

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



BÁO CÁO MÔN HỌC
PHÁT TRIỂN ỨNG DỤNG WEB

Đề tài

XÂY DỰNG ỨNG DỤNG
QUẢN LÝ DANH BẠ

Sinh viên thực hiện: Thái Phú An
Mã số: B2207512
Khoa: 48

Cần Thơ, 2/2025

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



BÁO CÁO MÔN HỌC
PHÁT TRIỂN ỨNG DỤNG WEB

Đề tài

XÂY DỰNG ỨNG DỤNG
QUẢN LÝ DANH BA

Giảng viên hướng dẫn: Sinh viên thực hiện: Thái Phú An
ThS. GVC. Nguyễn Minh Trung Mã số: B2207512
Khoa: 48

Cần Thơ, 2/2025

MỤC LỤC

1 ContactBook - Backend - Phần 1	1
1.1 Cài đặt Node và git	1
1.2 Tạo ứng dụng Node	1
1.3 Quản lý mã nguồn dự án với git và github	2
1.4 Cấu hình Visual code, ESLint và Prettier	4
1.5 Cài đặt Express	5
1.6 Định nghĩa controller và các route	8
1.7 Cài đặt xử lý lỗi	10
2 ContactBook - Backend - Phần 2	13
1.1 Cài đặt MongoDB	13
1.2 Cài đặt thư viện MongoDB, định nghĩa hàm trợ giúp kết nối và lớp dịch vụ truy xuất cơ sở dữ liệu (CSDL)	13
1.3 Cài đặt các handler	15

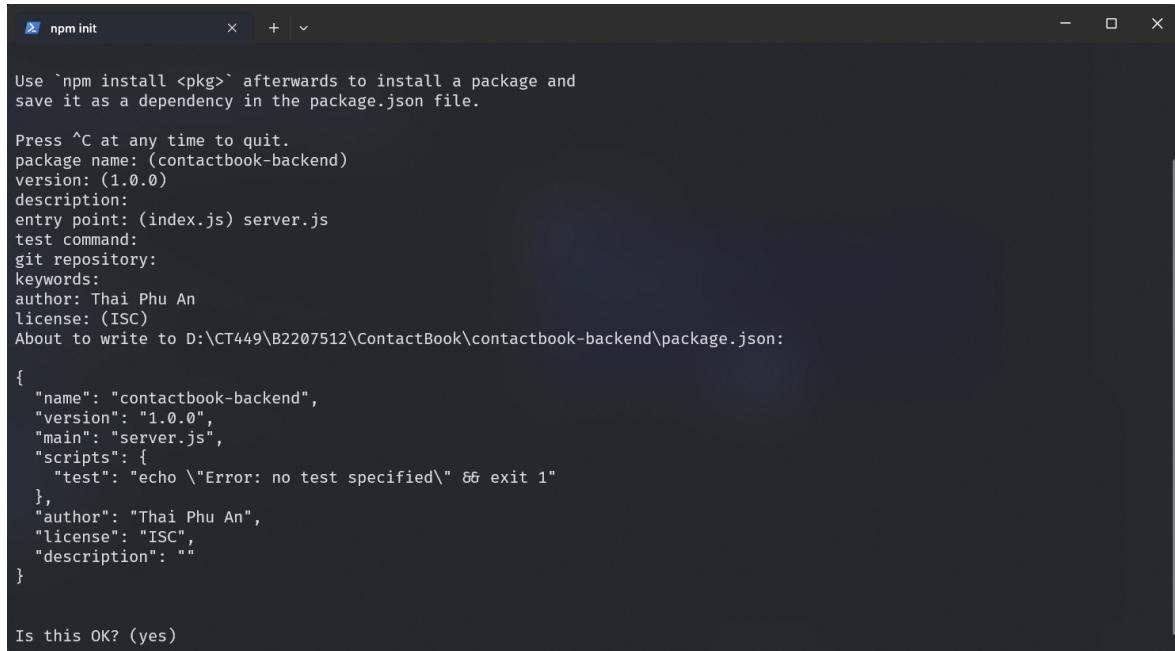
ContactBook - Backend - Phần 1

1.1 Cài đặt Node và git

Đã cài Node và git

1.2 Tạo ứng dụng Node

- Tạo thư mục dự án: contactbook-backend
 - Gõ lệnh: ‘npm init’



```
npm init

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)
version: (1.0.0)
description:
entry point: (index.js) server.js
test command:
git repository:
keywords:
author: Thai Phu An
license: (ISC)
About to write to D:\CT449\B2207512>ContactBook\contactbook-backend\package.json:

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

Is this OK? (yes)
```

- Cài đặt các package cần thiết: ‘npm install express cors’

```
> npm install express cors

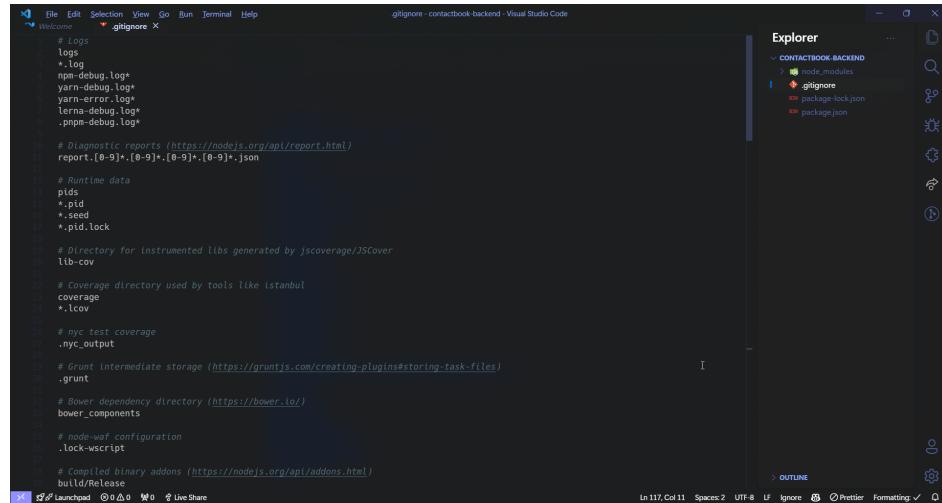
added 71 packages, and audited 72 packages in 5s

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

found 0 vulnerabilities
```

