

**TRƯỜNG ĐẠI HỌC CẦN THƠ**  
**CÔNG NGHỆ THÔNG TIN & TRUYỀN THÔNG**



**CT449 PHÁT TRIỂN ỨNG DỤNG WEB**

**Lab 1 + 2**

**Ứng dụng Contactbook - Backend**

Sinh viên thực hiện  
Huỳnh Nhật Hào  
MSSV: B1910062

Giảng viên hướng dẫn  
Ths. Nguyễn Minh Trung

Khóa: 45

# Lab 1:

# Cài đặt Node và git

```
ADMIN@DESKTOP-7UQDS3G MINGW64 /d/Hoc Tap/Phat trien ung dung web/repeat for labs
$ node -v
v16.17.1

ADMIN@DESKTOP-7UQDS3G MINGW64 /d/Hoc Tap/Phat trien ung dung web/repeat for labs
$ git --version
git version 2.35.1.windows.2
```

Đường dẫn đến dự án git: [https://github.com/mrhaomp2001/CT449\\_B1910062\\_Backend.git](https://github.com/mrhaomp2001/CT449_B1910062_Backend.git)

## 1. Tạo ứng dụng Node

## Tạo dự án Node bằng lệnh “*npm init*”

```
package name: (repeat-for-labs)
version: (1.0.0)
description:
entry point: (index.js) server.js
test command:
git repository:
keywords:
author: Hao B1910062
license: (ISC)
About to write to D:\Hoc Tap\Phat trien ung dung web\repeat for labs\package.json:

{
  "name": "repeat-for-labs",
  "version": "1.0.0",
  "description": "",
  "main": "server.js",
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "B1910062\u001b[D\u001b[D\u001b[D\u001b[D\u001b[D\u001b[D\u001b[D\u001b[DHao B1910062\u001b[2~\u001b[D2\u001b[D\u001b[D2~2",
  "license": "ISC"
}

Is this OK? (yes)
```

## 2. Quản lý mã nguồn dự án với git và GitHub

Dự án đã được quản lý bằng git.

```
PS D:\Hoc Tap\Phat trien ung dung web\contactbook-backend> git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
        modified:   app/controllers/contact.controller.js
        modified:   package-lock.json
        modified:   package.json

no changes added to commit (use "git add" and/or "git commit -a")
```

Đường dẫn đến dự án git: [https://github.com/mrhaomp2001/CT449\\_B1910062\\_Backend.git](https://github.com/mrhaomp2001/CT449_B1910062_Backend.git)

	mrhaomp2001 Cài đặt các handler truy xuất CSDL	8ba5ab5 2 weeks ago	🔒 5 commits
	app	Cài đặt các handler truy xuất CSDL	2 weeks ago
	.gitignore	Cài đặt express chạy thông báo xin chào	2 months ago
	app.js	Cài đặt các handler truy xuất CSDL	2 weeks ago
	package-lock.json	Cài đặt các handler truy xuất CSDL	2 weeks ago
	package.json	Cài đặt các handler truy xuất CSDL	2 weeks ago
	server.js	Cài đặt các handler truy xuất CSDL	2 weeks ago

## 3. Cài đặt Express

- tập tin app.js:

```
app.js > app.get("/") callback
1  const express = require("express");
2  const cors = require("cors");
3  const contactsRouter = require("./app/routes/contact.route");
4  const ApiError = require("./app/api-error");
5
6  const app = express();
7
8  app.use(cors());
9  app.use(express.json());
10
11 app.get("/", (req, res) => {
12   return res.status(200).json({message: "welcome"});
13 });
14
```

- tập tin server.js:

```
server.js > startServer
1  const app = require("../app");
2  const config = require("../app/config");
3  const MongoDB = require("../app/utils/mongodb.util");
4
5  //start server
6  async function startServer() {
7    try {
8      await MongoDB.connect(config.db.uri);
9      console.log("Connect to database");
10
11      const PORT = config.app.port;
12      app.listen(PORT, () => {
13        console.log("Server is running on port " + PORT);
14      });
15    } catch (error) {
16      console.log("Can't connect to database: " + error);
17      process.exit();
18    }
19  }
20
21  startServer();
```

