')
        .put((req, res) =>
            res.send('PUT request successfull !'))

        .delete((req, res) =>
            res.send('DELETE request successfull !'));
};


export default routes;
```

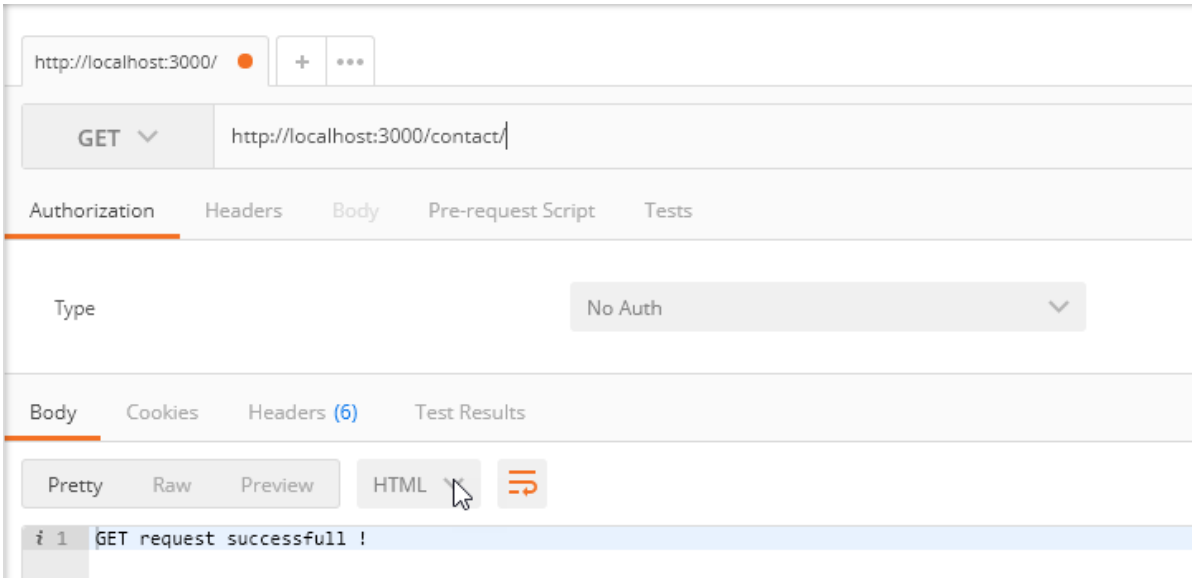
Inside index.js file add this :

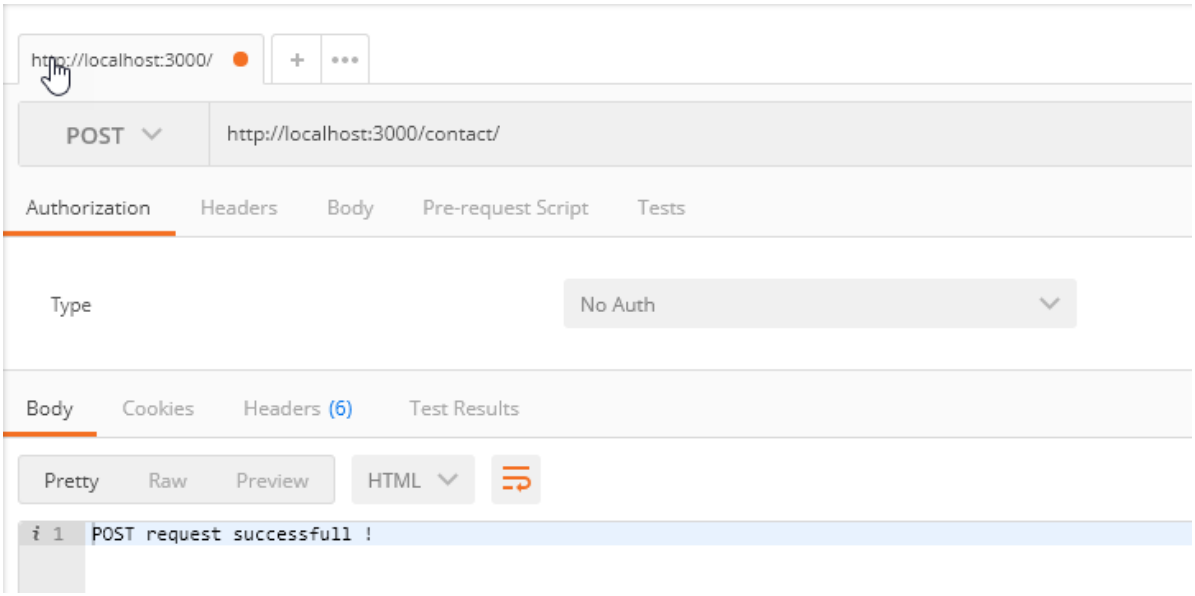
```
import routes from './src/routes/crmRoute' ;

