BỘ GIÁO DỤC VÀ DÀO TẠO
TRƯỜNG ĐẠI HỌC CẦN THƠ
TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

BÁO CÁO THỰC HÀNH BACKEND (1+2)

PHÁT TRIỂN ỨNG DỤNG WEB

CT449 - 01

Giảng viên hướng dẫn:
ThS. Nguyễn Minh Trung

Sinh viên thực hiện:
Tên: Trần Trương Ngọc Uyển
MSSV: B2207576
Khóa 48

Cần Thơ, năm 2025

MỤC LỤC

DANH MỤC HÌNH

Ứng dụng Contactbook - Backend - Phần 1

Bước 0: Cài đặt môi trường Node và git

- Thực hiện tải nodejs và git theo hướng dẫn.
- Sau đó kiểm tra bằng lệnh như ảnh:

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1> node -v
v22.14.0
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1> git --version
git version 2.45.1.windows.1
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1>
```

Hình 1: Kiểm tra môi trường

Bước 1: Tạo ứng dụng Node

- Tạo thư mục dự án có tên " ContactBook-backend "
- Trong thư mục này, ta khởi tạo ứng dụng Node bằng lệnh:

```
npm init
```

Chú thích: *npm* sẽ hỏi về một số thông tin để tạo tập tin package.json cho dự án

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> 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: (contactbook-backend)
```

Hình 2: Khởi động ứng dụng Node thành công

```
version: (1.0.0)
description:
entry point: (index.js) server.js
test command:
git repository:
keywords:
author: Uyen Tran
license: (ISC)
About to write to D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend\package.json:

{
  "name": "contactbook-backend",
  "version": "1.0.0",
  "main": "server.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "Uyen Tran",
  "license": "ISC",
  "description": ""
}
```

Hình 3: Thông tin để tạo tập tin package.json

– Cài đặt các package cần thiết

npm install express cors

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> npm install express cors

added 69 packages, and audited 70 packages in 2s

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

found 0 vulnerabilities
```

Hình 4: Tải package cần thiết

Bước 2: Quản lý mã nguồn dự án với git và github

– Tạo tập tin ".gitignore " bằng lệnh **npx gitignore node**

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> npx gitignore node
Need to install the following packages:
gitignore@0.7.0
Ok to proceed? (y) y

Created .gitignore file for flag type node.
```

Hình 5: Tạo tập tin ".gitignore "

– Gõ lệnh: " **git init** " để khởi tạo dự án Git.

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> git init
Initialized empty Git repository in D:/CT449_PTUD_WEB/THUC_HANH/BackEnd/BackEnd_Part1/ContactBook-back
end/.git/
```

Hình 6: Khởi tạo dự án Git

– Sau đó kiểm tra với " **git status** "

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> git status
On branch master

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
        .gitignore
        package-lock.json
        package.json

nothing added to commit but untracked files present (use "git add" to track)
```

Hình 7: Kiểm tra với " git status "

⇨ Ta thấy có 3 tệp tin CHƯA ĐƯỢC quản lý bởi git là . gitignore, package-lock.json, package.json

– Cho git quản lý 3 dự án .gitignore, package-lock.json, package.json bằng lệnh:

git add . gitignore package-lock.json package.json

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> git add .gitignore package-lock.json package.json
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'package-lock.json', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'package.json', LF will be replaced by CRLF the next time Git touches it
```

Hình 8: git quản lý 3 dự án .gitignore, package-lock.json, package.json

– Dùng lệnh **git status** để kiểm tra lại lần nữa

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> git status
On branch master

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
        new file:   .gitignore
        new file:   package-lock.json
        new file:   package.json
```

Hình 9: Kiểm tra với " git status "

⇨ Ta thấy 3 tệp tin ĐÃ ĐƯỢC quản lý bởi git là . gitignore, package-lock.json, package.json

– Thực thi lệnh: **git commit -m "Cài đặt dự án"**

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> git commit -m "Cai dat du an"
[master (root-commit) a48dc13] Cai dat du an
 3 files changed, 1002 insertions(+)
 create mode 100644 .gitignore
 create mode 100644 package-lock.json
 create mode 100644 package.json
```

Hình 10: Đã tạo thành công

– Upload dự án lên GitHub với các lệnh:

- git remote add origin https://github.com/4MM4-052/B2207576_Tran_Truong_Ngoc_Uyen_BACKEND_1.git
- git push origin main

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> git remote add origin https://github.com/4MM4-052/B2207576_Tran
_Truong_Ngoc_Uyen_BACKEND_1.git
error: remote origin already exists.
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> git push origin master
Everything up-to-date
```

Hình 11: Upload dự án lên GitHub

– Thực hiện chứng thực tài khoản và kiểm tra lại các tập tin đã được tải lên GitHub

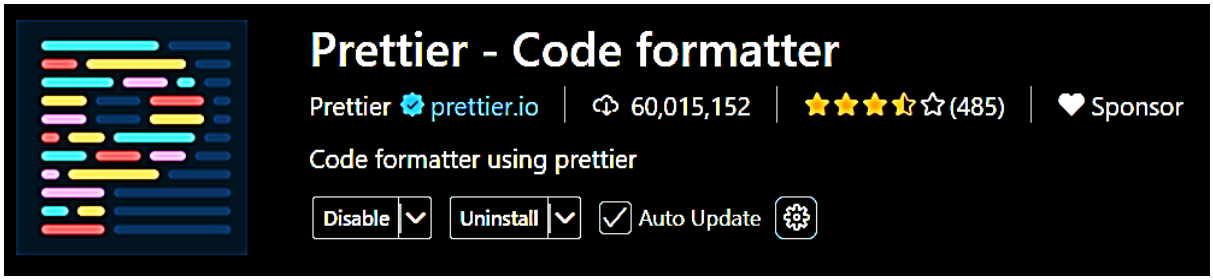
Hình 12: Chứng thực tài khoản và kiểm tra trên GitHub

Bước 3: Cấu hình Visual code, ESLint và Prettier

– Cài đặt ESLint extension. ESLint là một công cụ phân tích tĩnh mã Javascript, giúp tìm và sửa các vấn đề trong mã nguồn.

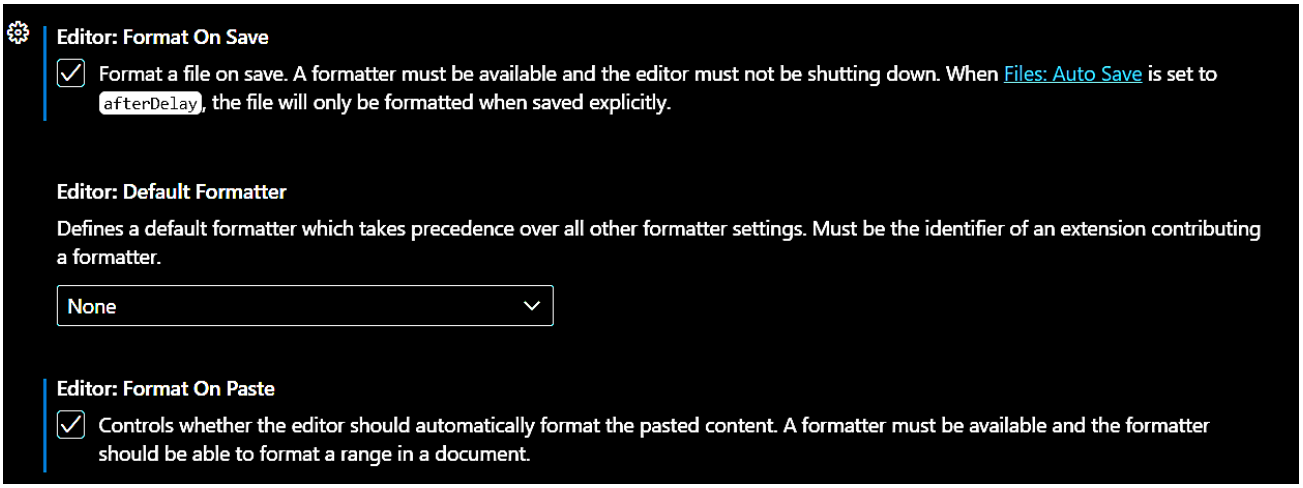
Hình 13: Cài đặt ESLint extension thành công

– Cài đặt Prettier extension. Prettier là công cụ hỗ trợ định dạng tự động các tập tin mã nguồn.



Hình 14: Cài đặt Prettier extension thành công

– Cấu hình VSCode cho phép định dạng mã lệnh khi save và paste



Hình 15: Cấu hình VSCode cho phép định dạng Save & Paste

– Cài đặt gói thư viện ESLint và Prettier với lệnh sau**: npm i -D eslint prettier eslint-config-prettier**

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> npm i -D eslint prettier eslint-config-prettier

added 86 packages, and audited 156 packages in 5s

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

found 0 vulnerabilities
```

Hình 16: Cài đặt gói thư viện ESLint và Prettier

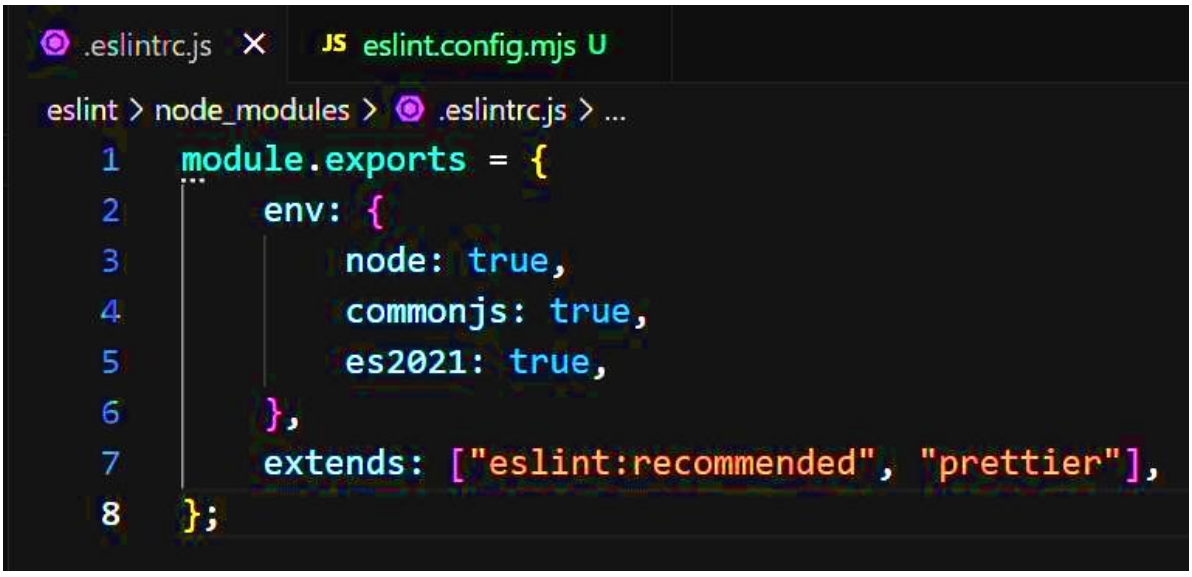
– Sau đó tạo một tập tin eslintrc.js tại thư mục gốc bằng lệnh: **npx eslint –init**

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> npx eslint --init
You can also run this command directly using 'npm init @eslint/config@latest'.
Need to install the following packages:
@eslint/create-config@1.10.0
Ok to proceed? (y)


> contactbook-backend@1.0.0 npx
> create-config

@eslint/create-config: v1.10.0

√ What do you want to lint? · javascript
√ How would you like to use ESLint? · problems
√ What type of modules does your project use? · esm
√ Which framework does your project use? · react
```



Hình 17: Tạo một tập tin eslintrc.js

– Và lưu thay đổi trên git

```
git add -u
git add eslint.config.mjs
git commit -m "Cau hinh ESLint cho du an"
```

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> git add -u
warning: in the working copy of 'package-lock.json', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'package.json', LF will be replaced by CRLF the next time Git touches it
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> git add .eslint.config.mjs
fatal: pathspec '.eslint.config.mjs' did not match any files
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> git add eslint.config.mjs
warning: in the working copy of 'eslint.config.mjs', LF will be replaced by CRLF the next time Git touches it
```

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   eslint.config.mjs
        modified:   package-lock.json
        modified:   package.json
```