- Tập tin app/config/index.js:

```
index.js
app > config > index.js > ...
1  const config = {
2    app: {
3      port: process.env.PORT || 3000,
4    },
5    db: {
6      uri: process.env.MONGODB_URI || "mongodb://127.0.0.1:27017/contactbook",
7    },
8  };
9
10 module.exports = config;
```

## 4. Định nghĩa Controller và các route

- Tập tin app/controllers/contact.controller.js

Đã cài đặt tất cả các handle:

```
app > controllers >  contact.controller.js > ...
1  const ContactService = require("../services/contact.service");
2  const MongoDB = require("../utils/mongodb.util");
3  const ApiError = require("../api-error");
4  const { client } = require("../utils/mongodb.util");
5
6  exports.create = async (req, res, next) => {
7    if (!req.body?.name) {
8      return next(new ApiError(400, "Name cannot be empty"));
9    }
10
11    try {
12      const contactService = new ContactService(MongoDB.client);
13      const document = await contactService.create(req.body);
14      return res.send(document);
15    } catch (error) {
16      return next(new ApiError(500, "An error occurred " + req.body.email));
17    }
18  };
19
20  exports.findAll = async (req, res, next) => {
21    let documents = [];
22    try {
23      const contactService = new ContactService(MongoDB.client);
24      const { name } = req.query;
25      if (name) {
26        documents = await contactService.findByName(name);
27      } else {
28        documents = await contactService.find({});
29      }
30    } catch (error) {
31      return next(new ApiError(500, "An error occurred "));
```

- tập tin app/routes/contact.route.js:

```
app > routes > ms contact.route.js > ...
1  const express = require("express");
2  const contacts = require("../controllers/contact.controller");
3
4  const router = express.Router();
5
6  router.route("/").get(contacts.findAll).post(contacts.create).delete(contacts.deleteAll);
7
8  router.route("/favorites").get(contacts.findAllFavorites);
9
10 router.route("/:id").get(contacts.findOne).put(contacts.update).delete(contacts.delete);
11
12 module.exports = router;
```

- Các route hoạt động tốt khi sử dụng Postman, các ảnh chụp minh họa gọi api bằng Postman được dán ở lab 2.

## 5. Cài đặt xử lý lỗi

- tập tin app/api-error.js:

```
app > ms api-error.js > ...
1  class ApiError extends Error {
2    constructor(statusCode, message) {
3      super();
4      this.statusCode = statusCode;
5      this.message = message;
6    }
7  }
8  module.exports = ApiError;
```

Thêm xử lý lỗi vào app.js:

```
app.js > app.get("/") callback
1  const express = require("express");
2  const cors = require("cors");
3  const contactsRouter = require("./app/routes/contact.route");
4  const ApiError = require("./app/api-error");
5
6  const app = express();
7
8  app.use(cors());
9  app.use(express.json());
10
11 app.get("/", (req, res) => {
12   return res.status(200).json({message: "welcome"});
13 });
14
15 app.use("/api/contacts", contactsRouter);
16
17 app.use((req, res, next) => {
18   //chạy khi không có route nào
19   return next(new ApiError(404, "Resource Not Found"));
20 });
21
22 app.use((err, req, res, next) => {
23   return res.status(err.statusCode || 500).json({
24     message: err.message || "Internal Server Error",
25   });
26 });
27
28 module.exports = app;
```

# Lab 2

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

- tệp: app/config/index.js:

```
app > config > index.js > ...
1  const config = {
2    app: {
3      port: process.env.PORT || 3000,
4    },
5    db: {
6      uri: process.env.MONGODB_URI || "mongodb://127.0.0.1:27017/contactbook",
7    },
8  };
9
10 module.exports = config;
```

- tệp: app/utils/mongodb.util.js:

```
app > utils > mongodb.util.js > ...
1  const { MongoClient } = require("mongodb");
2
3  class MongoDB {
4    static connect = async (uri) => {
5      if (this.client) return this.client;
6      this.client = await MongoClient.connect(uri);
7
8      return this.client;
9    };
10 }
11
12 module.exports = MongoDB;
```