***

routes(app)
```

In **Postman** :





The same reasult for PUT and DELETE Reaquest

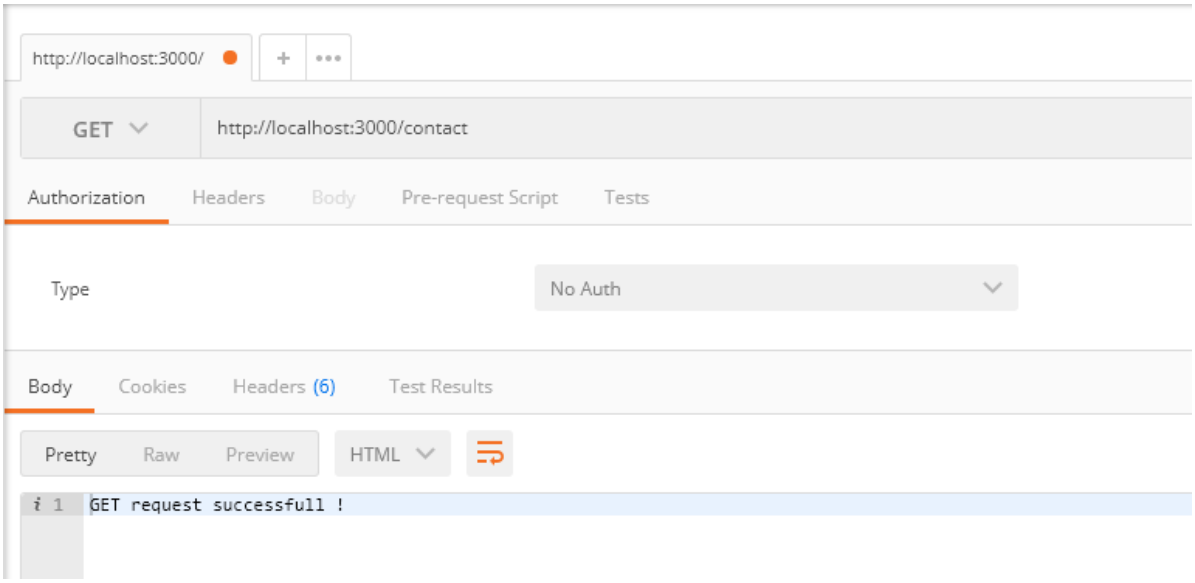
Inside crmRoutes Add this :