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> git commit -m "Cau hinh ESLint cho du an"
[master 739362f] Cau hinh ESLint cho du an
 3 files changed, 2973 insertions(+), 194 deletions(-)
 create mode 100644 eslint.config.mjs
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend>
```

Hình 18: Lưu thay đổi trên git

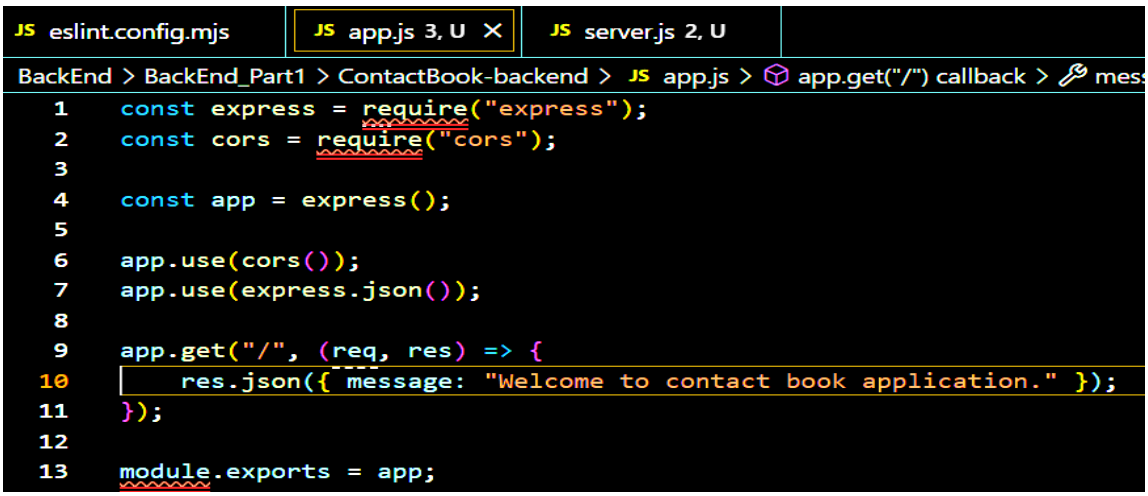
– Kiểm tra lại các tập tin đã được tải lên GitHub sau khi thực hiện lệnh **git push origin master**

Hình 19: Tệp đã tạo thành công trên GitHub

Bước 4: Cài đặt Express

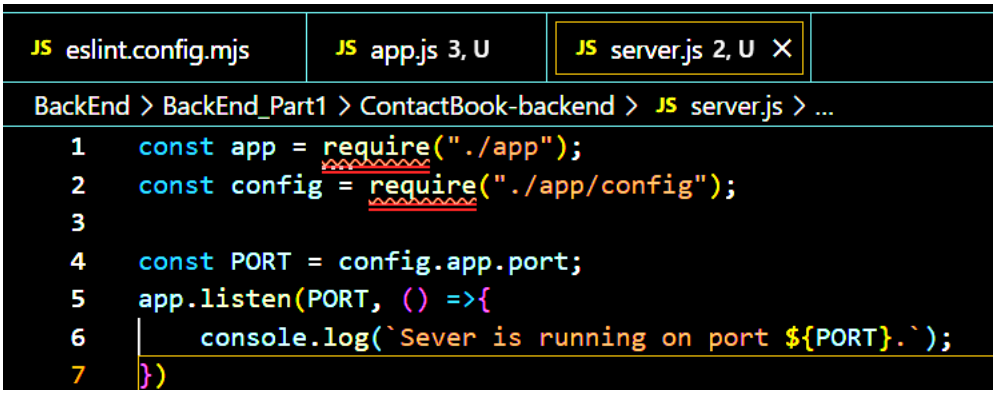
– Tạo tập tin app.js, server.js và index.js lần lượt với nội dung như sau:

- app.js



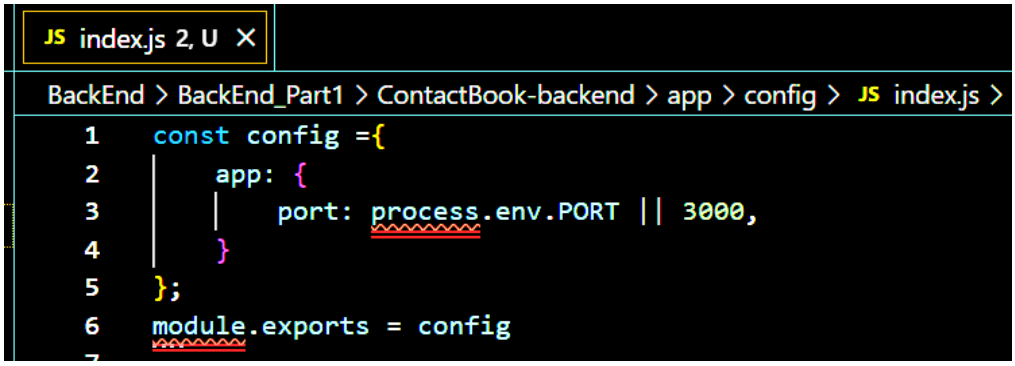
Hình 20: Tạo tập tin app.js

- server.js



Hình 21: Tạo tập tin server.js

– Lần lượt tạo thư mục app, app/config và tập tin app/config/index.js



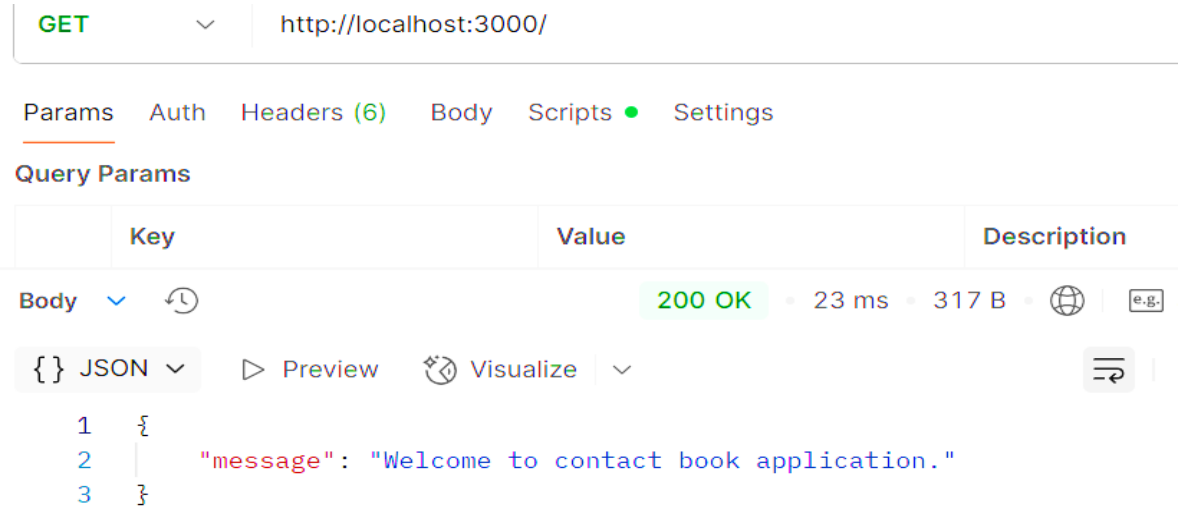
Hình 22: Tạo tập tin index.js

– Kiểm tra kết quả chạy server bằng lệnh: **node server.js**

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> node server.js
Sever is running on port 3000.
```

Hình 23: Kết quả chạy server thành công

– Kiểm tra kết quả bằng **ứng dụng Postman**



Hình 24: Kiểm tra kết quả bằng ứng dụng Posman

⇨ Kết nối thành công

– Cài đặt nodemon bằng lệnh: **npm install nodemon --save-dev**

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> npm install nodemon --save-dev

added 22 packages, and audited 274 packages in 2s

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

found 0 vulnerabilities
```

Hình 25: Cài đặt nodemon thành công

– Sau khi cài đặt, mở tập tin package.json và thay đổi cấu hình mục “scripts”

```
"scripts": {
  "test": "echo \"Error: no test specified\" && exit 1",
  "start": "nodemon server.js"
},
"author": "Uyen Tran",
```

Hình 26: Thay đổi cấu hình mục “scripts”

– Lưu thay đổi trên vào git

git add -u
git add app/ app.js server.js

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> git add -u
warning: in the working copy of 'package-lock.json', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'package.json', LF will be replaced by CRLF the next time Git touches it
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> git add app/ app.js server.js
```

Hình 27: Lưu thay đổi trên vào git

– Kiểm tra lại bằng lệnh **git status**

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> git status
On branch master
Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        new file:   app.js
        modified:   package-lock.json
        modified:   package.json
        new file:   server.js
```

Hình 28: Các tệp đã chạy thành công

– Thực hiện lệnh **git commit -m** để lưu các thay đổi lên git

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part1\ContactBook-backend> git commit -m "Cai dat express chay thong bao xin chao"
[master 5763349] Cai dat express chay thong bao xin chao
 5 files changed, 332 insertions(+), 1 deletion(-)
 create mode 100644 app.js
 create mode 100644 app/config/index.js
 create mode 100644 server.js
```

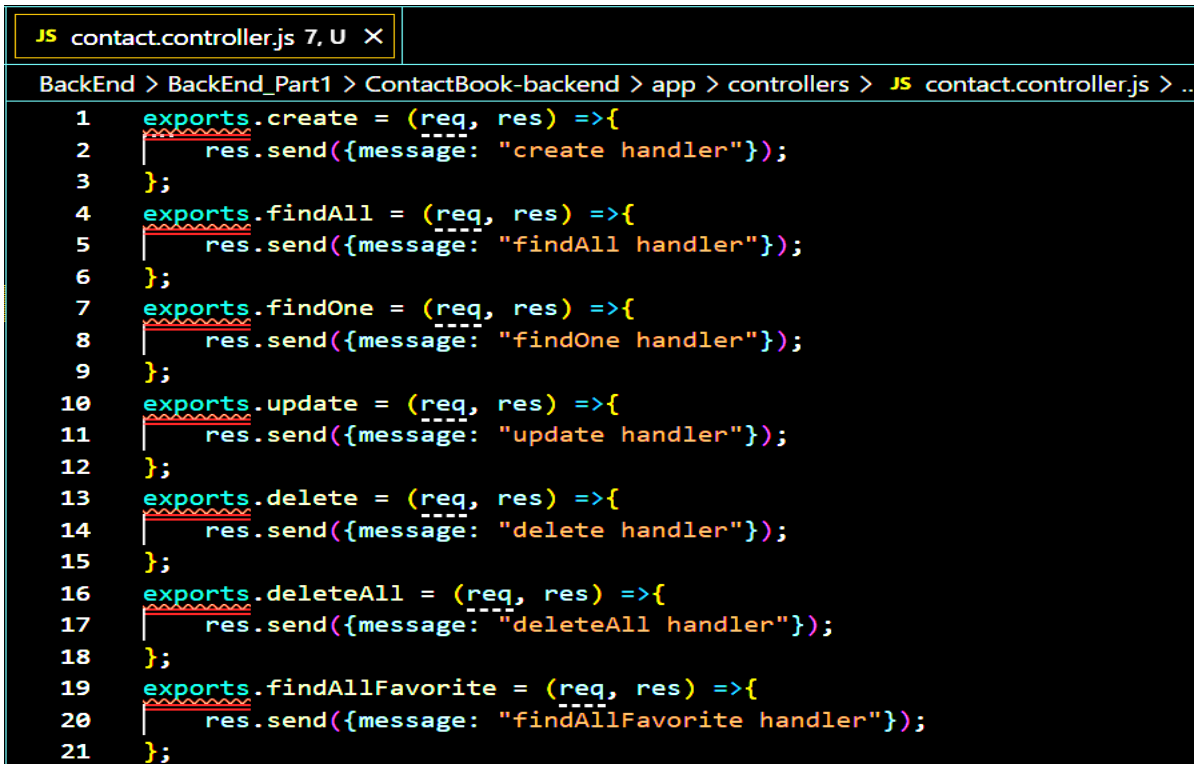
Hình 29: Được tạo thành công

– Kiểm tra lại các tập tin đã được tải lên GitHub sau khi thực hiện lệnh **git push**

Hình 30: Tệp đã tải lên GitHub thành công

Bước 5: Định nghĩa controller và các router

– Tạo thư mục controllers trong thư mục app
– Tạo tập tin contact.controller.js trong thư mục controllers



Hình 31: Tạo tập tin contact.controller.js

– Tạo thư mục routes trong thư mục app

– Tạo tập tin contact.route.js trong thư mục routes

```
JS contact.route.js 3, U ×
BackEnd > BackEnd_Part1 > ContactBook-backend > app > routes > JS contact.route.js > ...
1   const express = require("express");
2   const contacts = require("../controllers/contact.controllers");
3
4   const router = express.Router();
5
6   router.route("/")
7       .get(contacts.findAll)
8       .post(contacts.create)
9       .delete(contacts.deleteAll);
10
11  router.route("./favorite")
12      .get(contacts.findAllFavorite);
13
14  router.route("/:id")
15      .get(contacts.findOne)
16      .put(contacts.update)
17      .delete(contacts.delete);
18
19  module.exports = router
```