- tệp server.js:

```
server.js > startServer
1  const app = require("../app");
2  const config = require("../app/config");
3  const MongoDB = require("../app/utils/mongodb.util");
4
5  //start server
6  async function startServer() {
7    try {
8      await MongoDB.connect(config.db.uri);
9      console.log("Connect to database");
10
11      const PORT = config.app.port;
12      app.listen(PORT, () => {
13        console.log("Server is running on port " + PORT);
14      });
15    } catch (error) {
16      console.log("Can't connect to database: " + error);
17      process.exit();
18    }
19  }
20
21  startServer();
```

- tệp app/services/contact.service.js:

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

## 2. Cài đặt các handler

### 2.1 Cài đặt handler create:

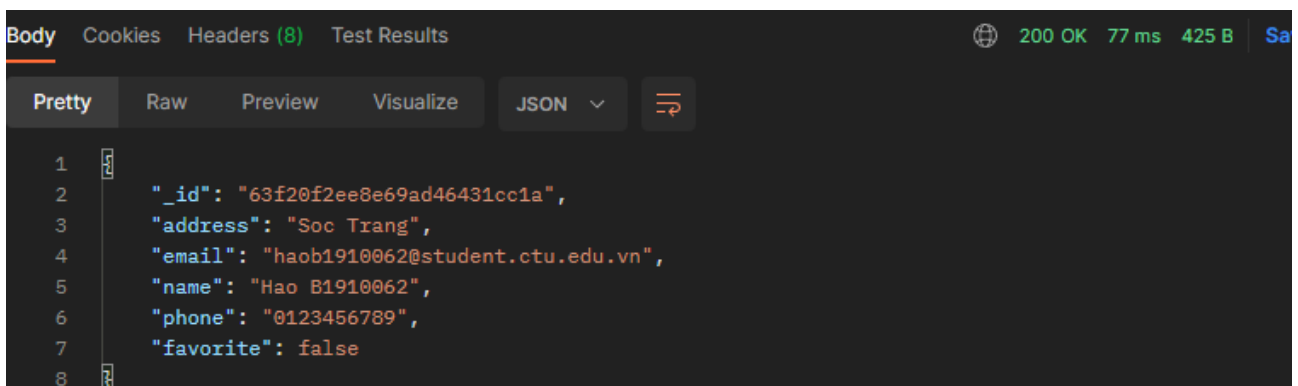
- tệp: app/controllers/contact.controller.js:

```
6 exports.create = async (req, res, next) => {
7   if (!req.body?.name) {
8     return next(new ApiError(400, "Name cannot be empty"));
9   }
10
11   try {
12     const contactService = new ContactService(MongoDB.client);
13     const document = await contactService.create(req.body);
14     return res.send(document);
15   } catch (error) {
16     return next(new ApiError(500, "An error occurred " + req.body.email));
17   }
18 };
19
```

- tệp: app/services/contact.service.js:

```
6 // Định nghĩa các phương thức truy xuất CSDL sử dụng mongodb API
7 extractContactData(payload) {
8   const contact = {
9     name: payload.name,
10    email: payload.email,
11    address: payload.address,
12    phone: payload.phone,
13    favorite: payload.favorite,
14  };
15  // Remove undefined fields
16  Object.keys(contact).forEach((key) => contact[key] === undefined && delete contact[key]);
17  return contact;
18 }
19
20 async create(payload) {
21   const contact = this.extractContactData(payload);
22   const result = await this.Contact.findOneAndUpdate(
23     contact,
24     {
25       $set:
26       { favorite: contact.favorite === true }
27     },
28     { returnDocument: "after", upsert: true },
29   );
30   return result.value;
31 }
32
```

Gọi api bằng Postman, nhận được phản hồi 200, thành công.