```
app.route('/contact')
    .get((req, res) => {
        // middleware
        console.log(`Request from: ${req.originalUrl}`);
```

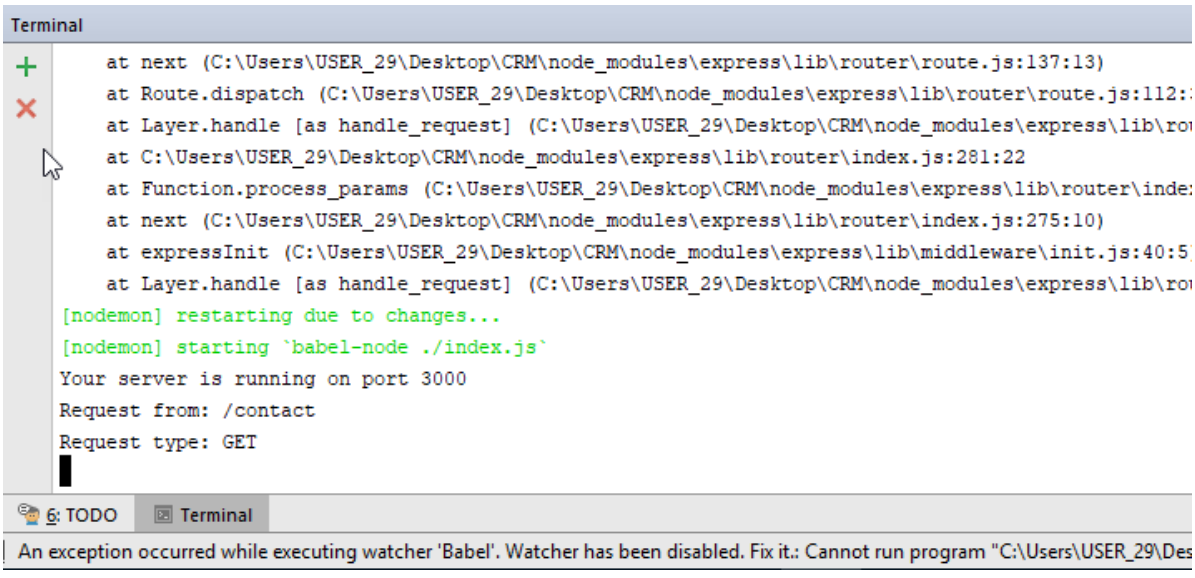
```
        console.log(`Request type: ${req.method}`);
        next();
    }, (req, res, next) => {
        res.send('GET request successfull !');
    })

    .post((req, res) =>
    res.send('POST request successfull !'));
```

And try this in **Postman** :



And in Terminal you well see :



MongoDB

- Database with collections

- Collections have documents or objects
- Documents look like JSON object
- Inside each document, you have the data with key-value pairs or arrays of items
- So that our database has contacts, which is a collection, and each contact is a document

Install Robomongo

MongoDB Managment tool

Run this command :

```
mkdir -p /data/db
cd /data/db/
sudo chmod 0755 /data/db sudo chown $USER /data/db
```

After that connect to Robomongo.

Database & **Schema** Setup:

- Database :

```
import express from 'express';
import mongoose from 'mongoose';
import bodyParser from 'body-parser';
import routes from './src/routes/crmRoute';

const app = express();
const PORT = 3000;

//Set up default mongoose connection
let mongoDB = 'mongodb://10.1.22.90/CRMdb';
mongoose.Promise = global.Promise;
mongoose.connect(mongoDB);

// bodyparser setup
app.use(bodyParser.urlencoded({ extended: true }));
app.use(bodyParser.json());

routes(app);

app.get('/', (req, res) =>
    res.send(`Node and express server is running on port ${PORT}`)
);

app.listen(PORT, () =>
    console.log(`Your server is running on port ${PORT}`)
);
```

- Schema :

```
import mongoose from 'mongoose';

const Schema = mongoose.Schema;

export const ContactSchema = new Schema({
    firstName: {
```

```
        type: String,
        required: 'Enter a first name'
    },
    lastName: {
        type: String,
        required: 'Enter a first name'
    },
    email: {
        type: String,
    },
    company: {
        type: String,
    },
    phone: {
        type: Number,
    },
    created_date: {
        type: Date,
        default: Date.now
    }
});
```

Create POST endpoint :

In crmController typing :