Hình 32: Tạo tập tin contact.route.js

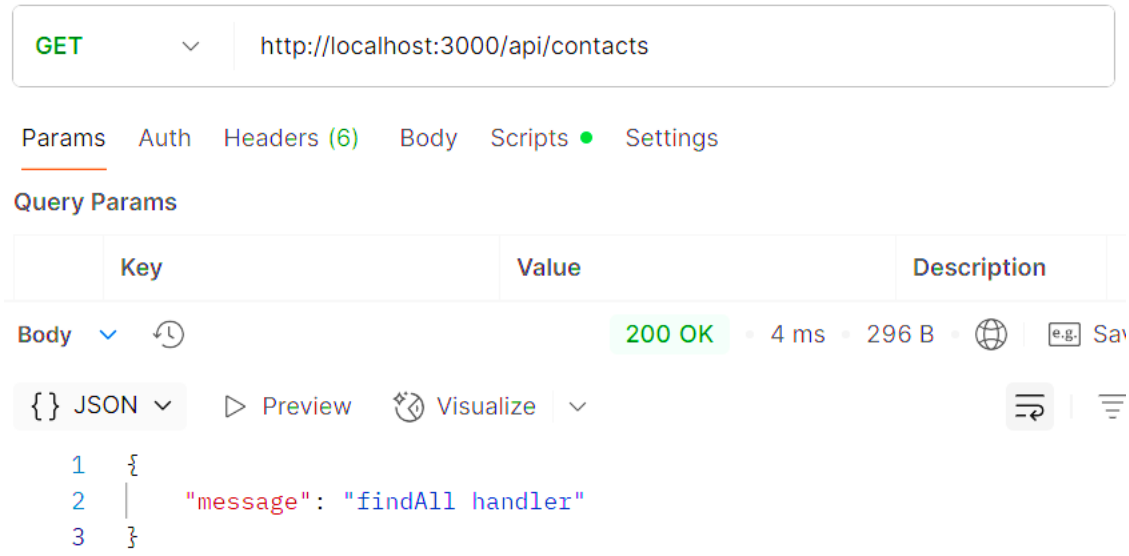
– Cập nhật lại app.js

```
JS contact.route.js 3, U   JS app.js 4, M ×
BackEnd > BackEnd_Part1 > ContactBook-backend > JS app.js > ...
1   const express = require("express");
2   const cors = require("cors");
3   const contactsRouter = require("./app/routes/contact.route");
4
5   const app = express();
6
7   app.use(cors());
8   app.use(express.json());
9   app.use("/api/contacts", contactsRouter);
10
11  app.get("/", (req, res) => {
12      res.json({ message: "Welcome to contact book application." });
13  });
14
15  module.exports = app;
```

Hình 33: Cập nhật lại app.js

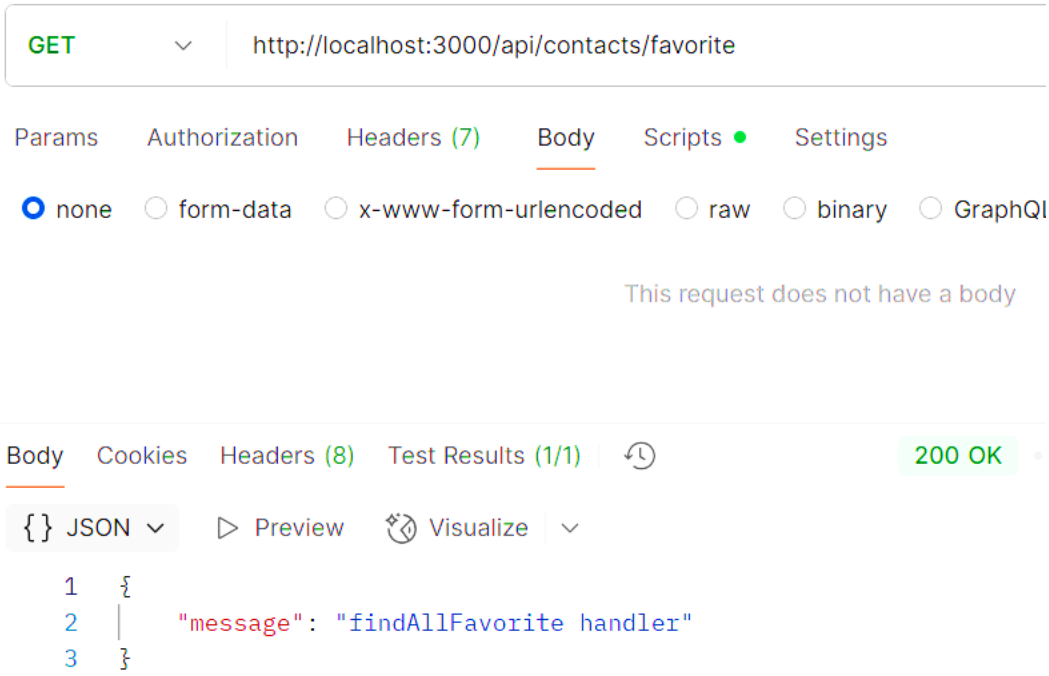
– Kiểm tra **các route** vừa tạo bằng **ứng dụng Postman**

- **Ví dụ 1**: findAll



Hình 34: Kết quả chạy thành công

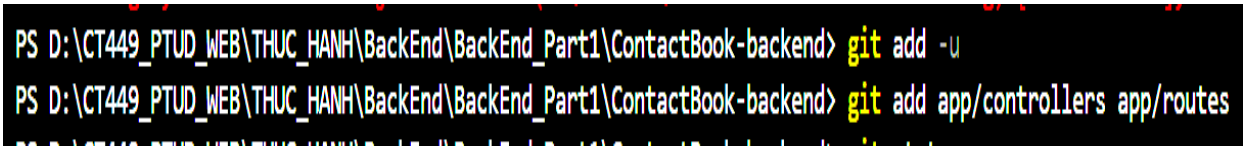
- **Ví dụ 2**: findAllFavorite



Hình 35: Kết quả chạy thành công

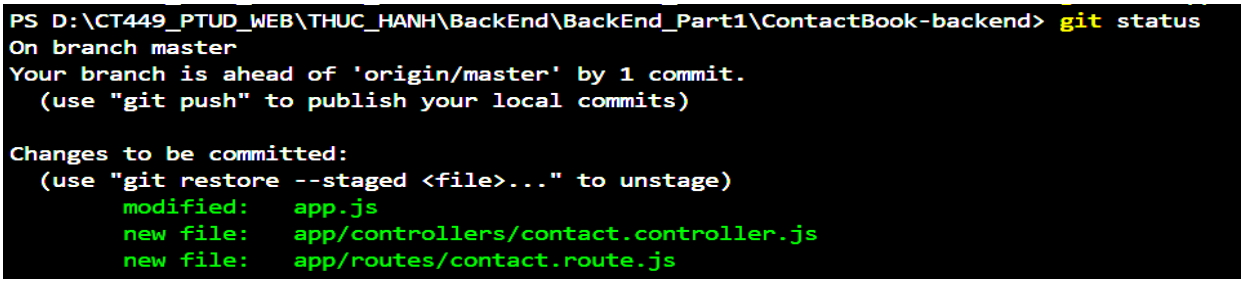
– Lưu thay đổi trên vào git

```
git add -u
git add app/controllers app/routes
```



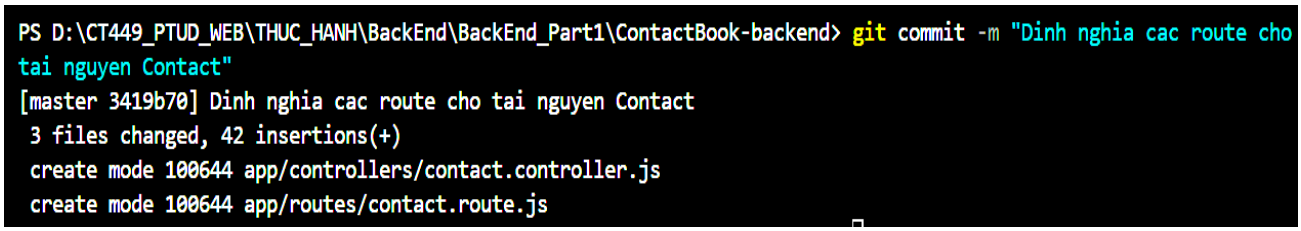
Hình 36: Lưu thay đổi trên vào git

– Kiểm tra lại bằng lệnh **git status**



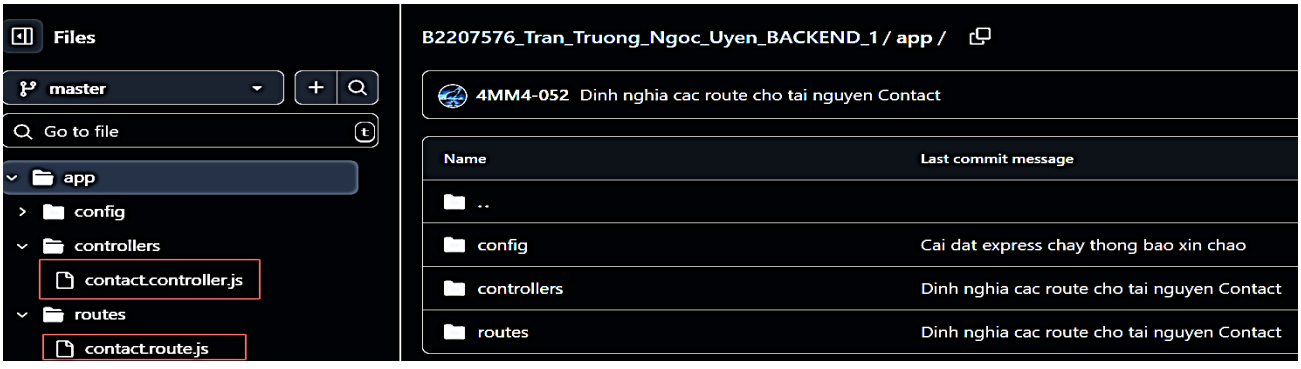
Hình 37: Các tệp đã chạy thành công

– Thực hiện lệnh **git commit -m** để lưu các thay đổi lên git



Hình 38: Được tạo thành công

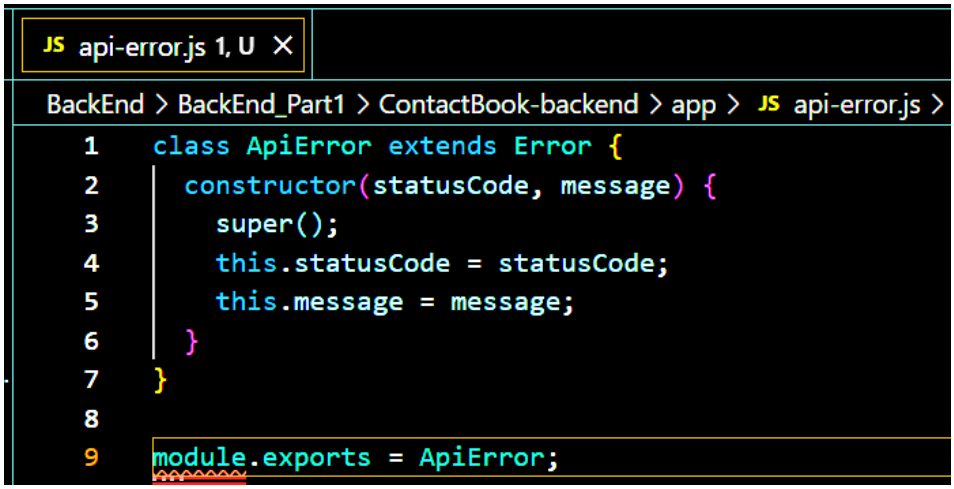
– Kiểm tra lại các tập tin đã được tải lên GitHub sau khi thực hiện lệnh **git push**