1.3 Quản lý mã nguồn dự án với git và github

- Tạo tập tin ‘.gitignore’



- Gõ lệnh: ‘git init’ để khởi tạo dự án Git. Sau đó kiểm tra với ‘git status’.

```
> git init
Initialized empty Git repository in D:/CT449/B2207512/ContactBook/contactbook-backend/.git/
> 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)

< VTOS on Tuesday at 10:19 PM ↵ master # ↵ ?3
> { □ D: ↵ CT449 ↵ B2207512 ↵ ContactBook ↵ contactbook-backend } *
```

- Cho git quản lý 3 dự án .gitignore, package-lock.json, package.json

```
> 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
> 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

< VTOS on Tuesday at 10:24 PM ↵ master # ↵ +3
> { □ D: ↵ CT449 ↵ B2207512 ↵ ContactBook ↵ contactbook-backend } *
```

- Thực thi lệnh: ‘git commit -m "Cài đặt dự án"’

```
> git commit -m "Cài đặt dự án"
[master (root-commit) 38cceff] Cài đặt dự án
  3 files changed, 1001 insertions(+)
  create mode 100644 .gitignore
  create mode 100644 package-lock.json
  create mode 100644 package.json

< VTOS on Tuesday at 10:27 PM ↵ master #
> { □ D: ↵ CT449 ↵ B2207512 ↵ ContactBook ↵ contactbook-backend } *
```

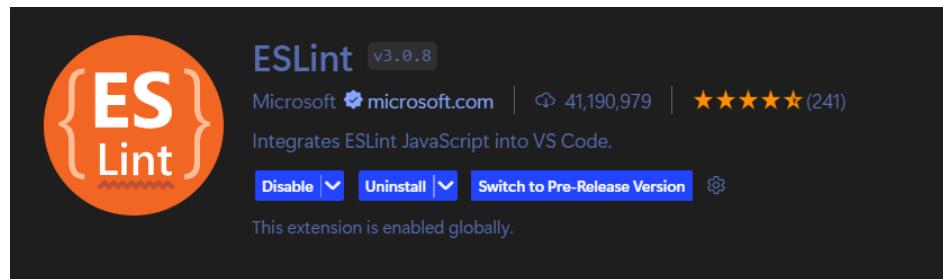
- Upload dự án lên GitHub với các lệnh:

```
git remote add origin git@github.com:p4wndev/B2207512_Thai_Phu_An_BACKEND_1.  
git push origin master
```

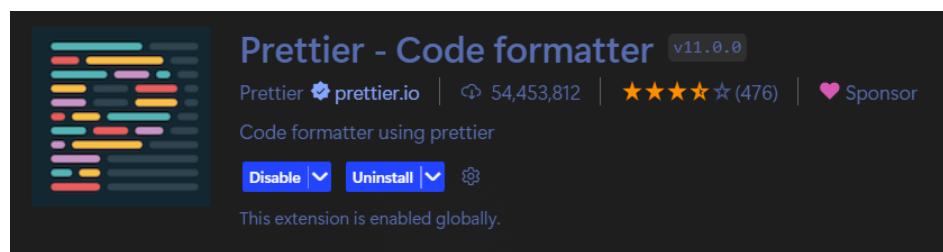
```
> git remote add origin git@github.com:p4wndev/B2207512_Thai_Phu_An_BACKEND_1.git  
> git push origin master  
Enumerating objects: 5, done.  
Counting objects: 100% (5/5), done.  
Delta compression using up to 8 threads  
Compressing objects: 100% (5/5), done.  
Writing objects: 100% (5/5), 9.25 KiB | 860.00 KiB/s, done.  
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)  
To github.com:p4wndev/B2207512_Thai_Phu_An_BACKEND_1.git  
 * [new branch]      master → master  
  
< VTOS on Tuesday at 10:40 PM ⚡ master ≠  
> { D: ➔ CT449 ➔ B2207512 ➔ ContactBook ➔ contactbook-backend } *
```

1.4 Cấu hình Visual code, ESLint và Prettier

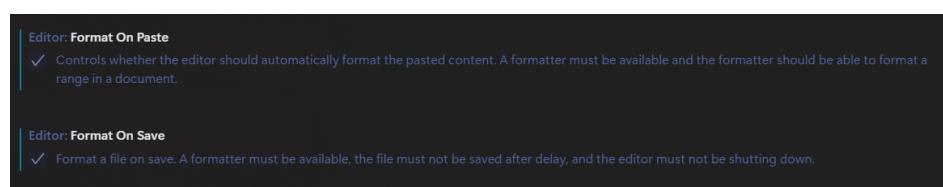
- Cài đặt ESLint extension. ESLint là một công cụ phân tích tinh tinh mã Javascript, giúp tìm và sửa các vấn đề trong mã nguồn.