```
import mongoose from 'mongoose';
import { ContactSchema } from '../models/crmModel';

const Contact = mongoose.model('Contact', ContactSchema);

export const addNewContact = (req, res) => {
    let newContact = new Contact(req.body);

    newContact.save((err, contact) => {
        if (err) {
            res.send(err);
        }
        res.json(contact);
    });
};
```

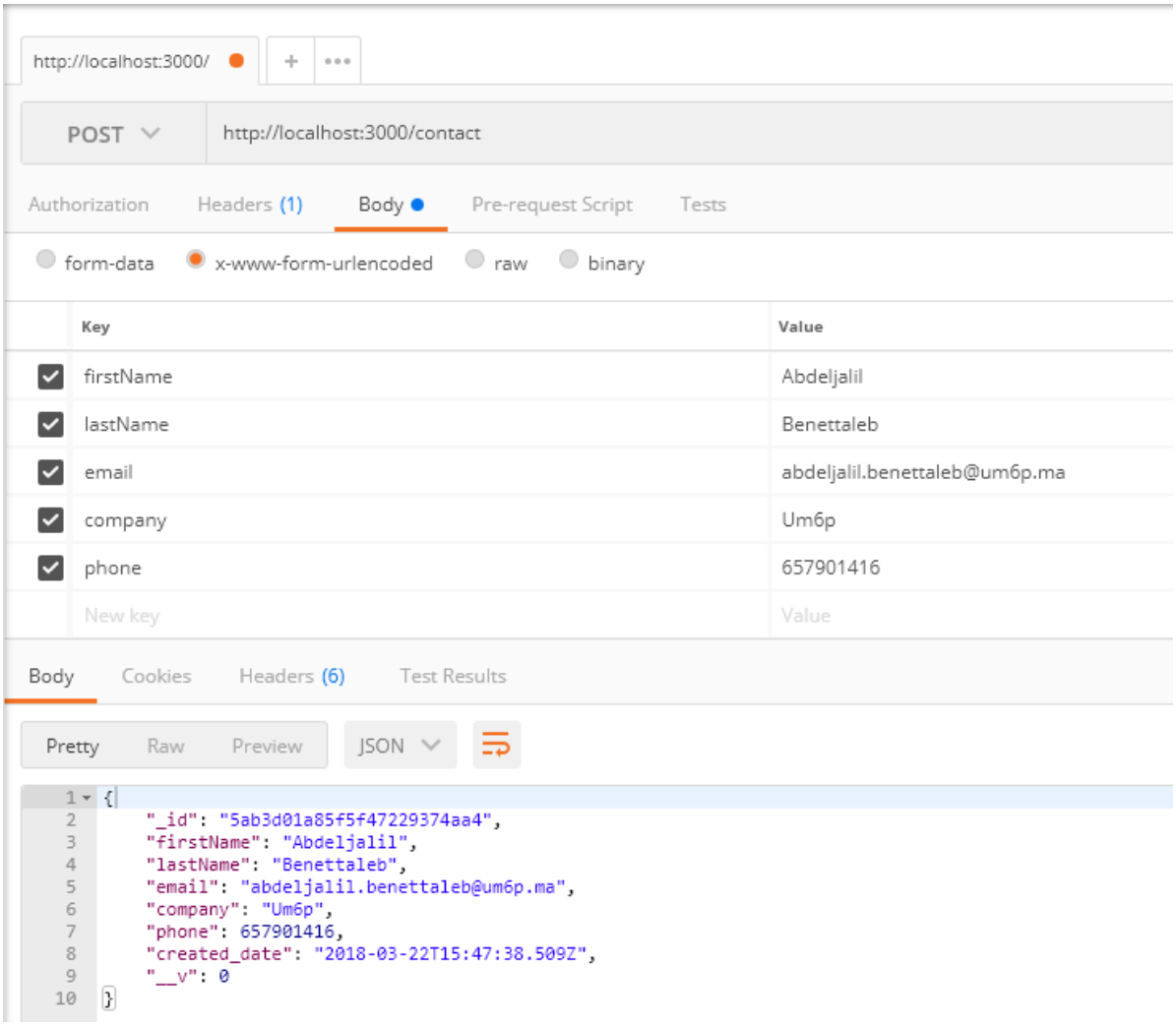
In crmRoute typing :

```
import { addNewContact } from '../controllers/crmController';

. . .