Hình 39: Tệp đã tải lên GitHub thành công

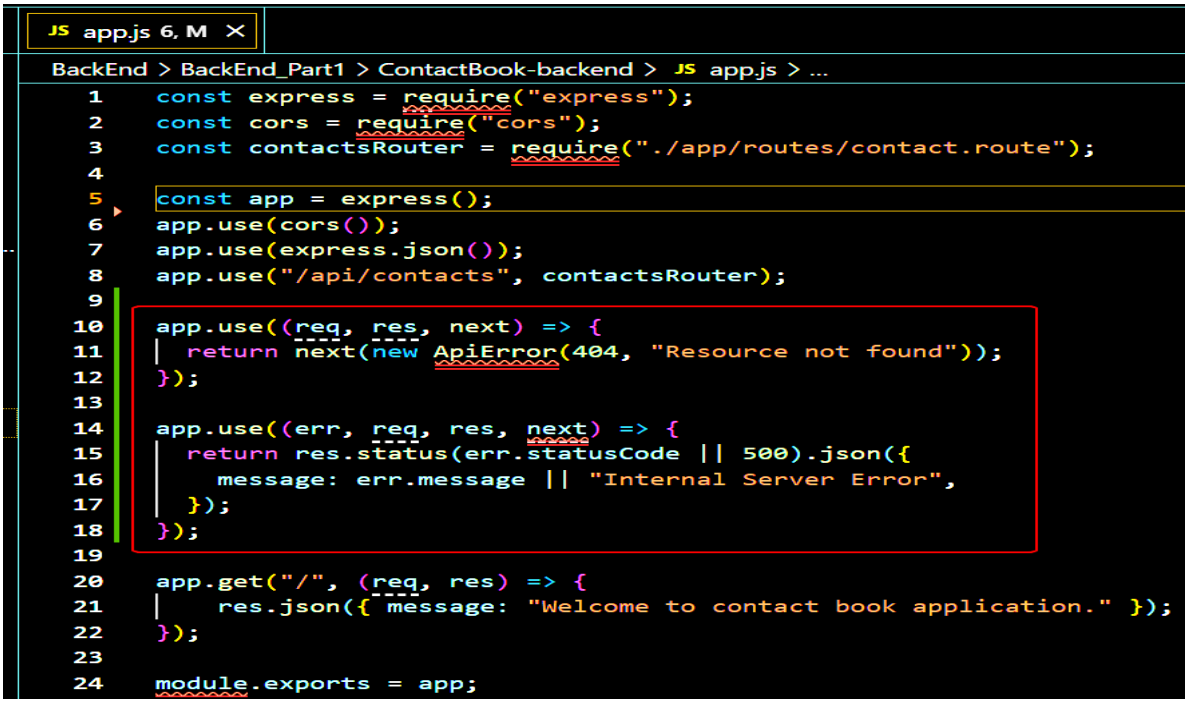
Bước 6: Cài đặt xử lý lỗi

– Tạo tập tin app/api-error.js



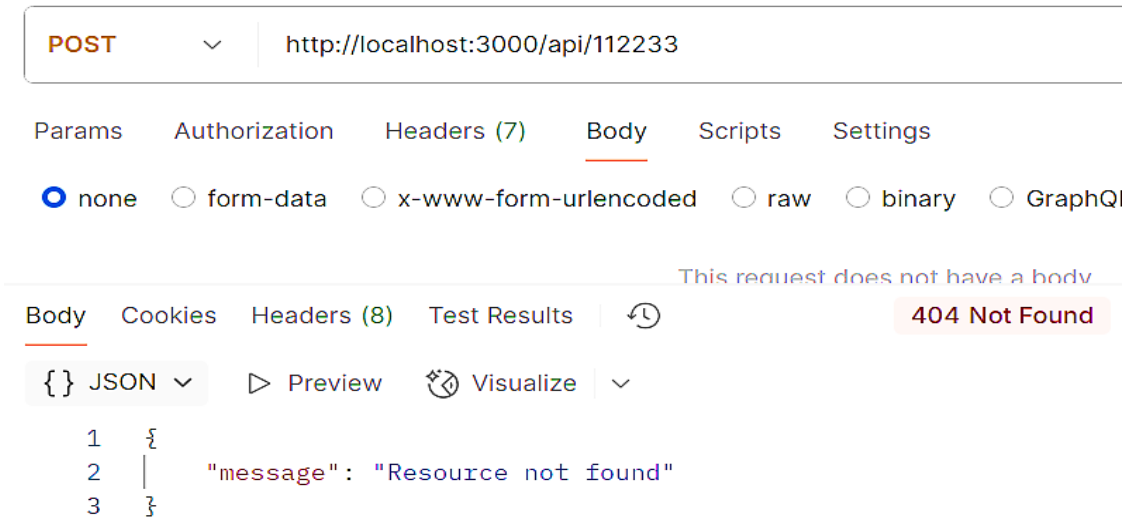
Hình 40: Tạo tập tin api-error.js

– Mở tập tin app.js thêm các middleware xử lý lỗi



Hình 41: Thêm các middleware xử lý lỗi trong app.js

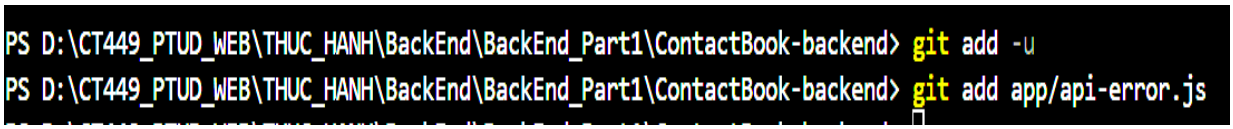
– Kiểm tra lại bằng **ứng dụng Postman**



Hình 42: Kiểm tra kết quả bằng ứng dụng Posman

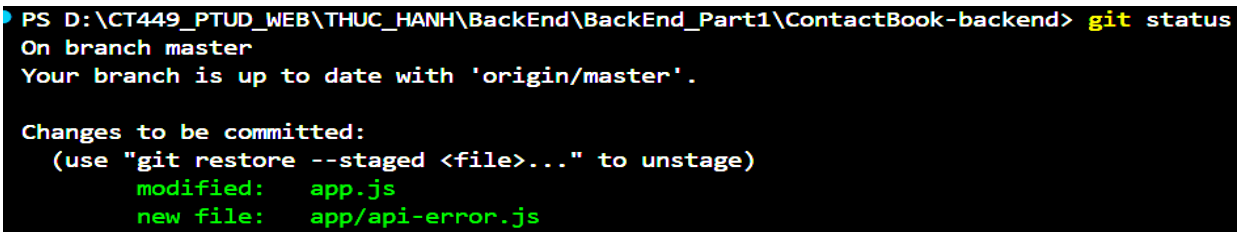
– Lưu thay đổi trên vào git

```
git add -u
git add app/api-error.js
```



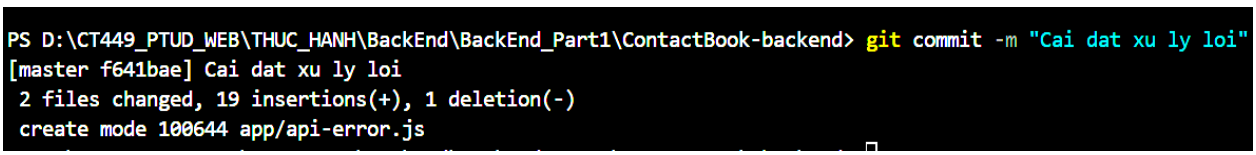
Hình 43: Lưu thay đổi trên vào git

– Kiểm tra lại bằng lệnh **git status**



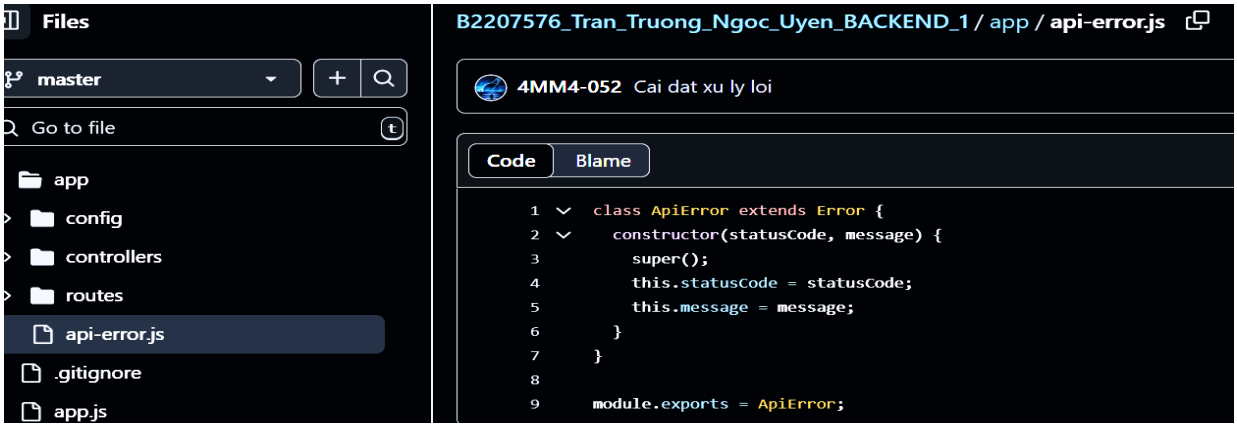
Hình 44: Các tệp đã chạy thành công

– Thực hiện lệnh **git commit -m** để lưu các thay đổi lên git



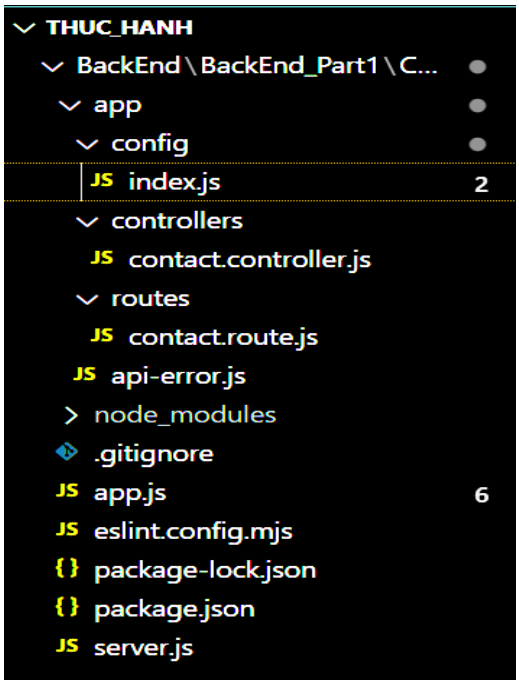
Hình 45: Được tạo thành công

– Kiểm tra lại các tập tin đã được tải lên GitHub sau khi thực hiện lệnh **git push**



Hình 46: Tệp đã tải lên GitHub thành công

– Cấu trúc thư mục dự án đến hiện tại như sau:



Hình 47: Cây thư mục hiện tại

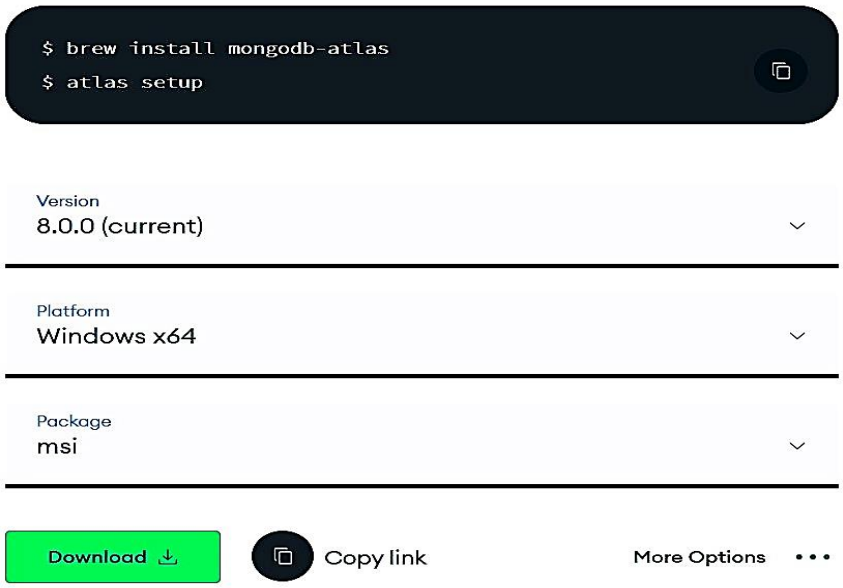
ĐƯỜNG LINK GITHUB:

Link mã nguồn:

https://github.com/4MM4-052/B2207576_Tran_Truong_Ngoc_Uyen_BACKEND_1

Ứng dụng Contactbook - Backend - Phần 2

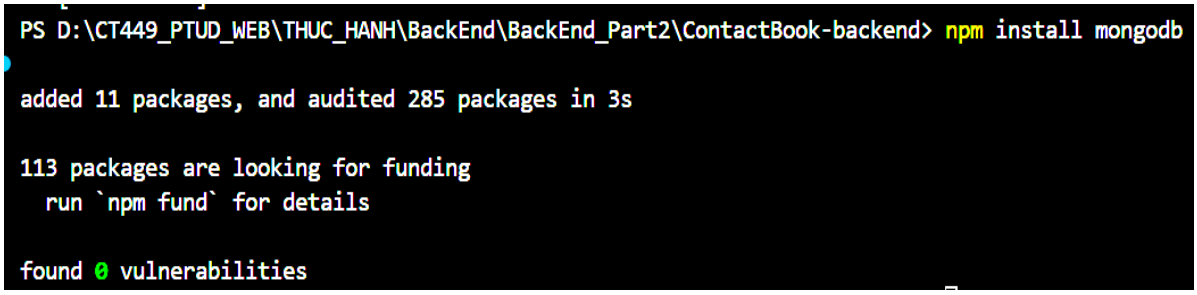
Bước 0: Cài đặt MongoDB



Hình 48: Cài đặt MongoDB

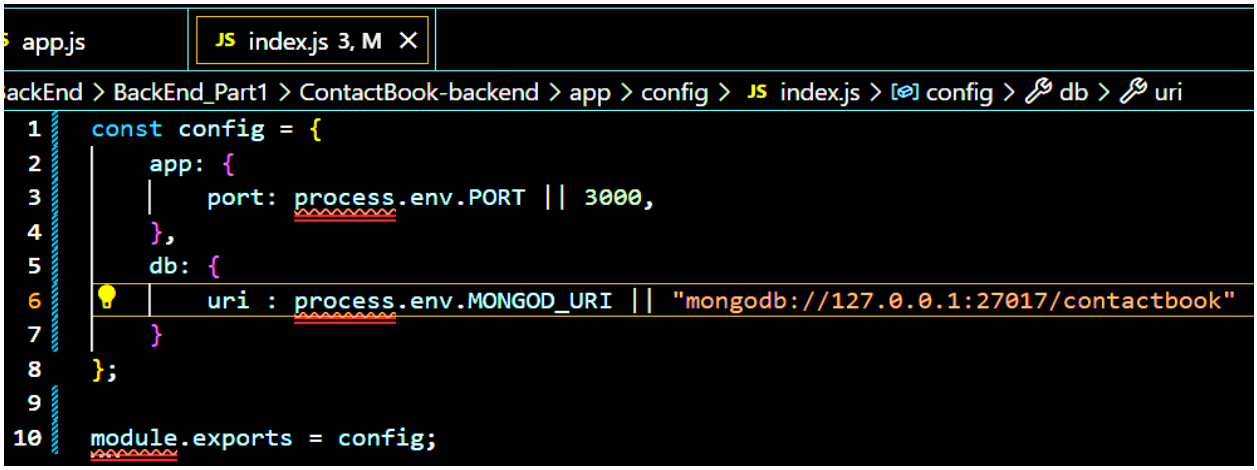
Bước 1: Cài đặt thư viên MongoDB, định nghĩa hàm trợ giúp kết nối và lớp dịch vụ truy xuất cơ sở dữ liệu (CSDL)

– Cài đặt thư viện mongodb vào dự án: **npm install mogodb**



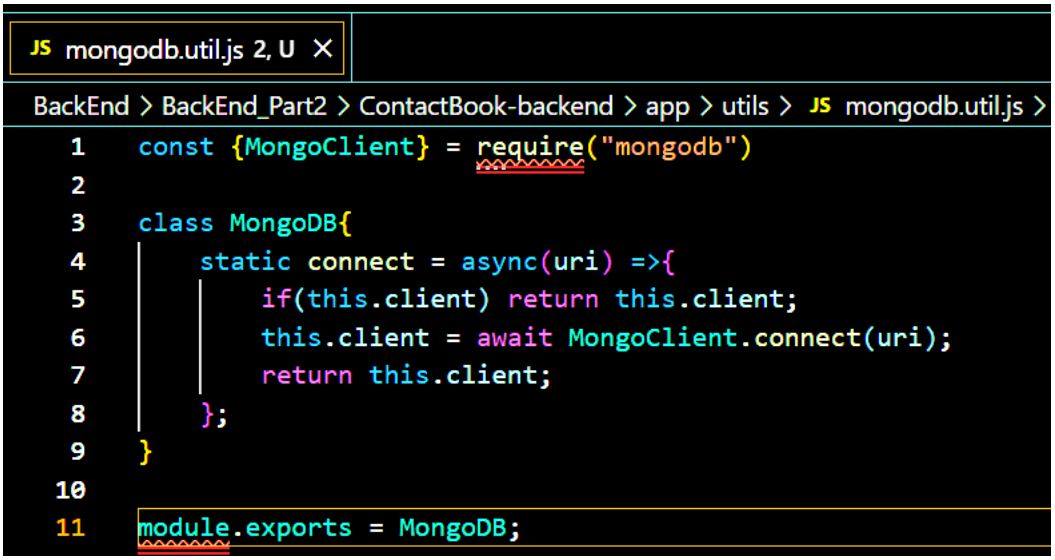
Hình 49: Cài đặt thư viện mongodb

– Trong thư mục app/config, hiệu chỉnh thêm đoạn code trong tập tin index,js



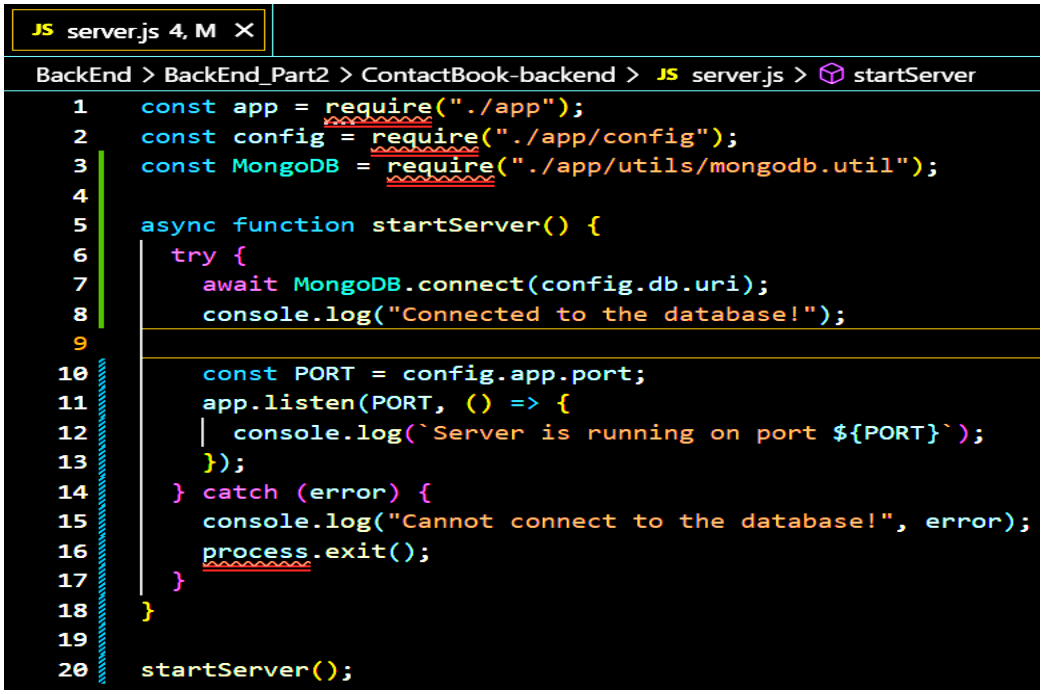
Hình 50: Hiệu chỉnh thêm đoạn code trong tập tin index,js

– Định nghĩ lớp trợ giúp kết nối đến mongoDB: app/utils/mongodb.util.js



Hình 51: Tạo tập tin mongodb.utils.js

– Thay toàn bộ tập tin server.js



Hình 52: Thay đổi nôi dung tập tin server.js

– Định nghĩa các lớp dịch vụ ContactServices (trong tập tin app/services/service.mjs) chứa các API của thư viện MongoDB để thực hiện thao tác với CSDL MongoDB

Hình 53: Định nghĩa các lớp dịch vụ ContactServices

Bước 2: Cài đặt các handler

1. Cài đặt handler create

– Hiệu chỉnh tập tin app/controllers/contact.controller.js

```
JS contact.controller.js 9+, M
BackEnd > BackEnd_Part2 > ContactBook-backend > app > controllers > JS contact.controller.js > ...
104  exports.create = async(req,res,next) =>{
105      if(!req.body?.name){
106          return next(new ApiError(400, "name can not be emty"));
107      }
108      try{
109          const contactService = new ContactService(MongoDB.client);
110          const document = await contactService.create(req.body);
111          return res.send(document);
112      } catch(error){
113          return next(
114              new ApiError(500, "An error occurred while creting the contact")
115          );
116      }
117  };
```

Hình 54: Cài đặt handler create

– Tạo phương thức create() trong lớp ContactService

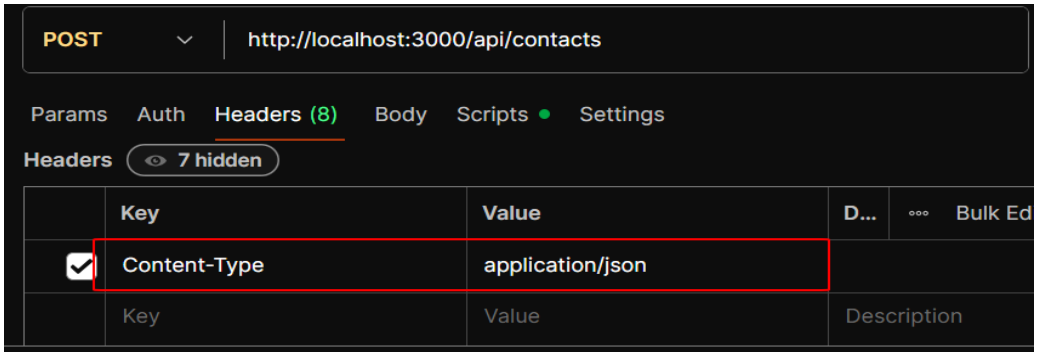
```
JS server.js 4, M    JS contact.service.js 2, U
BackEnd > BackEnd_Part2 > ContactBook-backend > app > services > JS contact.service.js > ...
1   const { ObjectId } = require("mongodb");
2
3   class ContactService {
4       constructor(client) {
5       this.Contact = client.db().collection("contacts");
6     }
7     // Định nghĩa các phương thức truy xuất CSDL sử dụng mongodb API
8     extractContactData(payload) {
9       const contact = {
10        name: payload.name,
11        email: payload.email,
12        address: payload.address,
13        phone: payload.phone,
14        favorite: payload.favorite,
15      };
16
17      // Remove undefined fields
18      Object.keys(contact).forEach(
19          (key) => contact[key] === undefined && delete contact[key]);
20
21      return contact;
22    }
```

```
24    async create(payload) {
25      const contact = this.extractContactData(payload);
26      const result = await this.Contact.findOneAndUpdate(
27        contact,
28        { $set: { favorite: contact.favorite === true } },
29        { returnDocument: "after", upsert: true }
30      );
31      return result;
32    }
```