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



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



- Cài đặt gói thư viện ESLint và Prettier với lệnh sau: npm i -D eslint prettier eslint-config-prettier
sau đó tạo một tập tin eslintrc.js : npx eslint --init

```

module.exports = {
  env: {
    node: true,
    comonjs: true,
    es2021: true,
  },
  extends: ["eslint:recommended", "prettier"],
};

```

Và lưu thay đổi trên git:

```

git add -u
git add .eslintrc.js
git commit -m "Cau hinh ESLint cho du an"

```

```

> 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
> git add .eslintrc.js
> git commit -m "Cau hinh ESLint cho du an"
[master d562b0c] Cau hinh ESLint cho du an
 3 files changed, 1154 insertions(+)
create mode 100644 .eslintrc.js
< VTOS on Wednesday at 12:15 PM o p master # @ ?1
> { D: CT449 B2207512 ContactBook contactbook-backend } *
0.07s 34 MEM: 85% (6/7GB)

```

1.5 Cài đặt Express

- app.js

```

const express = require('express');
const cors = require('cors');

const app = express();

app.use(cors());
app.use(express.json());

app.get('/', (req, res) => {
  res.json({ message: 'Welcome to contact book application.' });
});

module.exports = app;

```

- server.js

```
const app = require('./app');
const config = require('./app/config');

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

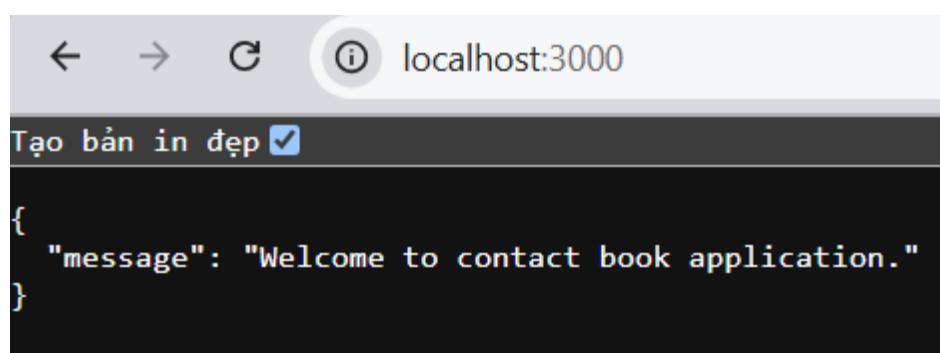
Lần lượt tạo các thư mục app, app/config và app/config/index.js

```
const config = {
  app: {
    port: process.env.PORT || 3000
  },
};

module.exports = config;
```

Mở cửa sổ terminal tại thư mục gốc của dự án và thực hiện lệnh chạy server: node server.js. Sau đó truy cập http://localhost:3000/ để kiểm tra kết quả.

```
> node server.js
Server is running on port 3000.
```



Để dễ dàng hơn cho việc phát triển code, chúng ta có thể cài đặt thêm gói hỗ trợ nodemon. Gói nodemon này sẽ giám sát các thay đổi trên mã nguồn và tự động khởi động server.

```
> npm install nodemon --save-dev
added 23 packages, and audited 186 packages in 4s
41 packages are looking for funding
  run `npm fund` for details
found 0 vulnerabilities

< VTOS on Wednesday at 12:30 PM  o  master  #  ?4 ~2
> { D: CT449 B2207512 ContactBook contactbook-backend } *
```

Sau khi cài đặt, mở tệp package.json và thay đổi cấu hình mục 'script' như sau:

```
{
  "name": "contactbook-backend",
  "version": "1.0.0",
  "main": "server.js",
  "scripts": {
    "start": "nodemon server.js"
  },
  "author": "Thai Phu An",
  "license": "ISC",
  "description": "",
  "dependencies": {
    "cors": "^2.8.5",
    "express": "^4.21.2"
  },
  "devDependencies": {
    "eslint": "^9.21.0",
    "eslint-config-prettier": "^10.0.1",
    "nodemon": "^3.1.9",
    "prettier": "^3.5.2"
  }
}
```

Và lưu các thay đổi vào git.

```
> git add -
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
> git add app/ server.js
> git commit -m "Cai dat express chay thong bao xin chao"
[master 8775c34] Cai dat express chay thong bao xin chao
 5 files changed, 342 insertions(+), 1 deletion(-)
  create mode 100644 app.js
  create mode 100644 app/config/index.js
  create mode 100644 server.js
0.069s  MEM: 89% (6/7GB)

< VTOS on Wednesday at 12:34 PM  o  master  #  ?1
> { D: CT449 B2207512 ContactBook contactbook-backend } *
```

1.6 Định nghĩa controller và các route

Tạo thư mục controllers trong thư mục app. Sau đó tạo tập tin contact.controller.js trong thư mục controllers với nội dung sau:

```
exports.create = (req, res) => {
  ...
  res.send({ message: 'create handler' });
};

exports.findAll = (req, res) => {
  res.send({ message: 'findAll handler' });
};

exports.findOne = (req, res) => {
  res.send({ message: 'findOne handler' });
};

exports.update = (req, res) => {
  res.send({ message: 'update handler' });
};

exports.delete = (req, res) => {
  res.send({ message: 'delete handler' });
};

exports.deleteAll = (req, res) => {
  res.send({ message: 'deleteAll handler' });
};

exports.findAllFavorite = (req, res) => {
  res.send({ message: "findAllFavorite handler" });
};
```

Tạo thư mục route trong thư mục app. Sau đó tạo tập tin contact.route.js trong thư mục route với nội dung sau:

```
const express = require('express');
const contact = require('../controllers/contact.controller');

const router = express.Router();

router.route('/')
  .get(contact.findAll)
  .post(contact.create)
  .delete(contact.deleteAll);

router.route('/favorite')
  .get(contact.findAllFavorite);

router.route('/:id')
  .get(contact.findOne)
  .put(contact.update)
  .delete(contact.delete);

module.exports = router;
```

Trong đoạn code trên, chúng ta định nghĩa các route quản lý liên hệ được hỗ trợ bởi ứng dụng web. Mỗi route là sự kết hợp của đường dẫn, phương thức HTTP(GET, POST, PUT, DELETE) và đoạn code xử lý. Tiếp theo, chúng ta sẽ đăng ký các route vừa tạo với express app. Hiệu chỉnh tập tin app.js như sau:

```
const express = require('express');
const cors = require('cors');
const contactRouter = require('./app/routers/contact.route');

