

LAB 3

MỤC TIÊU

Kết thúc bài thực hành sinh viên có khả năng:

- ✓ Connect dự án với Database MongoDB
- ✓ Tạo model bằng Mongoose.
- ✓ Tạo API (GET,POST,PUT)

NỘI DUNG : TẠO DABASSE VÀ API QUẢN LÝ THỰC PHẨM TRÁI CÂY

BÀI 1: TẠO DATABASE VÀ CONNECT

*Tạo database bằng Atlas (MongoDB Cloud)

Bước 1: Truy cập <https://www.mongodb.com/cloud/atlas/register>

Bước 2: Đăng nhập bằng tài khoản Google

MongoDB Atlas

- ✓ **Work with your data as code**
Documents in MongoDB map directly to objects in your programming language. Modify your schema as your apps grow over time.
- ✓ **Focus on building, not managing**
Let MongoDB Atlas take care of the infrastructure operations you need for performance at scale, from always-on security to point-in-time recovery.
- ✓ **Simplify your data dependencies**
Leverage application data for full-text search, real-time analytics, rich visualizations and more with a single API and minimal data movement.

Sign up

See what Atlas is capable of for free.

First Name*

Last Name*

Company

Email*

Password*

☐ I agree to the [Terms of Service](#) and [Privacy Policy](#).

Create your Atlas account

or

Sign up with Google

[Sign in](#)

Sau khi đăng nhập thành công và thực hiện các bước yêu cầu của **Atlas** chúng ta sẽ có một màn hình như bên dưới.

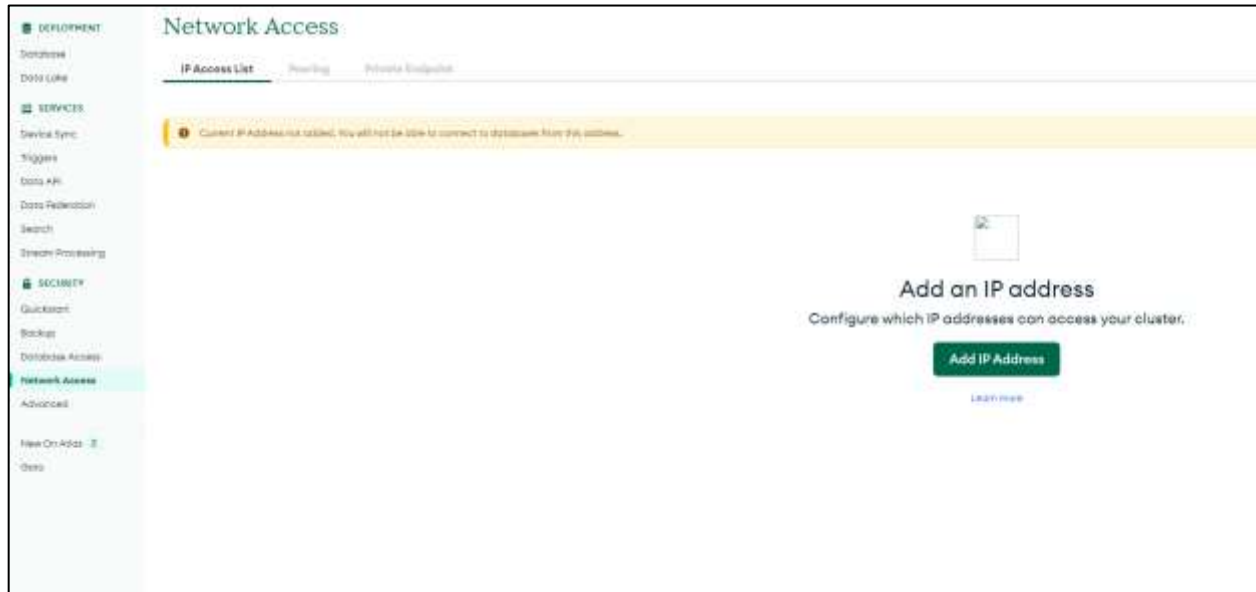
The screenshot shows the MongoDB Atlas web interface. The top navigation bar includes the Atlas logo, a dropdown menu for 'Phan's Org - ...', and links for 'Access Manager' and 'Billing'. Below this, a secondary navigation bar shows 'Project 0' and tabs for 'Data Services' (selected), 'App Services', and 'Charts'. The left sidebar contains a menu with sections: 'Overview', 'DEPLOYMENT' (with 'Database' selected), 'SERVICES' (with 'Device Sync' selected), and 'SECURITY'. The main content area is titled 'Database Deployments' and includes a search bar. A yellow warning banner at the top states: 'Current IP Address not added. You will not be able to connect to databases from this address.' Below this, a green button with a download icon is labeled 'Load sample datasets to Cluster0.' with a description: 'Atlas provides sample data you can load into your Atlas clusters. You can use this data to quickly...'. A section for 'Cluster0' has buttons for 'Connect', 'View Monitoring', 'Browse Collections', and a menu icon. Below that, a 'Visualize Your Data' section encourages building dashboards and charts, with a 'Dismiss' button and an 'Explore Charts' button. At the bottom, a table lists deployment details:

| VERSION | REGION | CLUSTER TIER | TYPE | BACKUPS |
|---------|-------------------------------|----------------------|-----------------------|----------|
| 6.0.8 | AWS / N. Virginia (us-east-1) | M0 Sandbox (General) | Replica Set - 3 nodes | Inactive |