//POST endpoint
.post(addNewContact);
```

Test that in Postman:



Create all Items GET endpoint:

Inside crmController.js Typing :

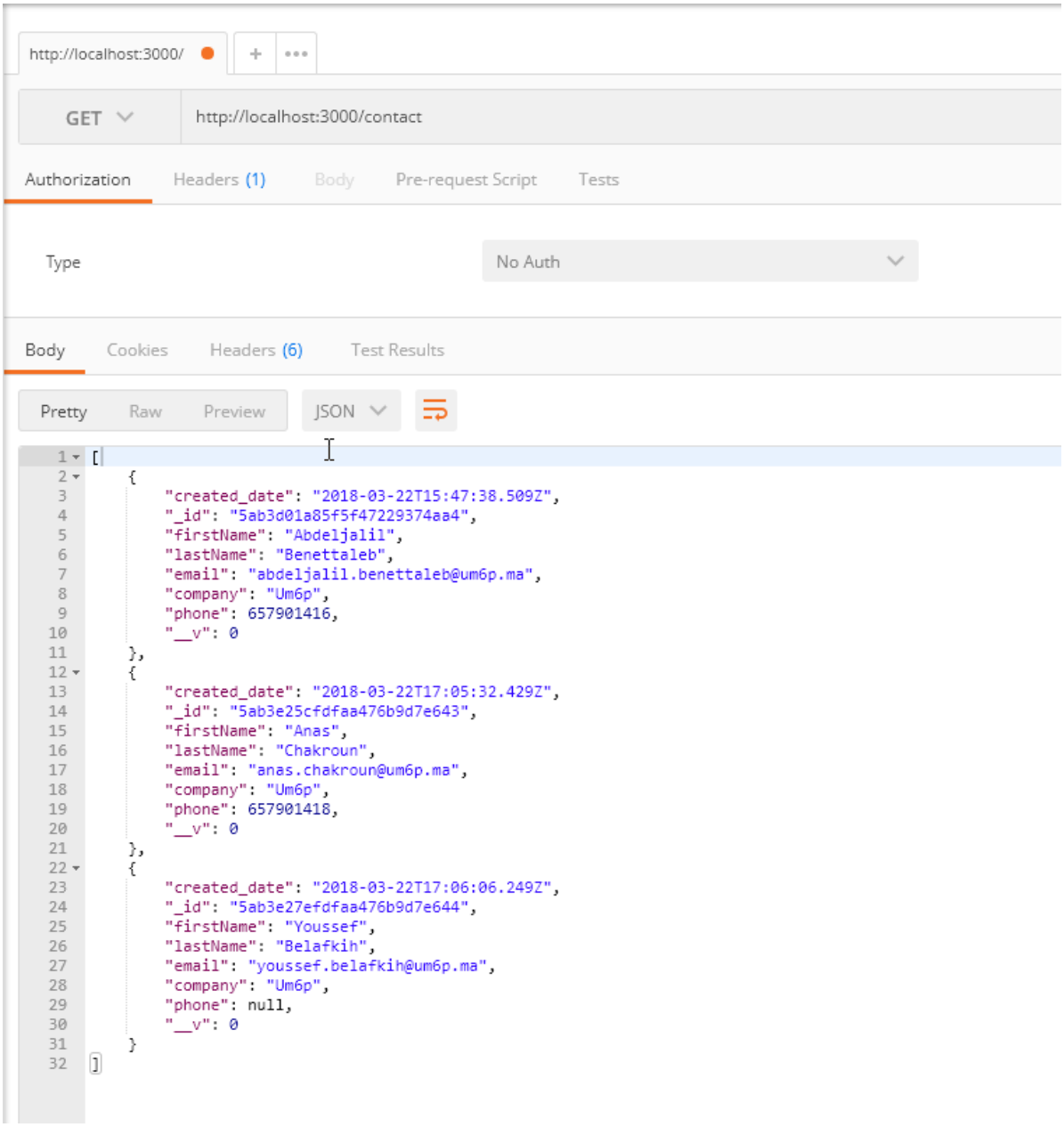
```
. . .