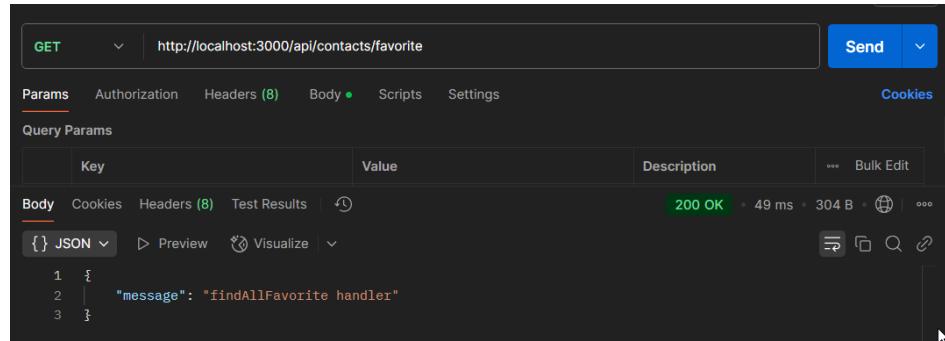
const app = express();

app.use(cors());
app.use(express.json());
app.use("/api/contacts", contactRouter);

app.get('/', (req, res) => {
  res.json({ message: 'Welcome to contact book application.' });
});

module.exports = app;      You, 53 minutes ago • Cai dat express ch
```

Các route quản lý liên hệ sẽ được dùng khi đường dẫn bắt đầu là /api/contacts. Ví dụ để yêu cầu server gửi về các contact được yêu thích, client cần gửi yêu cầu GET đến URL: http://localhost:3000/api/contacts/favorite.



Lưu các thay đổi lên git:

```
> git add -u
> git add app/controllers app/routers
> git commit -m "Dinh nghia cac route cho tai nguyen Contact"
[master aa082fd] Dinh nghia cac route cho tai nguyen Contact
 3 files changed, 52 insertions(+), 2 deletions(-)
  create mode 100644 app/controllers/contact.controller.js
  create mode 100644 app/routers/contact.route.js

< VTOS on Wednesday at 1:50 PM ⌂ ↵ master ⌂ ?2
> { └ D: ⇢ CT449 ⇢ B2207512 ⇢ ContactBook ⇢ contactbook-backend } ↵
```

1.7 Cài đặt xử lý lỗi

Tạo tập tin app/api-error.js

```
class ApiError extends Error {
  constructor(statusCode, message) {
    super();
    this.statusCode = statusCode;
    this.message = message;
  }
}

module.exports = ApiError;
```

Mở tập tin app.js và thêm vào các middleware xử lý lỗi như sau:

```
const express = require('express');
const cors = require('cors');
const contactRouter = require('./app/routers/contact.route');
const ApiError = require('./app/api-error');

const app = express();

app.use(cors());
app.use(express.json());
app.use('/api/contacts', contactRouter);

app.use((req, res, next) => {
    return next(new ApiError(404, 'Resource not found'));
});

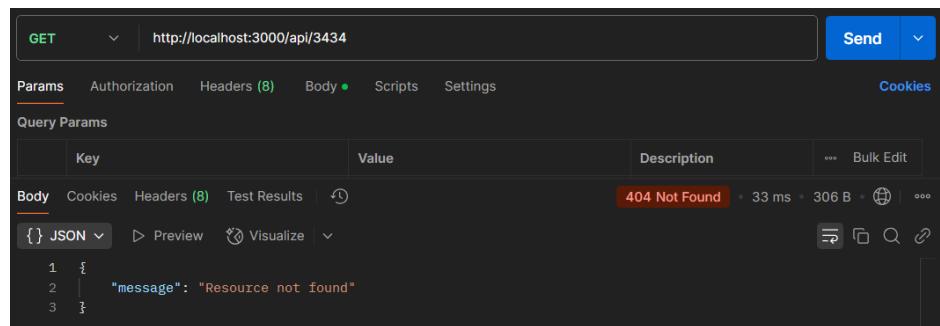
app.use((err, req, res, next) => {
    res.status(err.statusCode || 500).json({ message: err.message || "Internal Server Error" });
});

app.get('/', (req, res) => {
    res.json({ message: 'Welcome to contact book application.' });
});

app.use("/api/contacts", contactRouter);

module.exports = app;
```

Sử dụng một HTTP client và gửi yêu cầu đến một URL không được định nghĩa trong ứng dụng, kiểm tra nhận được lỗi 404 và thông báo "Resource not found".

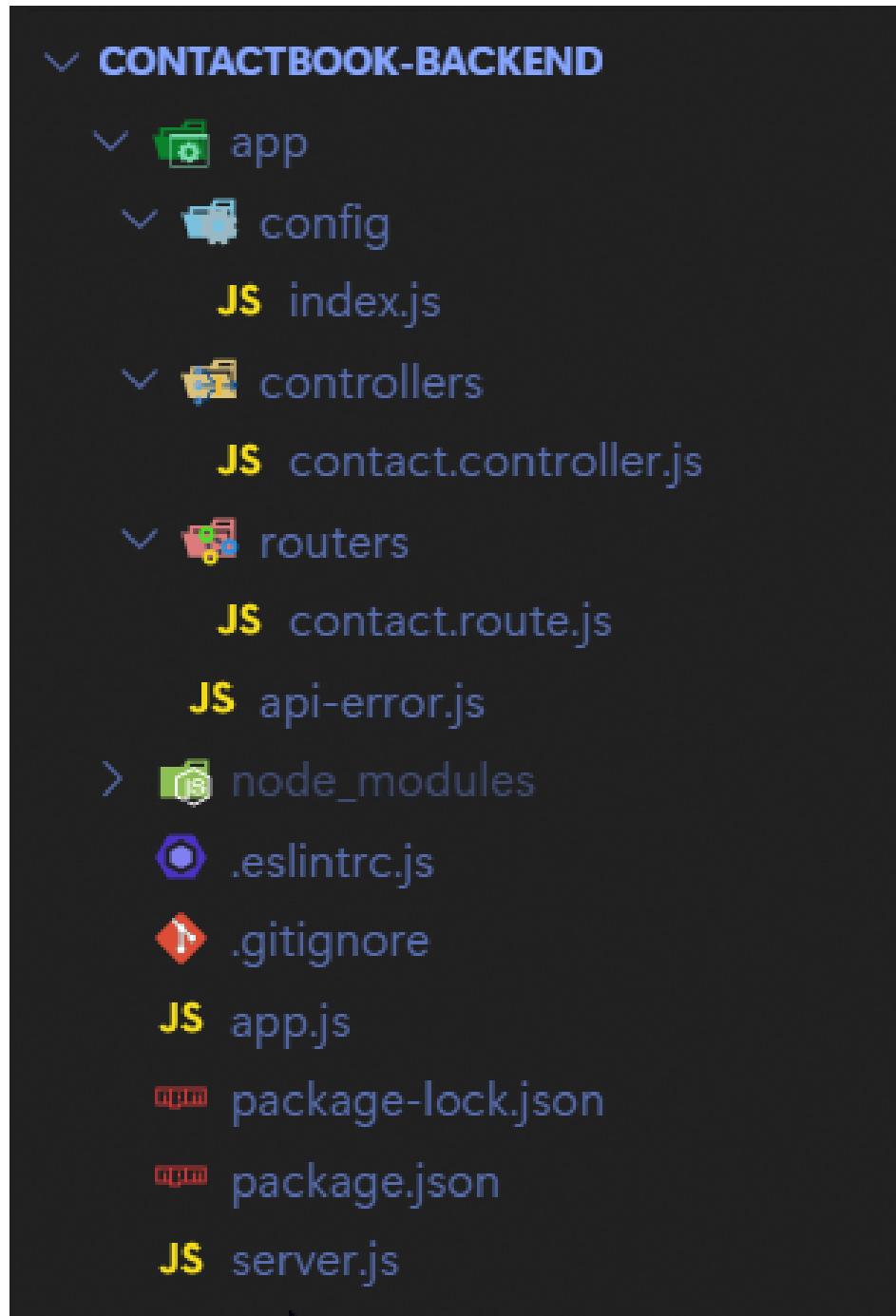


Lưu thay đổi trên git:

```
> git add -u
> git add app/api-error.js
> git commit -m "Cai dat xu ly loi"
[master 344e4ae] Cai dat xu ly loi
 2 files changed, 19 insertions(+)
   create mode 100644 app/api-error.js
> git push origin master
Enumerating objects: 30, done.
Counting objects: 100% (30/30), done.
Delta compression using up to 8 threads
Compressing objects: 100% (23/23), done.
Writing objects: 100% (27/27), 23.08 KiB | 1.00 MiB/s, done.
Total 27 (delta 8), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (8/8), completed with 1 local object.
To github.com:p4wndev/B2207512_Thai_Phu_An_BACKEND_1.git
 38cce1..344e4ae  master -> master

< VTOS on Wednesday at 2:06 PM ⌚ ↵ master # 
> { █ D: ↵ CT449 ↵ B2207512 ↵ ContactBook ↵ contactbook-backend } *
```

Cấu trúc thư mục dự án đến thời điểm này sẽ như sau:



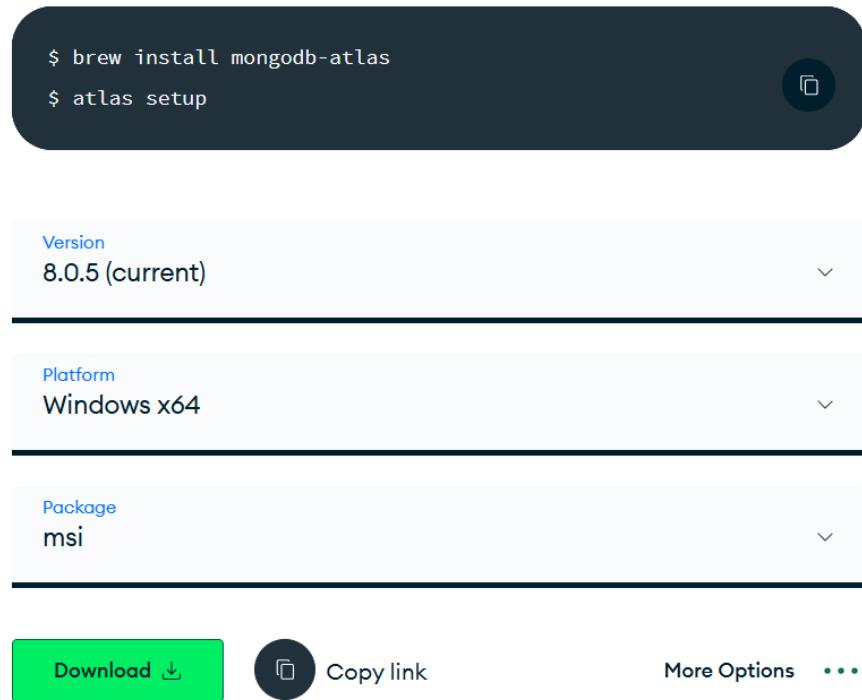
Dường dẫn github của dự án :

https://github.com/p4wndev/B2207512_Thai_Phú_An_BACKEND_1

ContactBook - Backend - Phần 2

1.1 Cài đặt MongoDB

Tải và cài đặt MongoDB Community Server



1.2 Cài đặt thư viện 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 mongodb

```
> npm install mongodb
added 11 packages, and audited 197 packages in 12s
41 packages are looking for funding
  run `npm fund` for details
found 0 vulnerabilities

< VTOS on Wednesday at 10:57 PM  o ✘ master ≠ ~2
> { ■ D: ➔ CT449 ➔ B2207512 ➔ ContactBook ➔ contactbook-backend } ✘
```

Trong tập tin app/config, hiệu chỉnh tập tin index.js:

```
const config = {
  app: {
    port: process.env.PORT || 3000
  },
  db: {
    uri : process.env.MONGODB_URI || 'mongodb://127.0.0.1:27017/contact-book'
  }
};

module.exports = config;
```

Dịnh nghĩa lớp trợ giúp kết nối đến MongoDB: app/utils/mongodb.utils.js