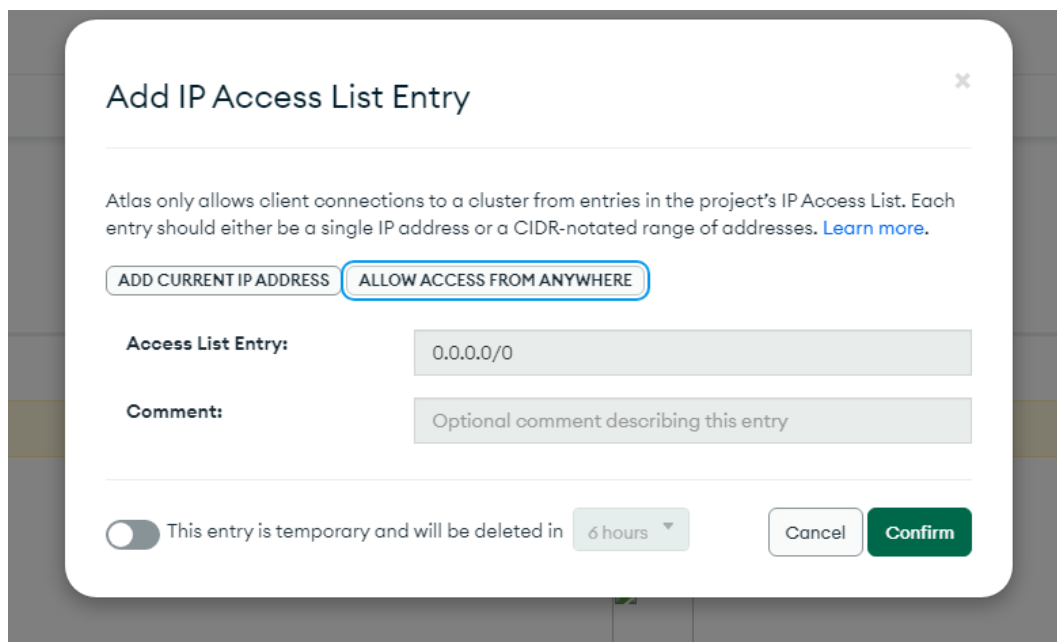
A '+ Add Tag' button is located below the table.

Bước 2: Truy cập vào **Network Access**

Chọn **Add IP Address**

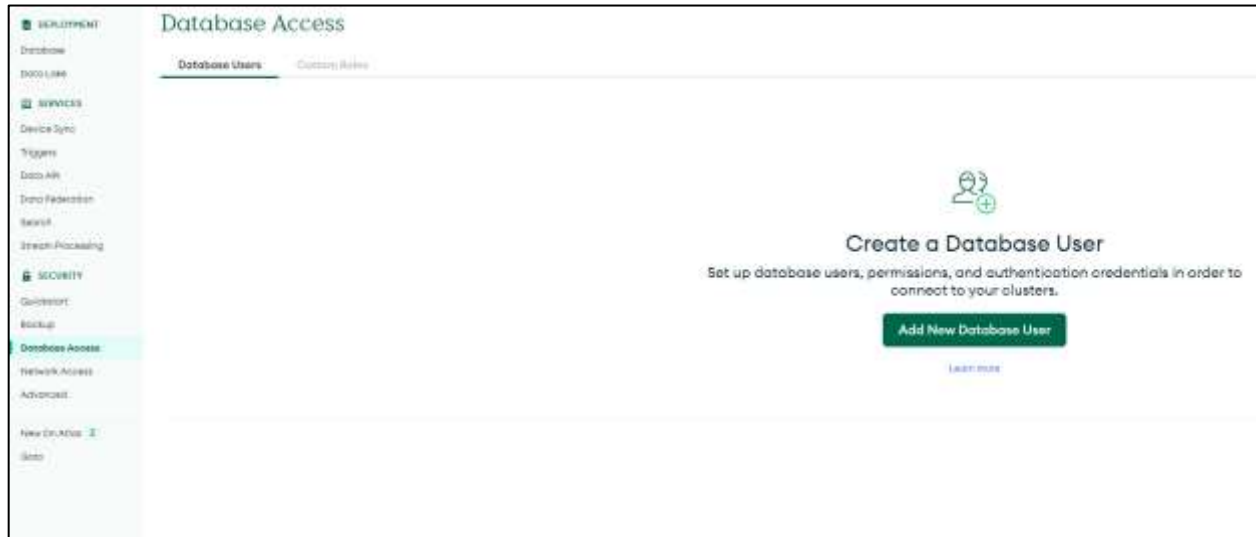


Bước 3: Sau đó nhấn vào **ALLOW ACCESS FROM ANYWHERE**, Atlas sẽ tự động tạo ra **Access List Entry** như hình dưới. Sau đó nhấn **Confirm**



Bước 4 : Truy cập vào phần **Database Access**

Chọn **Add New Database User**



Điền username và password vào ô tương ứng như hình dưới, sau đó nhấn **Add User**. Hệ thống sẽ tạo cho ta 1 tài khoản để kết nối đến database

Add New Database User

Create a database user to grant an application or user, access to databases and collections in your clusters in this Atlas project. Granular access control can be configured with default privileges or custom roles. You can grant access to an Atlas project or organization using the corresponding [Access Manager](#).