## 2.2 handler findAll:

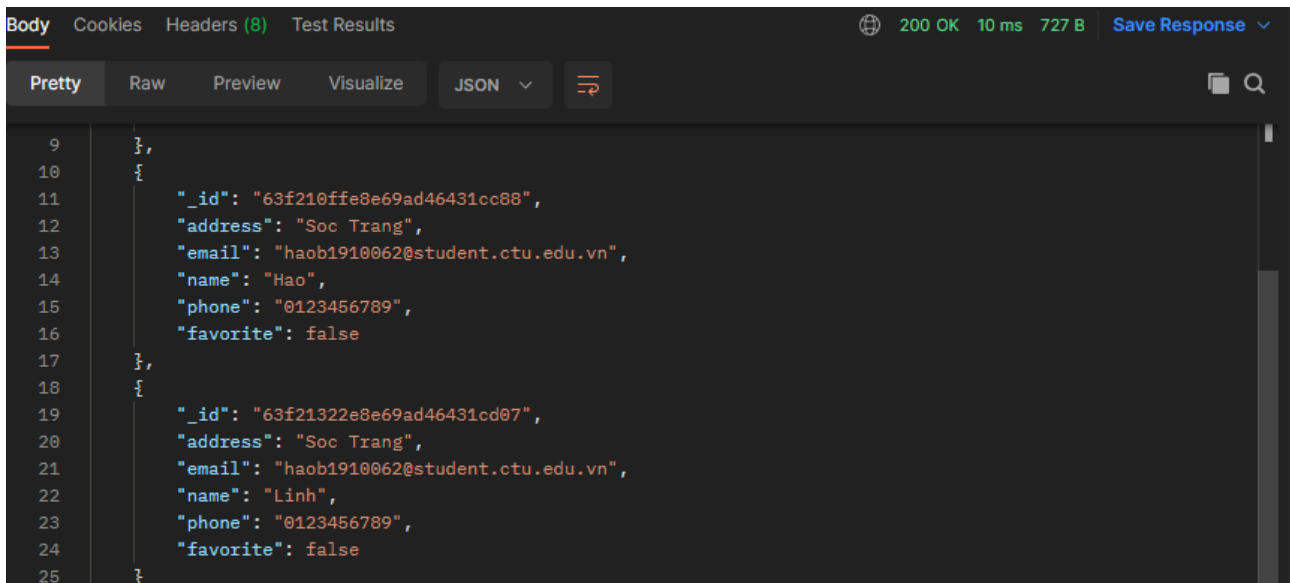
- tệp: app/controllers/contact.controller.js:

```
20 exports.findAll = async (req, res, next) => {
21   let documents = [];
22   try {
23     const contactService = new ContactService(MongoDB.client);
24     const { name } = req.query;
25     if (name) {
26       documents = await contactService.findByName(name);
27     } else {
28       documents = await contactService.find({});
29     }
30   } catch (error) {
31     return next(new ApiError(500, "An error occurred "));
32   }
33   return res.send(documents);
34 };
```

- tệp: app/services/contact.service.js:

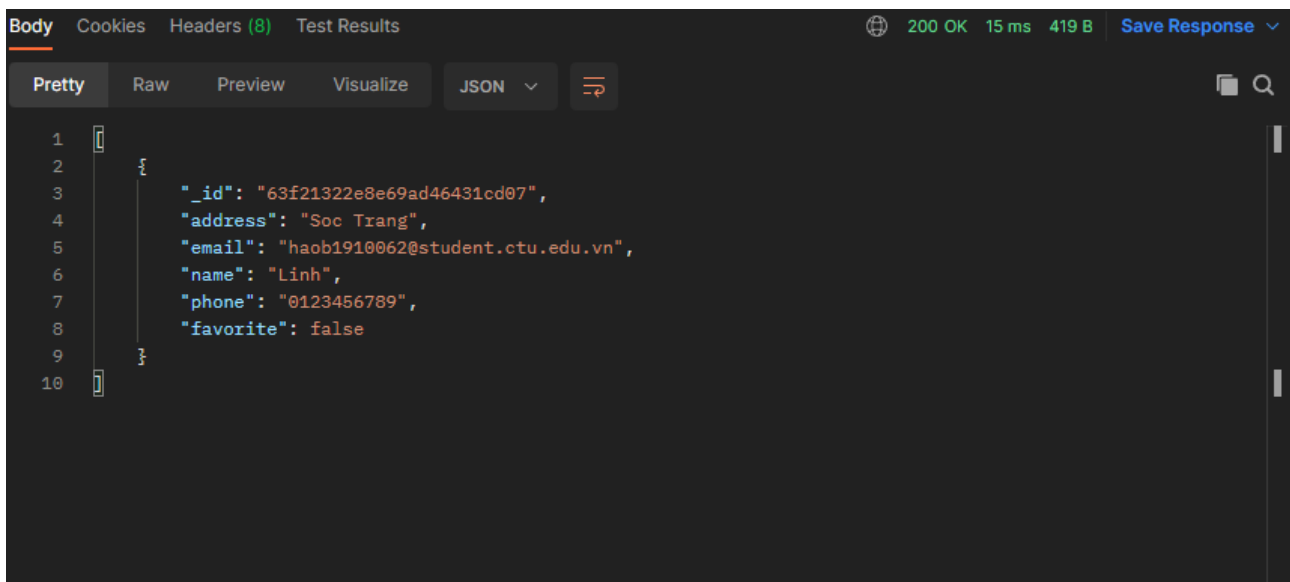
```
33 async find(filter) {
34   const cursor = await this.Contact.find(filter);
35   return await cursor.toArray();
36 }
37
38 async findByName(name) {
39   return await this.find({
40     name: { $regex: new RegExp(name), $options: "i" },
41   });
42 }
```