```
const { MongoClient } = require('mongodb');

class MongoDB {
  static connect = async (uri) => {
    if (this.client) return this.client;
    this.client = await MongoClient.connect(uri);
    return this.client;
  }
}

module.exports = MongoDB;
```

Thực hiện kết nối đến CSDL mongoDB khi chạy server, thay toàn bộ nội dung tập tin server.js bằng đoạn mã dưới đây:

```
const app = require("./app");
const config = require("./app/config");
const MongoDB = require("./app/utils/mongodb.util");

async function startServer() {
  try {
    await MongoDB.connect(config.db.uri);
    console.log("Connected to the database!");

    const PORT = config.app.port;
    app.listen(PORT, () => {
      console.log(`Server is running on port ${PORT}`);
    });
  } catch (error) {
    console.log("Cannot connect to the database!", error);
    process.exit();
  }
}

startServer();
```

You, 4 minutes ago • Uncommitted changes

Dịnh nghĩa các lớp dịch vụ ContactServices (trong tập tin app/services/services.js) chứa các API của thư viện MonggoDB để thực hiện thao tác với CSDL MongoDB:

```
const { ObjectId } = require('mongodb');

class ContactService {
  constructor(client) {
    this.Contact = client.db().collection('contacts');
  }
}

module.exports = ContactService;
```

1.3 Cài đặt các handler

Cài đặt các handler create

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

```
exports.create = async (req, res, next) => {
  if (!req.body?.name) {
    return next(new ApiError(400, "Name can not be empty"));
  }

  try {
    const contactService = new ContactService(MongoDB.client);
    const document = await contactService.create(req.body);
    return res.send(document);
  } catch (error) {
    return next(new ApiError(500, "An error occurred while creating the contact"));
  }
};
```

You, 1 second ago • Uncommitted changes

Lời gọi `contactService.create()` lưu thông tin đối tượng contact xuống CSDL. Phương

thức `create()` được định nghĩa trong lớp `ContactService(app/services/contact.services.js)` như sau:

```
class ContactService {
    constructor(client) {
        this.Contact = client.db().collection('contacts');
    }

    extractContactData(payload) {
        const contact = {
            name: payload.name,
            email: payload.email,
            address: payload.address,
            phone: payload.phone,
            favorite: payload.favorite,
        };

        Object.keys(contact).forEach(
            (key) => contact[key] === undefined && delete contact[key]);
        return contact;
    }

    async create(payload) {
        const contact = this.extractContactData(payload);
        const result = await this.Contact.findOneAndUpdate(
            contact,
            { $set: { favorite: contact.favorite === true } },
            { returnDocument: "after", upsert: true }
        );
        return result;
    }
}

module.exports = ContactService;
```

Dùng Postman kiểm tra handler hoạt động đúng:

The screenshot shows the Postman interface with the following details:

- Method:** POST
- URL:** http://localhost:3000/api/contacts
- Body:** Raw JSON (selected)
Content:

```
1 {
2     "name" : "Thai Phu An",
3     "email" : "thaian@gmail.com",
4     "address": "Vinh Long",
5     "phone" : "0123456789",
6     "favorite" : true
7 }
```
- Response:** 200 OK
Time: 43 ms
Size: 409 B

Cài đặt handler `findAll` trong tập tin `app/controllers/contact.controllers.js` như sau:

```
exports.findAll = async (req, res, next) => {
  let documents = [];

  try {
    const contactService = new ContactService(MongoDB.client);
    const { name } = req.query;
    if (name) {
      documents = await contactService.findByName(name);
    } else {
      documents = await contactService.find({});
    }
  } catch (error) {
    return next(new ApiError(500, "An error occurred while retrieving contacts"));
  }

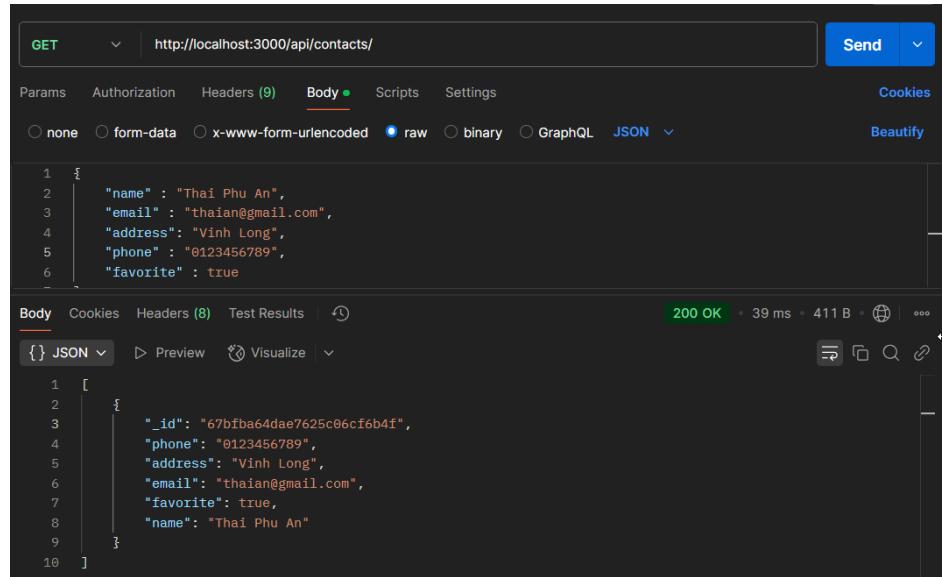
  return res.send(documents);
};
```

`contactService.find(condition)` và `contactService.findByName(name)` lần lượt tìm các tài liệu thỏa điều kiện chỉ định trong đối tượng condition, theo tên name. Hai phương thức này có thể được định nghĩa như sau trong tập tin `app/services/contact.services.js`