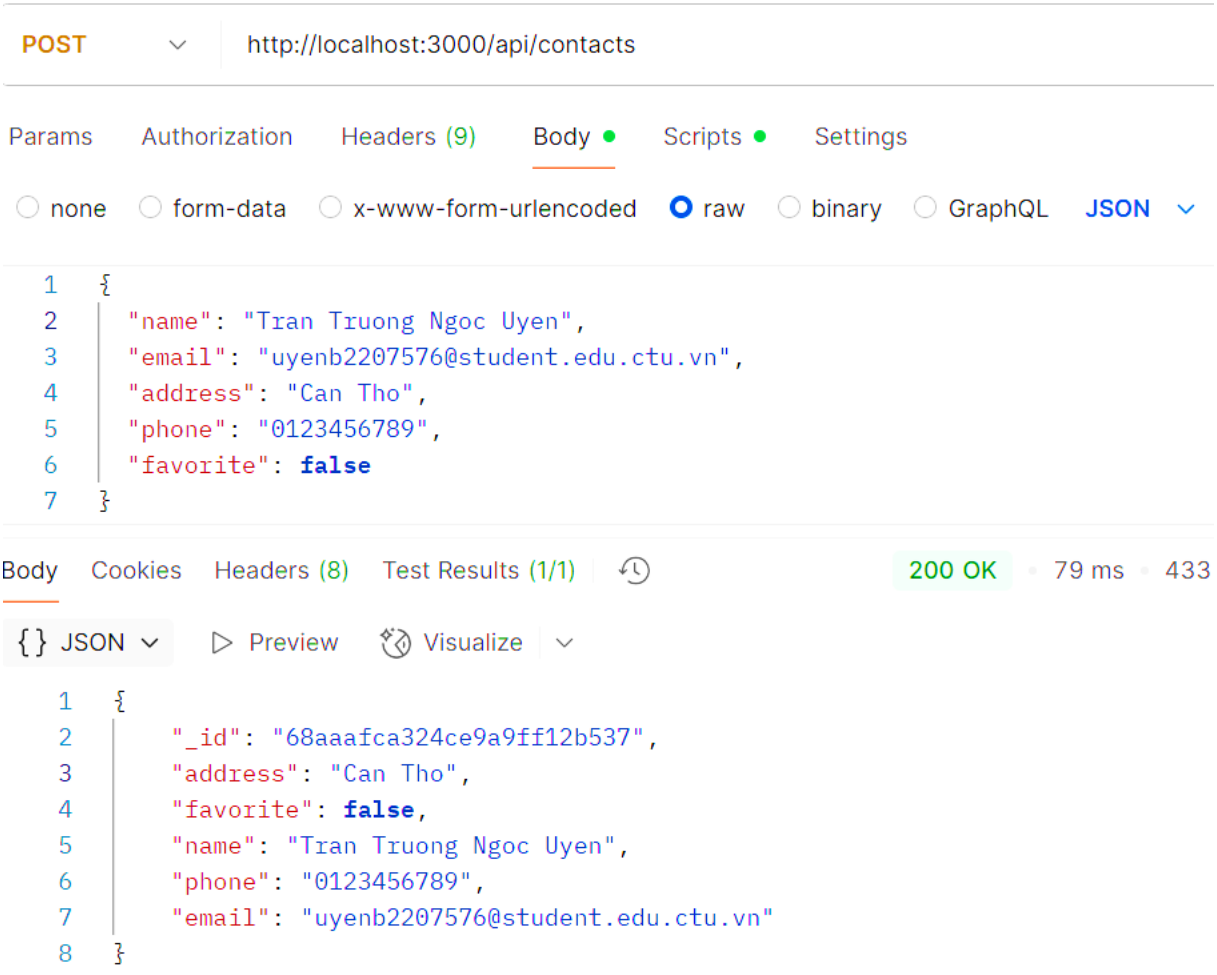
Hình 55: Tạo phương thức create()

– Kiểm tra với **ứng dụng Postman cho create**

- Để gửi dữ liệu JSON về sever với Postman, cần đặt Header “Content-Type: application/json” và đặt dữ liệu JSON trong phần Body của yêu cầu

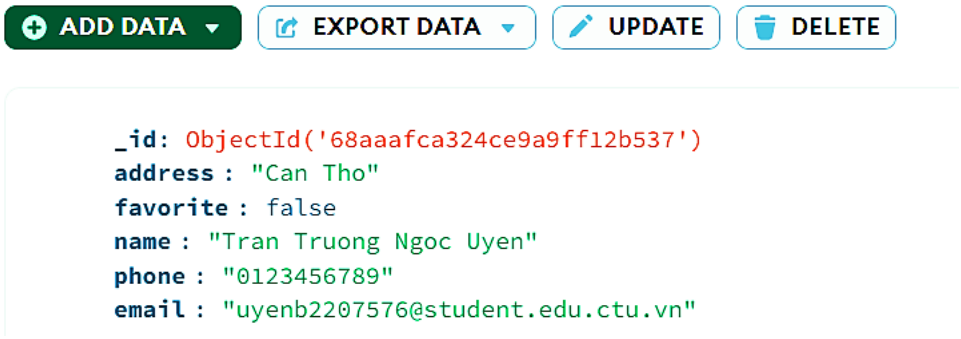


Hình 56: Hiệu chỉnh Header "Content-Type: application/json"



Hình 57: Đặt dữ liệu JSON trong phần Body

– Kết quả bên MongoDB:



Hình 58: Kết quả MongoDB

2. Cài đặt handler findAll

– Hiệu chỉnh tập tin app/controllers/contact.controller.js

```
exports.findAll = async (req, res, next)=>{
    let document = [];
    try{
        const contactService = new ContactService(MongoDB.client);
        const {name} = req.query;
        if(name){
            document = await contactService.findByName(name);

        }
        else{
            document = await contactService.find({});
        }
    } catch(error){
        return next(
            new ApiError(500, "An error occurred while retrieving contacts")
        );
    }
    return res.send(document)
}
```

Hình 59: Cài đặt handler findAll

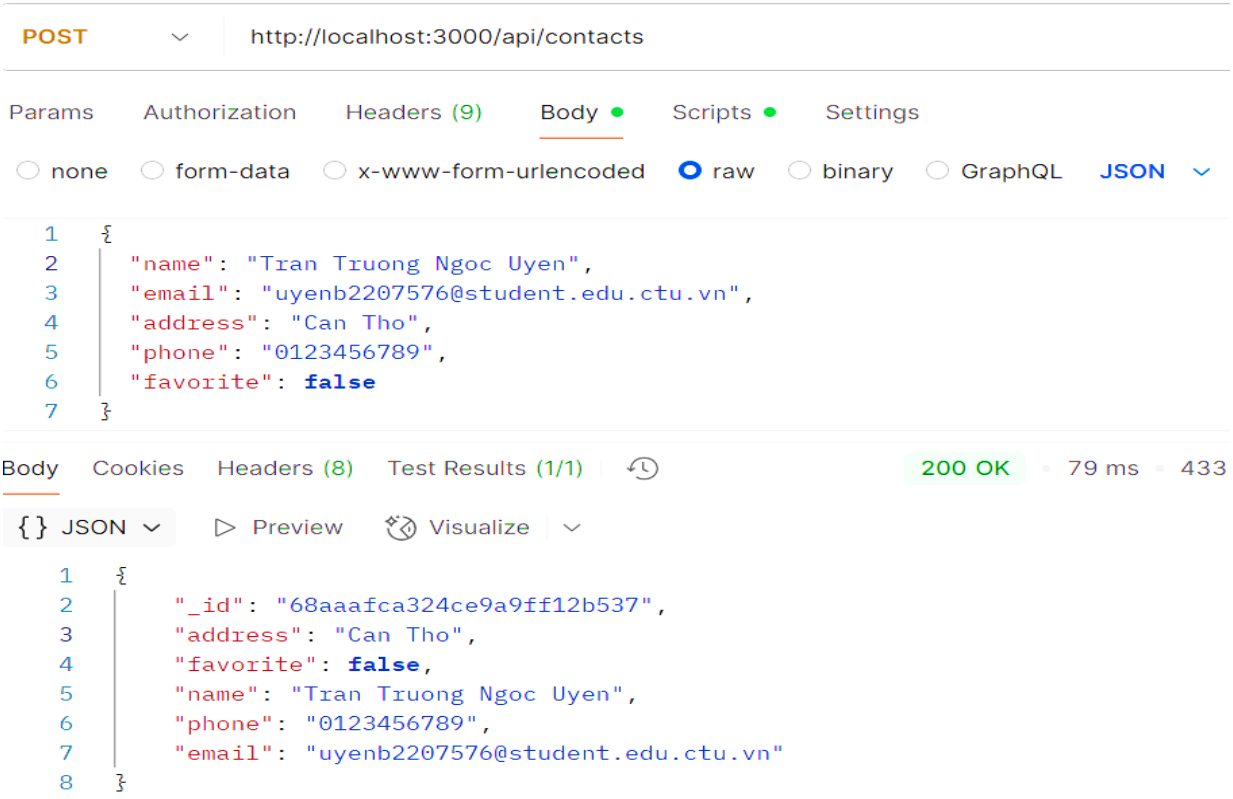
– Tạo 2 phương thức find(), findByName() trong lớp ContactService

```
async find(filter) {
  const cursor = await this.Contact.find(filter);
  return await cursor.toArray();
}

async findByName(name) {
  return await this.find({
    name: { $regex: new RegExp(name), $options: "i" },
  });
}
```

Hình 60: Tạo 2 phương thức find(), findByName()

– Kiểm tra với **ứng dụng Postman cho findAll**



Hình 61: Kết quả chạy thành công

3. Cài đặt handler findOne

– Hiệu chỉnh tập tin app/controllers/contact.controller.js

```
exports.findOne = async (req, res, next) =>{
    try{
        const contactService = new ContactService(MongoDB.client);
        const document = await contactService.findById(req.params.id);
        if(!document){
            return next(new ApiError(404, "Contact not found"));
        }
        return res.send(document);
    } catch(error){
        return next(
            new ApiError(
                500,
                `Error retrieving contact with id = ${req.params.id}`
            )
        );
    }
};
```

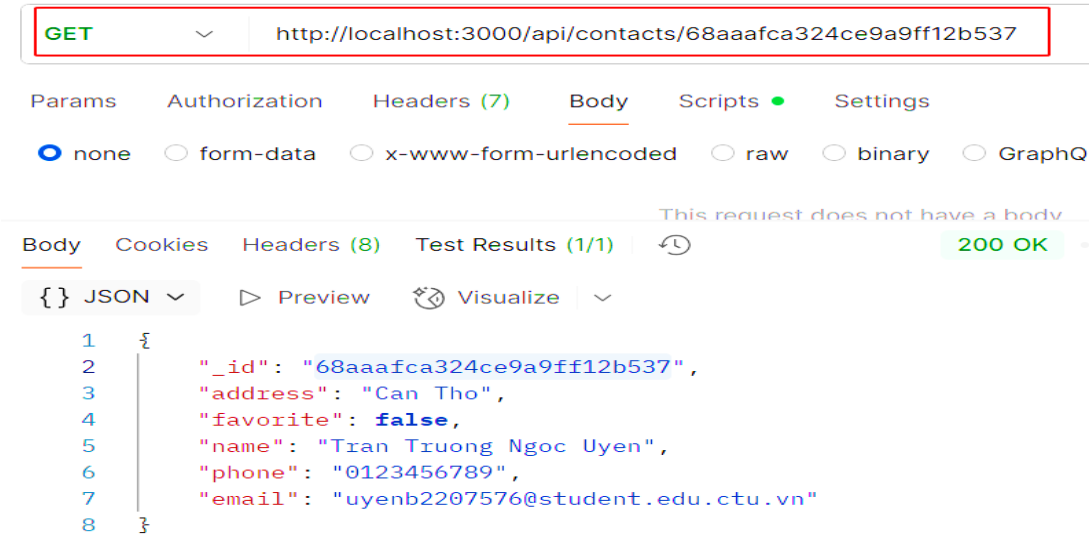
Hình 62: Cài đặt handler findOne

– Tạo phương thức findById() trong lớp ContactService

```
async findById(id) {
    return await this.Contact.findOne({
        _id: ObjectId.isValid(id) ? new ObjectId(id) : null,
    });
}
```

Hình 63: Tạo phương thức findById()

– Kiểm tra với **ứng dụng Postman cho findOne**



Hình 64: Kết quả chạy thành công

4. Cài đặt handler update

– Hiệu chỉnh tập tin app/controllers/contact.controller.js

```
exports.update = async (req, res, next) =>{
    if(Object.keys(req.body).length ===0){
        return next(new ApiError(400, "data to update can not be emty"));
    }
    try{
        const contactService = new ContactService(MongoDB.client);
        const document = await contactService.update(req.params.id, req.body);
        if(!document){
            return next(new ApiError(404, "contact not found"));
        }
        return res.send({massege: "Contact was updated succeccfuly"});
    } catch(error){
        return next(
            new ApiError(404, `Error updating contact with id =${req.params.id}`)
        );
    }
};
```