Gọi api khi không truyền tham số “name” bằng Postman, nhận được phản hồi 200, thành công.



```
9      },
10     {
11       "_id": "63f210ffe8e69ad46431cc88",
12       "address": "Soc Trang",
13       "email": "haob1910062@student.ctu.edu.vn",
14       "name": "Hao",
15       "phone": "0123456789",
16       "favorite": false
17     },
18     {
19       "_id": "63f21322e8e69ad46431cd07",
20       "address": "Soc Trang",
21       "email": "haob1910062@student.ctu.edu.vn",
22       "name": "Linh",
23       "phone": "0123456789",
24       "favorite": false
25     }
  }
```

Gọi api khi truyền tham số “name=linh”, nhận được phản hồi 200, thành công lấy ra nội dung liên hệ có tên “linh”.



```
1  [
2    {
3      "_id": "63f21322e8e69ad46431cd07",
4      "address": "Soc Trang",
5      "email": "haob1910062@student.ctu.edu.vn",
6      "name": "Linh",
7      "phone": "0123456789",
8      "favorite": false
9    }
10 ]
```

## 2.3 handler findOne:

Tìm một liên hệ theo id:

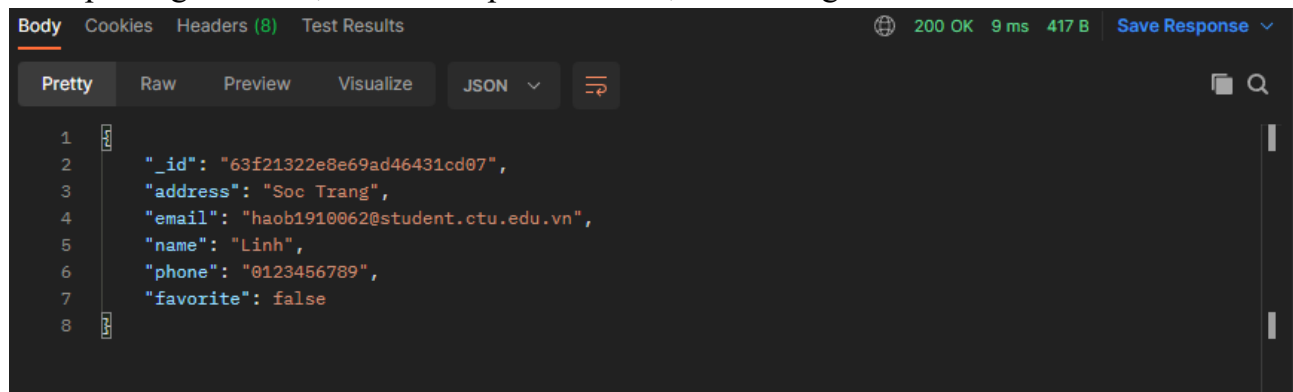
- tệp: app/controllers/contact.controller.js:

```
36 exports.findOne = async (req, res, next) => {
37   try {
38     const contactService = new ContactService(MongoDB.client);
39     const document = await contactService.findById(req.params.id);
40     if (!document) {
41       return next(new ApiError(404, "Contact not found "));
42     }
43     return res.send(document);
44   } catch (error) {
45     return next(new ApiError(500, `Error retrieving contact id: ${req.params.id}`));
46   }
47   // res.send({ message: "findOne" });
48 };
49
50
```

- tệp: app/services/contact.service.js:

```
44 async findById(id) {
45   return await this.Contact.findOne({
46     _id: ObjectId.isValid(id) ? new ObjectId(id) : null,
47   });
48 }
49
```

Gọi api bằng Postman, nhận được phản hồi 200, thành công.