export const getContacts = (req, res) => {
    Contact.find({}, (err, contact) => {
        if (err) {
            res.send(err)
        }
        res.json(contact)
    });
};
```

Use the method inside crmRoute.js :

```
.get((req, res, next) => {
    // middleware
    console.log(`Request from: ${req.originalUrl}`);
    console.log(`Request type: ${req.method}`);
    next();
}, getContacts)
```

Try this in Postman:



Create specific ID GET endpoint:

Inside crmController.js Typing :

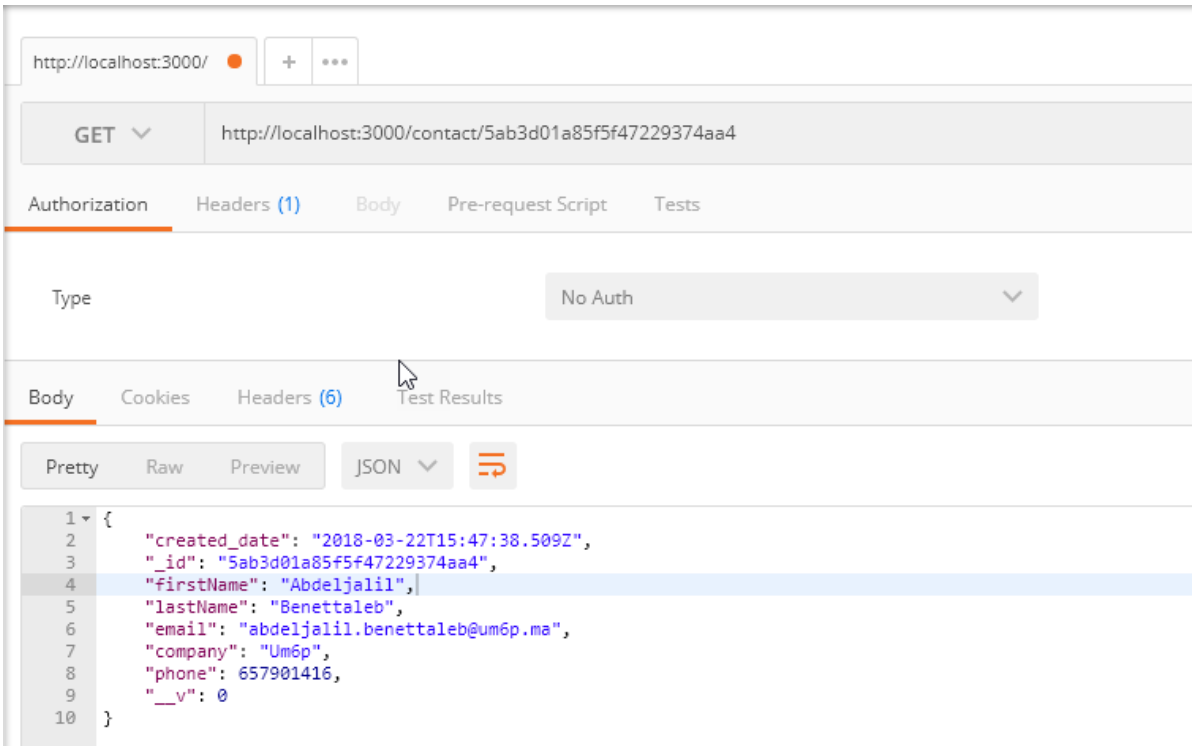
```
export const getContactWithID = (req, res) => {
    Contact.findById(req.params.contactId, (err, contact) => {
        if (err) {
            res.send(err)
        }
        res.json(contact)
    });
};
```

Use the method inside crmRoute.js :

```
app.route('/contact/:contactId')
    // get specific contact
    .get(getContactWithID)
    // put request
    .put((req, res) =>
        res.send('PUT request successfull !'))