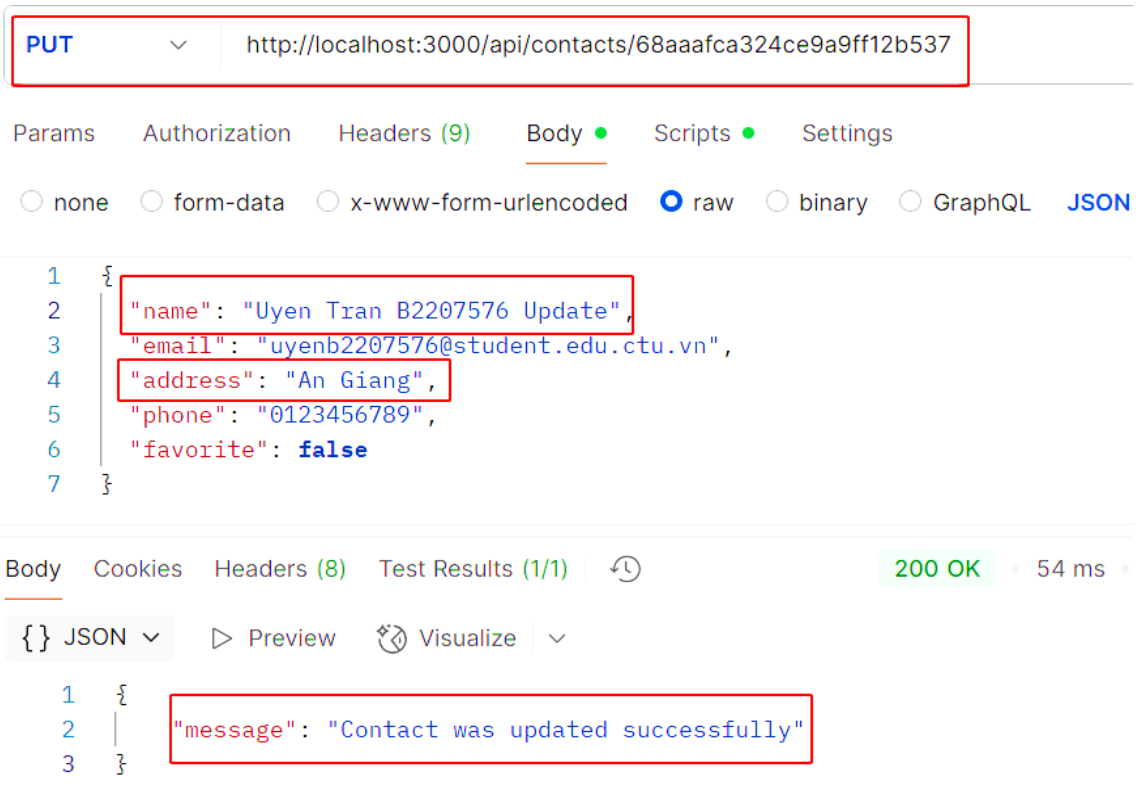
Hình 65: Cài đặt handler update

– Tạo phương thức update() trong lớp ContactService

```
async update(id, payload) {
  const filter = {
    _id: ObjectId.isValid(id) ? new ObjectId(id) : null,
  };
  const update = this.extractContactData(payload);
  const result = await this.Contact.findOneAndUpdate(
    filter,
    { $set: update },
    { returnDocument: "after" }
  );
  return result.value; //return the updated document
}
```

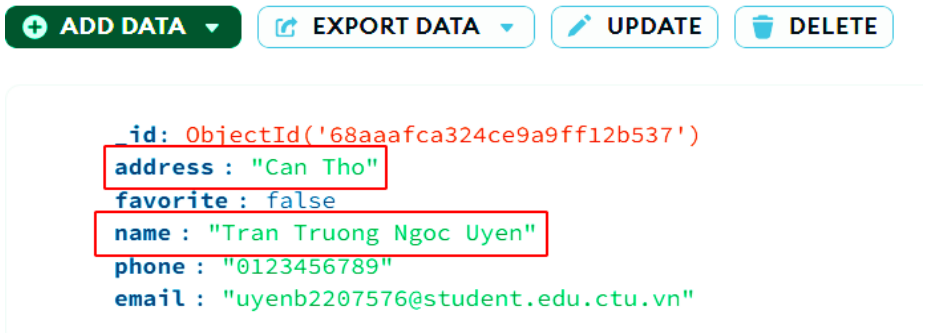
Hình 66: Tạo phương thức update()

– Kiểm tra với **ứng dụng Postman cho update**



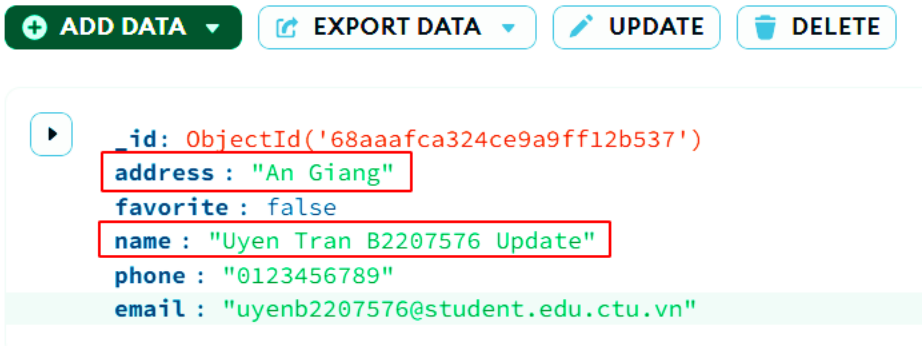
Hình 67: Kết quả chạy thành công

– Trước khi update bên MongoDB



Hình 68: Trước khi update bên MongoDB

– Sau khi update bên MongoDB



Hình 69: Sau khi update bên MongoDB

⇨ Update thành công

5. Cài đặt handler delete

– Hiệu chỉnh tập tin app/controllers/contact.controller.js

```
exports.delete = async (req, res, next) =>{
    try{
        const contactService = new ContactService(MongoDB.client);
        const document = await contactService.delete(req.params.id);
        if(!document){
            return next (new ApiError(404, " Contact not found "));
        }
        return res.send({massege: "Contact was deleted successfuly"});
    } catch(error){
        return next(
            new ApiError(
                500,
                `Could not delete contact with id =${req.params.id}`
            )
        );
    }
};
```

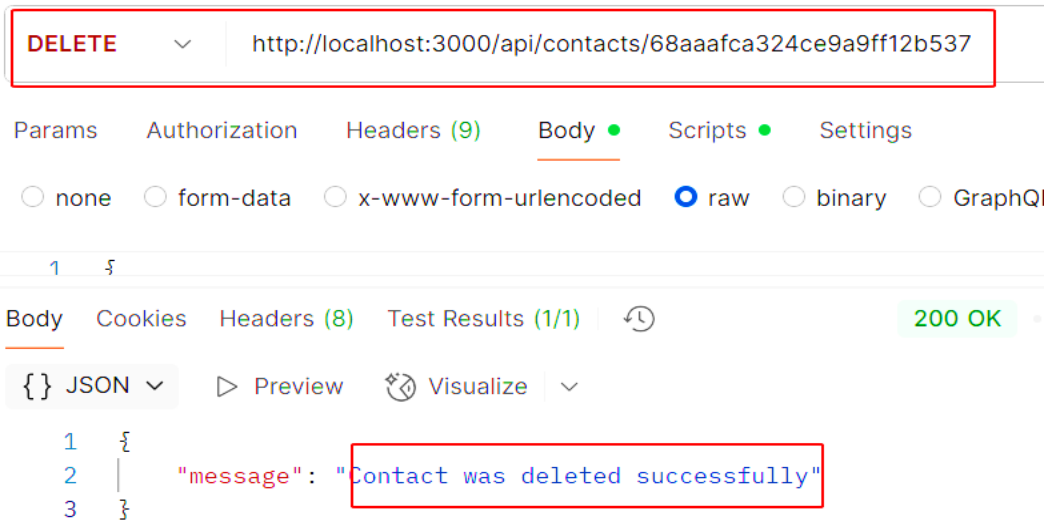
Hình 70: Cài đặt handler delete

– Tạo phương thức delete() trong lớp ContactService

```
async delete(id) {
  const result = await this.Contact.findOneAndDelete({
    _id: ObjectId.isValid(id) ? new ObjectId(id) : null,
  });
  return result;
}
```

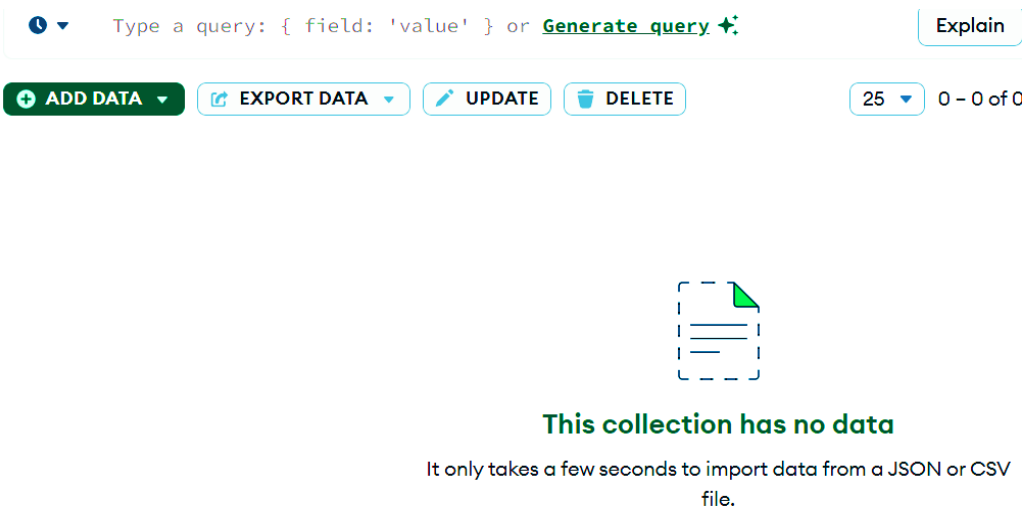
Hình 71: Tạo phương thức create()

– Kiểm tra với **ứng dụng Postman cho delete**



Hình 72: Kết quả chạy thành công

– **Kiểm tra bên MongoDB** được xóa thành công



Hình 73: Kết quả được xóa thành công

6. Cài đặt handler findAllFavorite

– Hiệu chỉnh tập tin app/controllers/contact.controller.js

```
exports.findAllFavorite = async (_req, res, next) =>{
    try{
        const contactService = new ContactService(MongoDB.client);
        const document = await contactService.findFavorite();
        return res.send(document);
    } catch(error){
        return next(
            new ApiError(
                500,
                "An error occurred while retrieving favorite contacts"
            )
        );
    }
};
```

Hình 74: Cài đặt handler findAllFavorite

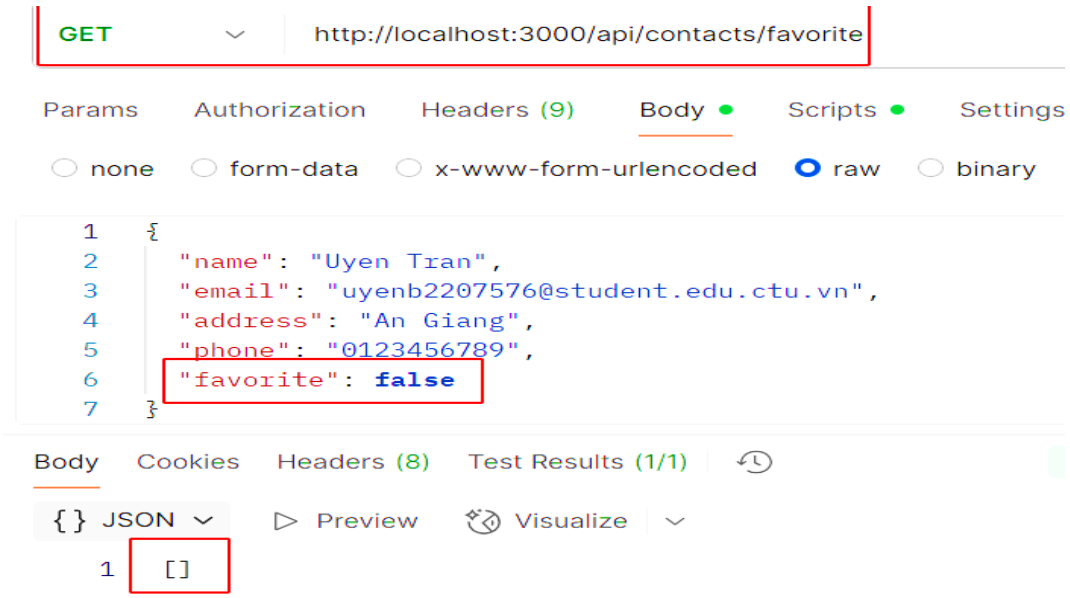
– Tạo phương thức findFavorite() trong lớp ContactService

```
async findFavorite() {
    return await this.find({ favorite: true });
}
```

Hình 75: Tạo phương thức findFavorite()