## 2.4 handler update:

Cập nhật một liên hệ

- tệp: app/controllers/contact.controller.js:

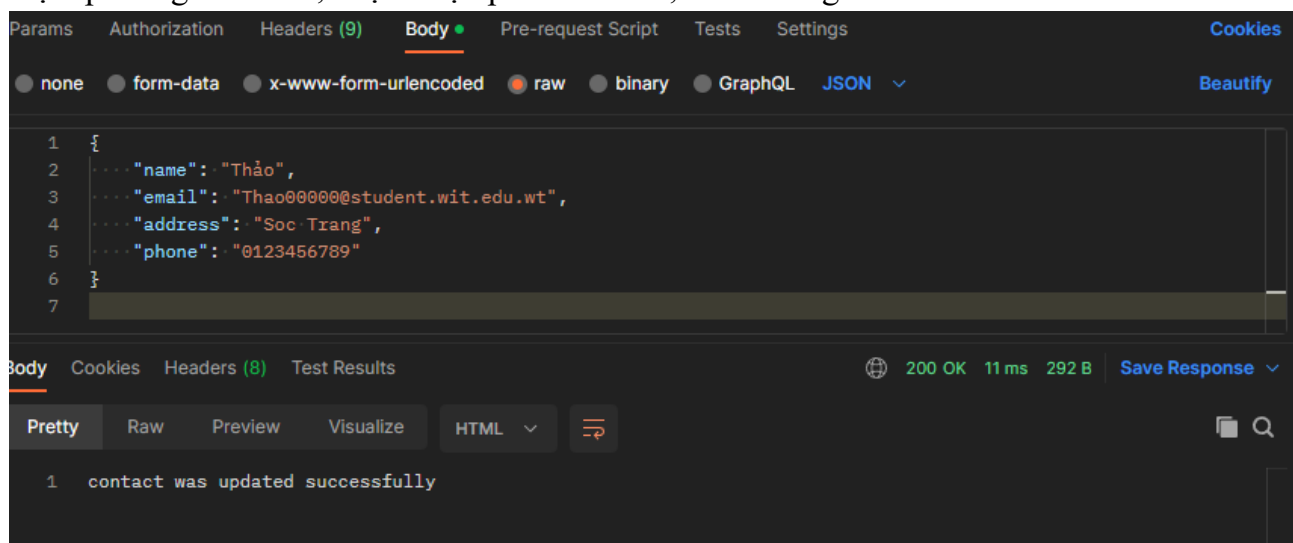
```
51 exports.update = async (req, res, next) => {
52   if (Object.keys(req.body).length === 0) {
53     return next(new ApiError(404, "Data to update can not be empty"));
54   }
55
56   try {
57     const contactService = new ContactService(MongoDB.client);
58     const document = await contactService.update(req.params.id, req.body);
59     return res.send("contact was updated successfully");
60   } catch (error) {
61     return next(new ApiError(500, `Error updating contact id: ${req.params.id}`));
62   }
63   // res.send({ message: "update" });
64 };
65
```

- tệp: app/services/contact.service.js:

```
50 async update(id, payload) {
51   const filter = {
52     _id: ObjectId.isValid(id) ? new ObjectId(id) : null,
53   };
54   const update = this.extractContactData(payload);
55   const result = await this.Contact.findOneAndUpdate(filter, { $set: update }, { returnDocument: "after" });
56   return result.value;
57 }
58
```

Sửa tên “linh” thành “Thảo”, sửa lại “email”.

Gọi api bằng Postman, nhận được phản hồi 200, thành công.



## 2.5 handler delete:

Xóa một liên hệ.