    // delete request
    .delete((req, res) =>
        res.send('DELETE request successfull !'));
```

Try this in Postman:



Create PUT endpoint:

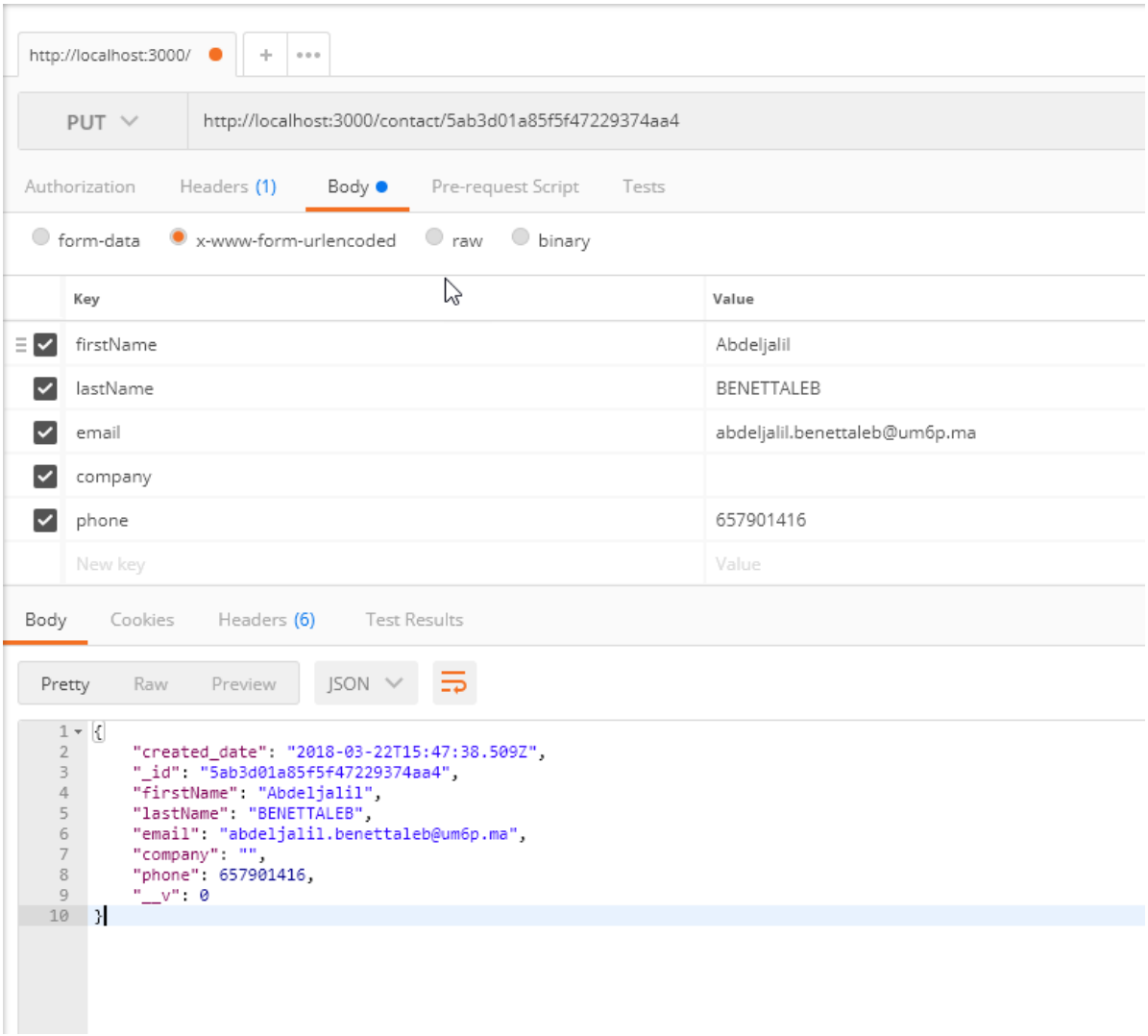
Inside crmController.js Typing :