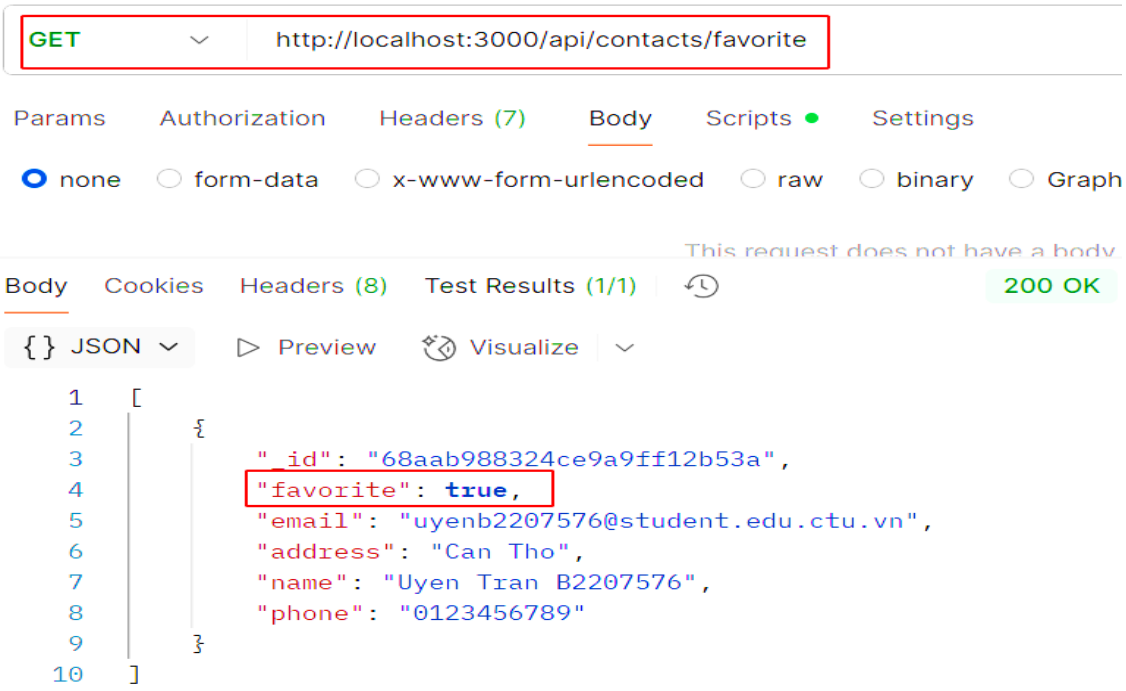
– Kiểm tra với **ứng dụng Postman cho findAllFavorite**

- Khi **favorite = false** thì kết quả sẽ là **rỗng**



Hình 76: Kết quả chạy thành công cho ra rỗng

- Khi **favorite = true** thì kết quả sẽ hiện thị toàn bộ thông tin của người đó



Hình 77: Kết quả hiện thị thành công các thông tin

7. Cài đặt handler deleteAll

– Hiệu chỉnh tập tin app/controllers/contact.controller.js

```
exports.deleteAll = async (req, res, next) =>{
    try{
        const contactService = new ContactService(MongoDB.client);
        const document = await contactService.deleteAll();
        return res.send({
            massege: `${deleteCount} contacts were deleted successfuly`,
        });
    }catch(error){
        return next(
            new ApiError(500, "An error occurred while removing all contacts")
        );
    }

};
```

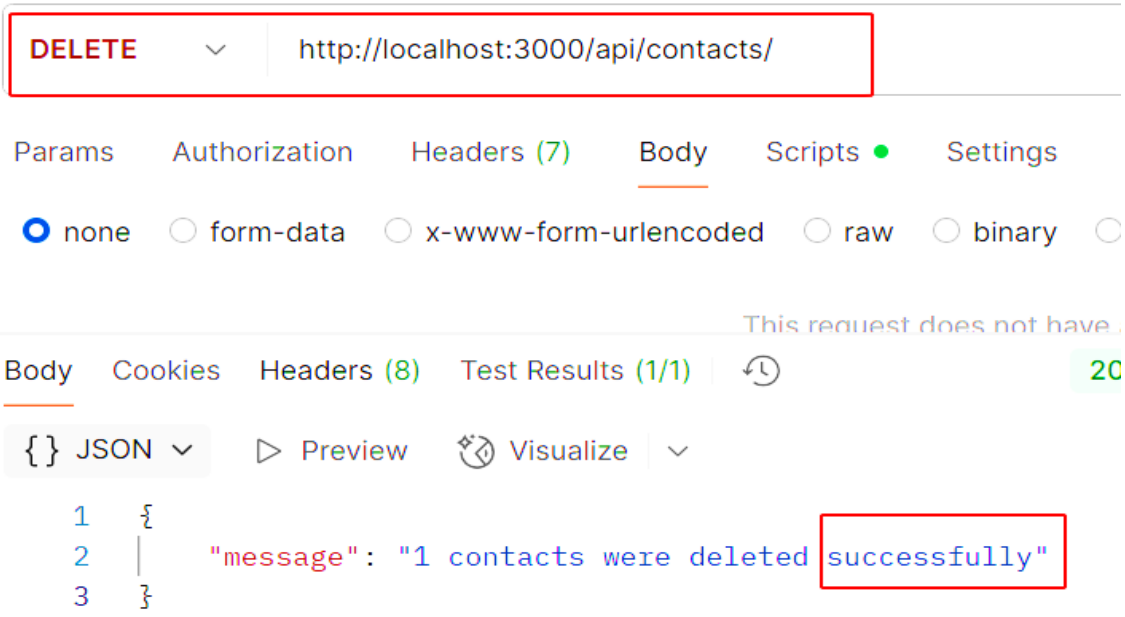
Hình 78: Cài đặt handler deleteAll

– Tạo phương thức deleteAll() trong lớp ContactService

```
  async deleteALL() {
    const result = await this.Contact.deleteMany({});
    return result.deletedCount;
  }
}
module.exports = ContactService;
```

Hình 79: Tạo phương thức deleteAll()

– Kiểm tra với **ứng dụng Postman cho deleteAll**



Hình 80: Kết quả được xóa thành công

– Lưu thay đổi trên vào git

git add -u
git add app/utils app/services

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part2\ContactBook-backend> git add -u
warning: in the working copy of '.gitignore', LF will be replaced by CRLF the next time Git touches it
warning: in the working copy of 'eslint.config.mjs', LF will be replaced by CRLF the next time Git touches it

PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part2\ContactBook-backend> git add app/utils app/services
```

Hình 81: Lưu thay đổi trên vào git

– Kiểm tra lại bằng lệnh **git status**

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part2\ContactBook-backend> git status
On branch master
Your branch is up to date with 'backup/master'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
        modified:   app.js
        modified:   app/api-error.js
        modified:   app/config/index.js
        modified:   app/controllers/contact.controller.js
        modified:   app/routes/contact.route.js
        new file:   app/services/contact.service.js
        new file:   app/utils/mongodb.util.js
```

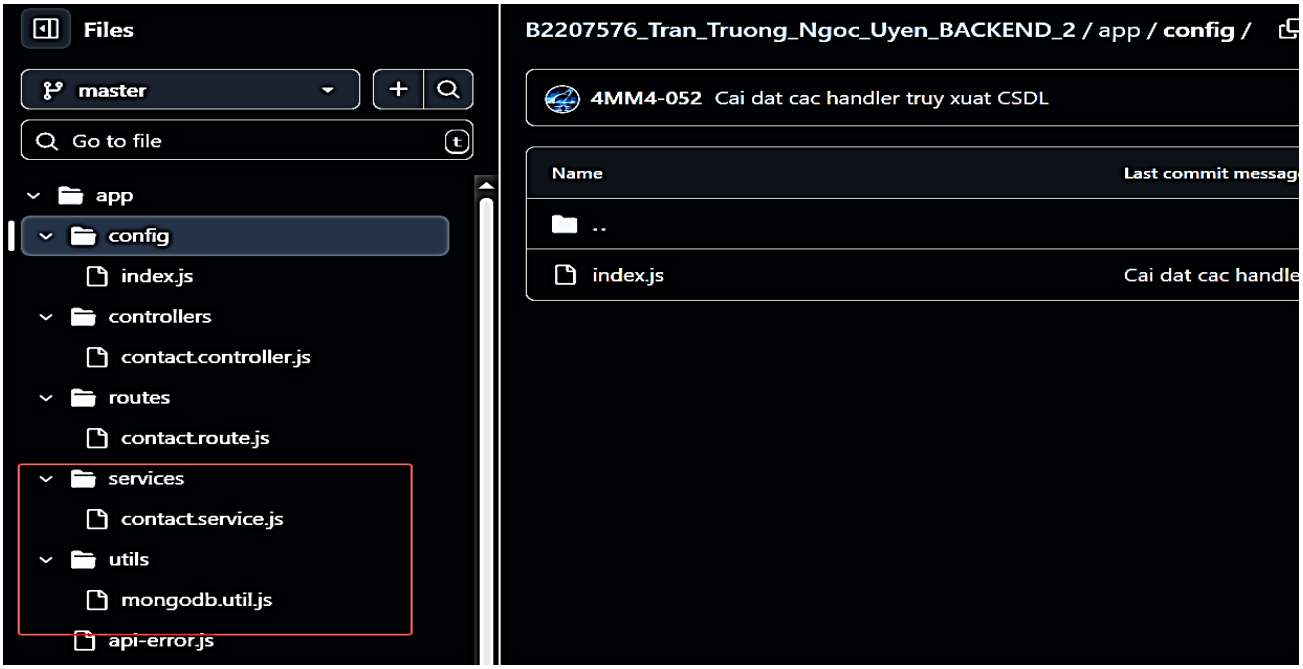
Hình 82: Các tệp đã chạy thành công

– Thực hiện lệnh **git commit -m** để lưu các thay đổi lên git

```
PS D:\CT449_PTUD_WEB\THUC_HANH\BackEnd\BackEnd_Part2\ContactBook-backend> git commit -m "Cai dat cac handler
truy xuat CSDL"
[master cb889e4] Cai dat cac handler truy xuat CSDL
 10 files changed, 420 insertions(+), 46 deletions(-)
 create mode 100644 app/services/contact.service.js
 create mode 100644 app/utils/mongodb.util.js
```

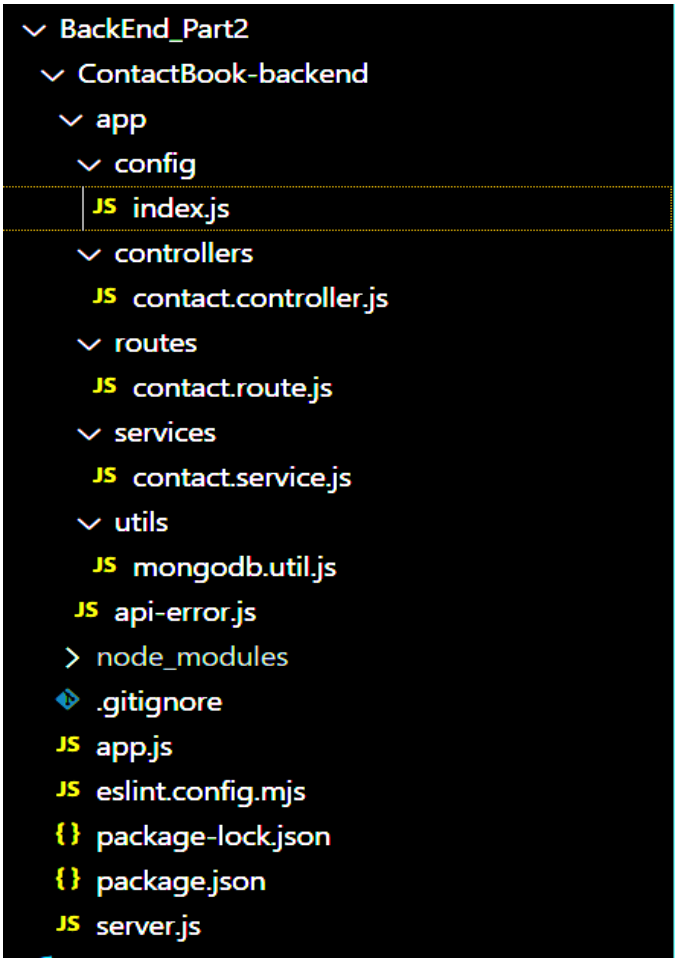
Hình 83: Được tạo thành công

– Kiểm tra lại các tập tin đã được tải lên GitHub sau khi thực hiện lệnh **git push**



Hình 84: Tệp đã tải lên GitHub thành công

– Cấu trúc thư mục dự án đến hiện tại như sau:



Hình 85: Cây thư mục hiện tại

ĐƯỜNG LINK GITHUB:

Link mã nguồn:

https://github.com/4MM4-052/B2207576_Tran_Truong_Ngoc_Uyen_BACKEND_2