- tệp: app/controllers/contact.controller.js:

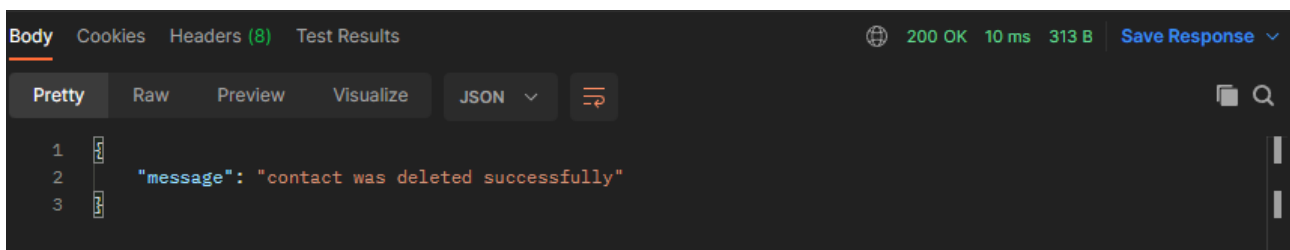
```
66 exports.delete = async (req, res, next) => {
67   try {
68     const contactService = new ContactService(MongoDB.client);
69     const document = await contactService.delete(req.params.id);
70     if (!document) {
71       return next(new ApiError(404, "Contact not found "));
72     }
73     res.send({ message: "contact was deleted successfully" });
74   } catch (error) {
75     return next(new ApiError(500, `Error delete contact id: ${req.params.id}`));
76   }
77   // res.send({ message: "delete" });
78 };
```

- tệp: app/services/contact.service.js:

```
59 async delete(id) {
60   const result = await this.Contact.findOneAndDelete({
61     _id: ObjectId.isValid(id) ? new ObjectId(id) : null,
62   });
63   return result.value;
64 }
```

Xóa liên hệ có tên là Hao với id là 63f21322e8e69ad46431cd07

Gọi api bằng Postman, nhận được phản hồi 200, thành công.



## 2.6 handler findAllFavorite:

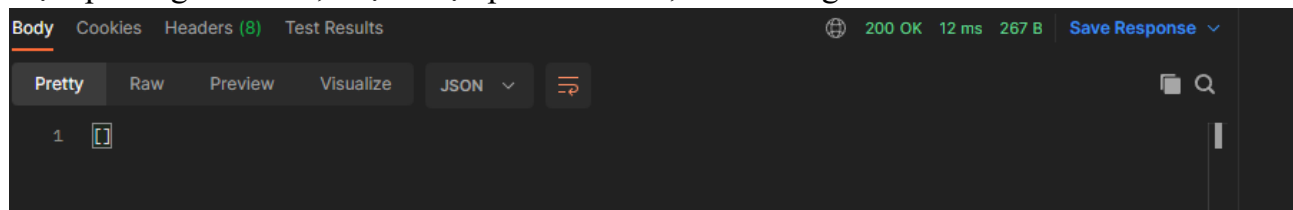
- tệp: app/controllers/contact.controller.js:

```
94 exports.findAllFavorites = async (req, res, next) => {
95   try {
96     const contactService = new ContactService(MongoDB.client);
97     const documents = await contactService.findFavorite();
98     return res.send(documents);
99   } catch (error) {
100     return next(new ApiError(500, `Error find all favorites.`));
101   }
102   // res.send({ message: "findAllFavorites" });
103 };
```

- tệp: app/services/contact.service.js:

```
71 async findFavorite() {
72   return await this.find({ favorite: true });
73 }
74 }
```

Gọi api bằng Postman, nhận được phản hồi 200, thành công.





## 2.7 handler deleteAll:

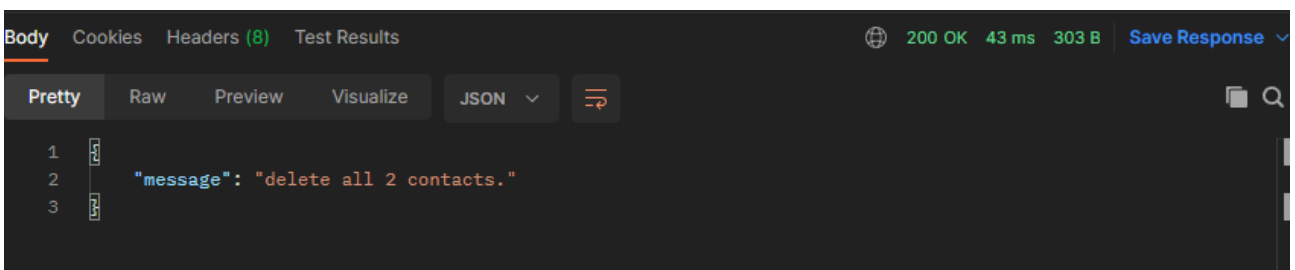
- tệp: app/controllers/contact.controller.js:

```
81 exports.deleteAll = async (req, res, next) => {
82   try {
83     const contactService = new ContactService(MongoDB.client);
84     const deleteCount = await contactService.deleteAll();
85     return res.send({
86       message: `delete all ${deleteCount} contacts.`,
87     });
88   } catch (error) {
89     return next(new ApiError(500, `Error delete all contacts.`));
90   }
91   // res.send({ message: "deleteAll" });
92 };
93
```