```
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" },
  });
}
```

Dùng Postman kiểm tra handler hoạt động đúng:



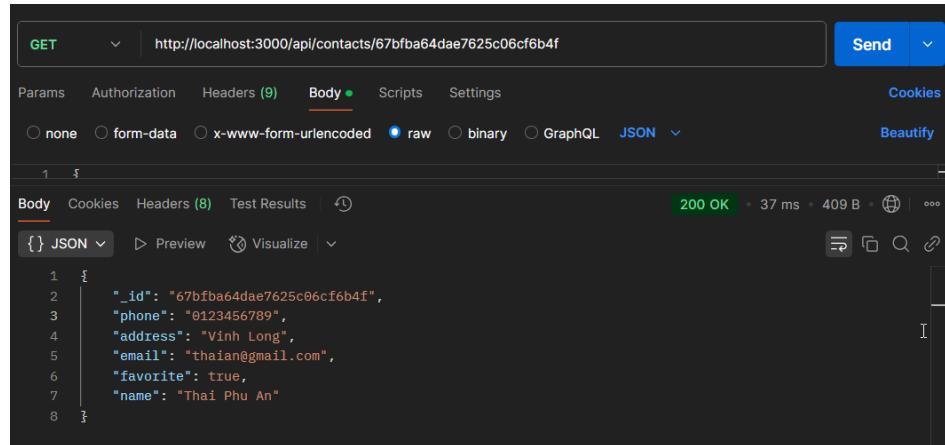
Cài đặt handler `findOne` trong tập tin `app/controllers/contact.controllers.js` như sau:

```
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}`));
  }
};
```

`contactServices.findById(id)` tìm kiếm tài liệu theo Id. Phương thức `findById(id)` có thể được định nghĩa như sau trong tập tin `app/services/contact.service.js`

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

Dùng Postman kiểm tra handler hoạt động đúng.



Cài đặt handler update trong tập tin app/controllers/contact.controllers.js như sau:

```
exports.update = async (req, res, next) => {
  if (Object.keys(req.body).length === 0) {
    return next(new ApiError(400, "Data to update can not be empty"));
  }

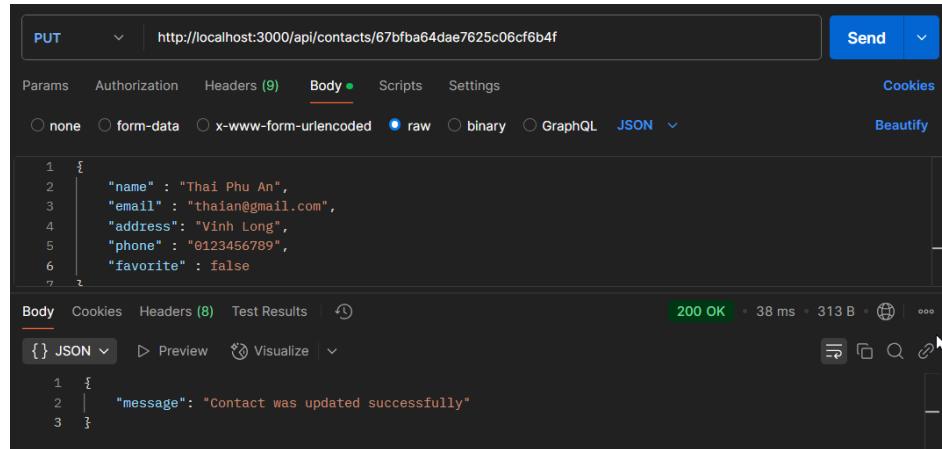
  try {
    const contactService = new ContactService(MongoDB.client);
    const document = await contactService.update(req.params.id, req.body);
    if (!document) {
      return next(new ApiError(404, "Contact not found"));
    }

    return res.send({ message: "Contact was updated successfully" });
  } catch (error) {
    return next(new ApiError(500, `Error updating contact with id=${req.params.id}`));
  }
};
```

contactService.update(id,document) tìm kiếm tài liệu theo id và cập nhật tài liệu này với dữ liệu trong đối tượng document. Phương thức update(id, document) có thể được định nghĩa như sau trong tập tin app/services/contact.services.js.

```
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
}
```

Dùng Postman kiểm tra handler hoạt động đúng:



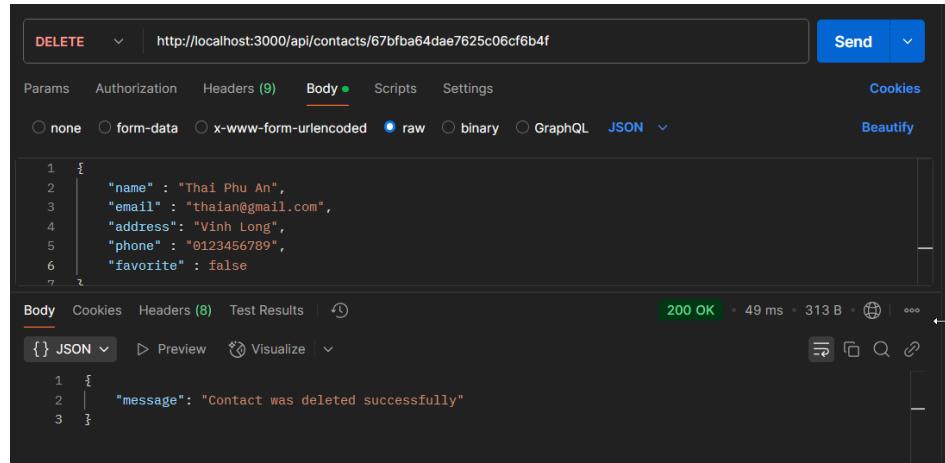
Cài đặt handler delete trong tập tin app/controllers/contact.controllers.js như sau:

```
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({ message: "Contact was deleted successfully" });  
  } catch (error) {  
    return next(new ApiError(500, `Could not delete contact with id=${req.params.id}`));  
  }  
};
```

contactService.update(id) tìm kiếm tài liệu theo id và xóa tài liệu này. Phương thức delete(id) có thể được định nghĩa như sau trong tập tin app/services/contact.services.js.

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

Dùng Postman kiểm tra handler hoạt động đúng:



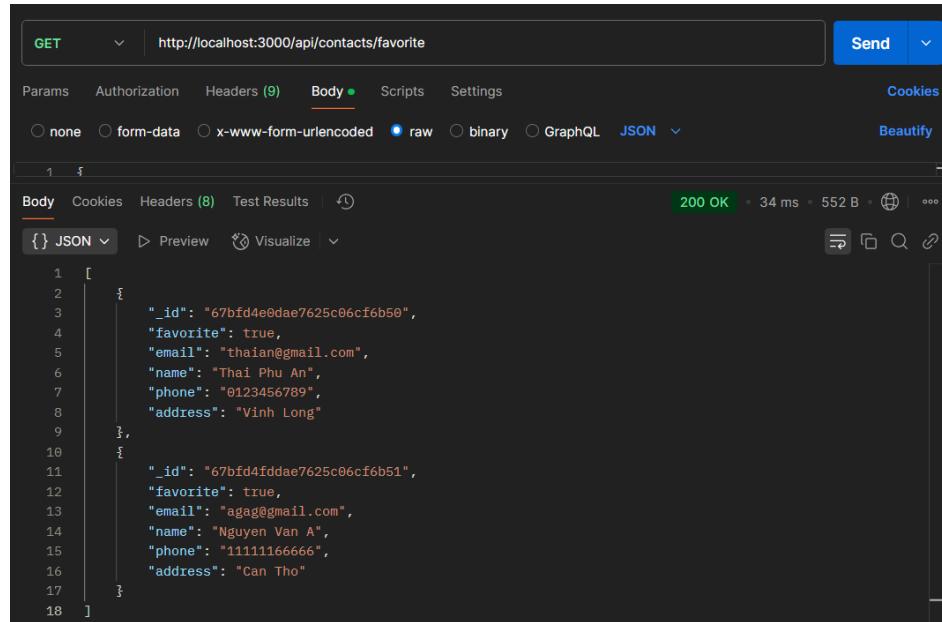
Cài đặt handler `findAllFavorite` trong tập tin `app/controllers/contact.controllers.js` như sau:

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

Phương thức `findFavorite()` trong lớp `ContactServices` có thể được định nghĩa như sau trong tập tin `app/services/contact.services.js`.

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

Dùng Postman kiểm tra handler hoạt động đúng.



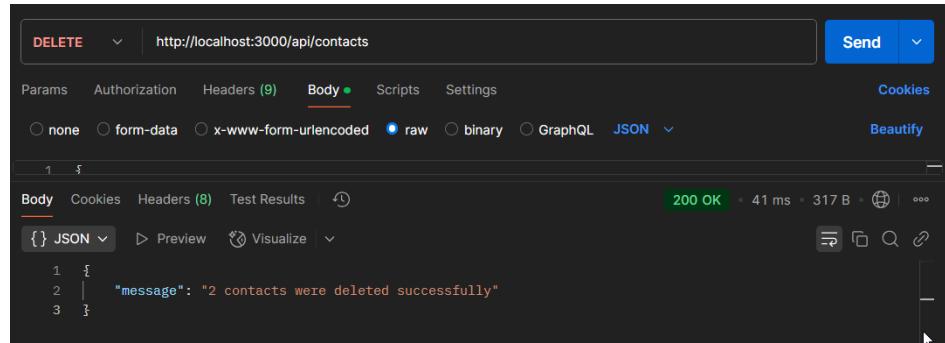
Cài đặt handler deleteALL

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

contactServices.deleteMany() xóa tất cả các đối tượng trong collection. Phương thức deleteAll() có thể định nghĩa như sau:

```
async deleteAll() {
  const result = await this.Contact.deleteMany({});
  return result.deletedCount;
}
```

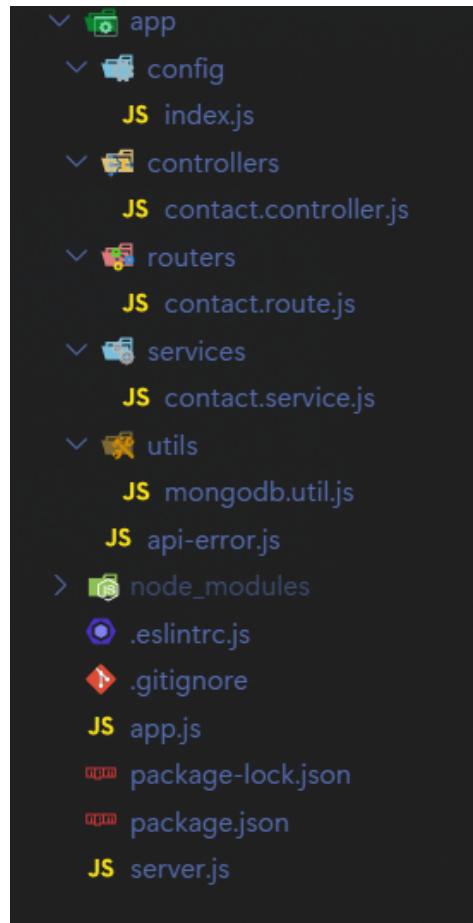
Dùng Postman hoặc curl kiểm tra handler hoạt động đúng.



Dưới đây là các thư mục đã được push lên GitHub:

```
Enumerating objects: 35, done.
Counting objects: 100% (35/35), done.
Delta compression using up to 8 threads
Compressing objects: 100% (31/31), done.
Writing objects: 100% (35/35), 4.50 MiB | 1.84 MiB/s, done.
Total 35 (delta 10), reused 0 (delta 0), pack-reused 0 (from 0)
remote: Resolving deltas: 100% (10/10), done.
To github.com:p4wndev/B2207512_Thai_Phu_An_BACKEND_2.git
 * [new branch]      master → master
```

Cấu trúc thư mục dự án đến thời điểm này sẽ như sau:



Dường dẫn github của dự án :

https://github.com/p4wndev/B2207512_Thai_Phu_An_BACKEND_2