```
export const updateContact = (req, res) => {
    Contact.findOneAndUpdate({ _id: req.params.contactId }, req.body, { new: true
}, (err, contact) => {
        if (err) {
            res.send(err)
        }
        res.json(contact)
    })
};
```

Use the method inside crmRoute.js :

```
// put request
.put(updateContact)
```

Try this in Postman:



Create DELETE endpoint:

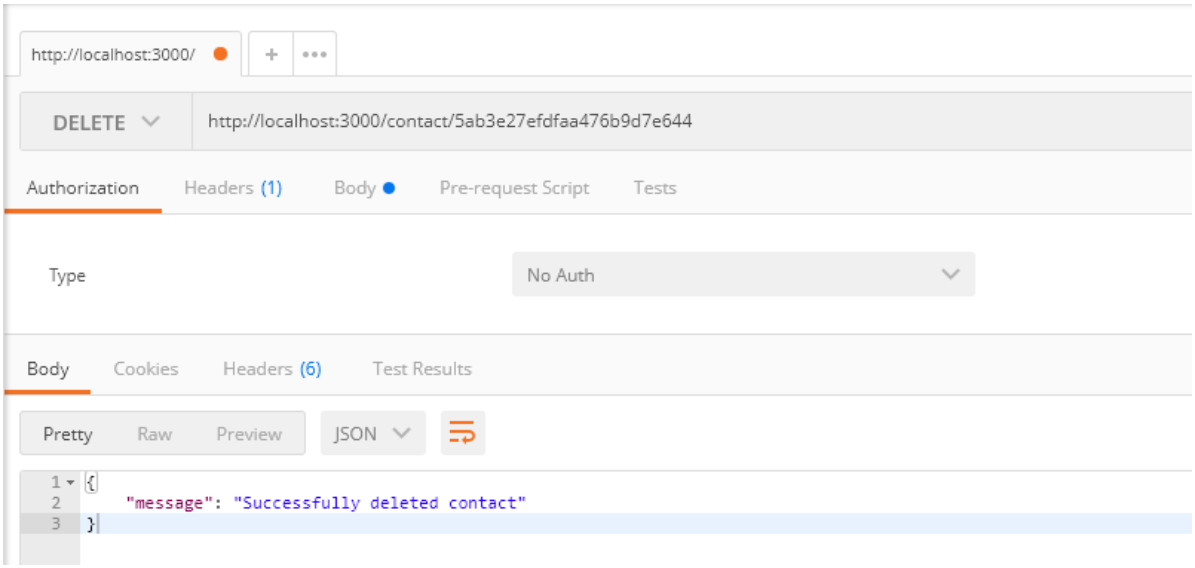
Inside crmController.js Typing :

```
export const deleteContact = (req, res) => {
    Contact.remove({_id: req.params.contactId}, (err, contact) => {
        if (err) {
            res.send(err)
        }
        res.json ({ message: 'Successfully deleted contact' });
    })
};
```

Use the method inside crmRoute.js :

```
// delete request
.delete(deleteContact);
```

Try this in Postman:



Note!

You will need add all this method in crmRoute like that :

```
import {
    addNewContact,
    getContacts,
    getContactWithID,
    updateContact,
    deleteContact, } from '../controllers/crmController';
```

Okey, so now we've got a full API