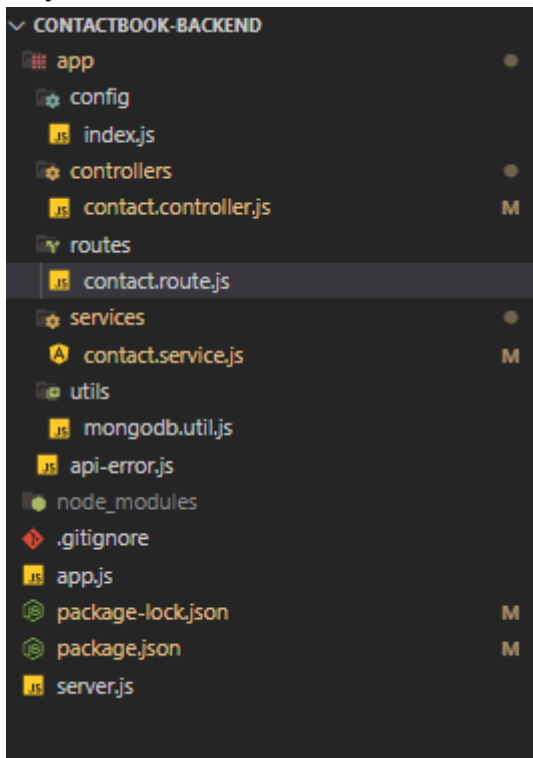
- tệp: app/services/contact.service.js:

```
66 async deleteAll() {
67   const result = await this.Contact.deleteMany({});
68   return result.deletedCount;
69 }
70
```

Gọi api bằng Postman, nhận được phản hồi 200, thành công.



Cây thư mục dự án hiện tại:



Đường dẫn đến dự án git: [https://github.com/mrhaomp2001/CT449\\_B1910062\\_Backend.git](https://github.com/mrhaomp2001/CT449_B1910062_Backend.git)

## Phần làm thêm

Thêm phần ghi chú cho các liên hệ:

Sửa lại hàm `extractContactData()` trong tệp: `app/services/contact.service.js`, ta thêm trường `note`:

```
7   extractContactData(payload) {
8     const contact = {
9       name: payload.name,
10      email: payload.email,
11      address: payload.address,
12      phone: payload.phone,
13      favorite: payload.favorite,
14      note: payload.note,
15    };
16    // Remove undefined fields
17    Object.keys(contact).forEach((key) => contact[key] === undefined && delete contact[key]);
18    return contact;
19  }
```

Khi gọi api, chúng ta cần thêm trường `note` trong body:

```
1  {
2    "name": "Thảo",
3    "email": "Thao00000@student.wit.edu.wt",
4    "address": "Soc Trang",
5    "phone": "0123456789",
6    "note": "Đây là số của Thảo"
7  }
```

Ở hàm `create`, nếu ta không cần `note`, ta có thể để trống, nếu để trống, `note` mặc định bằng “`none`”:

```
async create(payload) {
  const contact = this.extractContactData(payload);
  const note = payload.note || 'none';
  const result = await this.Contact.findOneAndUpdate(
    contact,
    {
      $set: {
        favorite: contact.favorite === false,
        note,
      },
    },
    { returnDocument: "after", upsert: true },
  );
  return result.value;
}
```

Kiểm tra dữ liệu bằng api findAll:

```
{
  "_id": "63f43819e8e69ad46431d19d",
  "address": "Soc Trang",
  "email": "Thao00000@student.wit.edu.wt",
  "name": "Thảo",
  "note": "Đây là số của Thảo",
  "phone": "0123456789",
  "favorite": false
}
```

Ta thấy, liên hệ đã có ghi chú. Các api và phương thức khác hoạt động tương tự.