Authentication Method

Password **Certificate** **AWS IAM** (MongoDB 4.4 and up) **PREVIEW** Federated Auth (MongoDB 7.0 and up)

MongoDB uses **SCRAM** as its default authentication method.

Password Authentication

Username: **Password**: [SHOW](#)

[Autogenerate Secure Password](#) [Copy](#)

Database User Privileges

Configure role based access control by assigning database \$user a mix of one built-in role, multiple custom roles, and multiple specific privileges. A user will gain access to all actions within the roles assigned to them, not just the actions those roles share in common. **You must choose at least one role or privilege.** [Learn more about roles.](#)

Built-In Role
Select one **built-in role** for this user. **0 SELECTED**

[Add Built In Role](#)

Custom Roles
Select your **pre-defined custom role(s)**. Create a custom role in the **Custom Roles** tab.

Specific Privileges
Select multiple privileges and what database and collection they are associated with. Leaving collection blank will grant this role for all collections in the database.

Restrict Access to Specific Clusters/Federated Database Instances
Enable to specify the resources this user can access. By default, all resources in this project are accessible. ☐

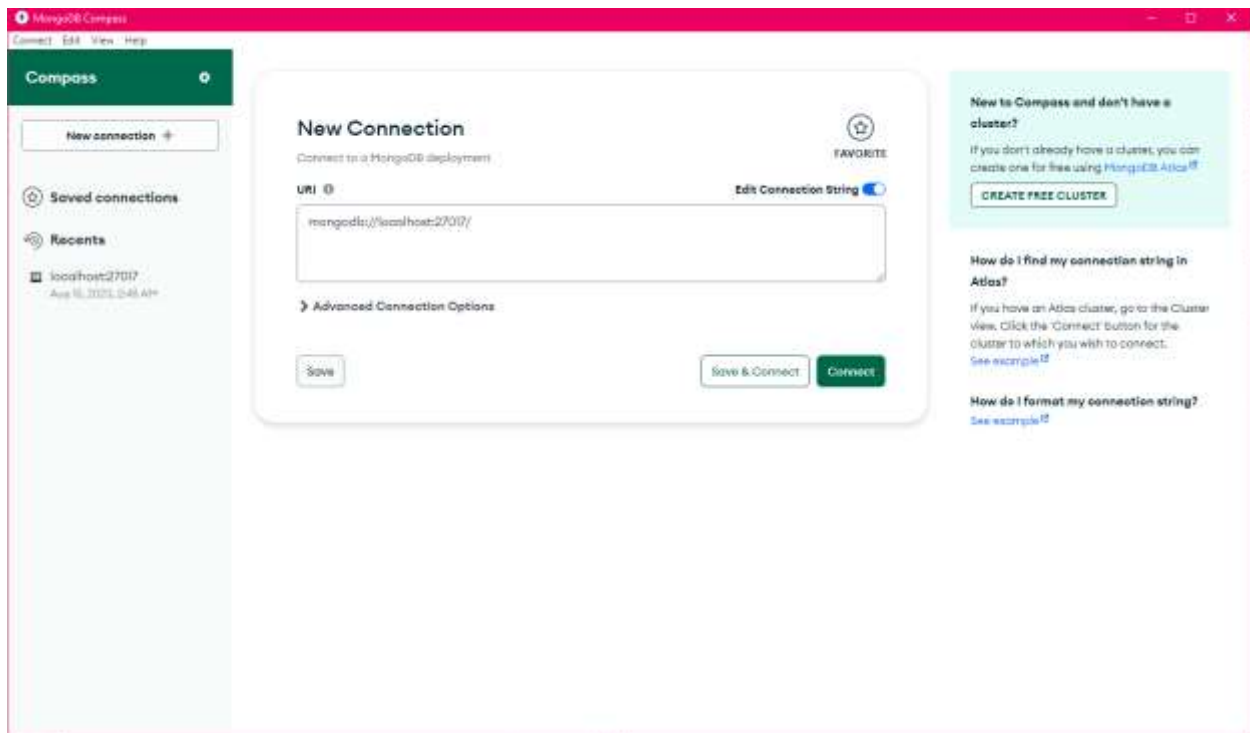
Temporary User
This user is temporary and will be deleted after your specified duration of 6 hours, 1 day, or 1 week. ☐

[Cancel](#) [Add User](#)

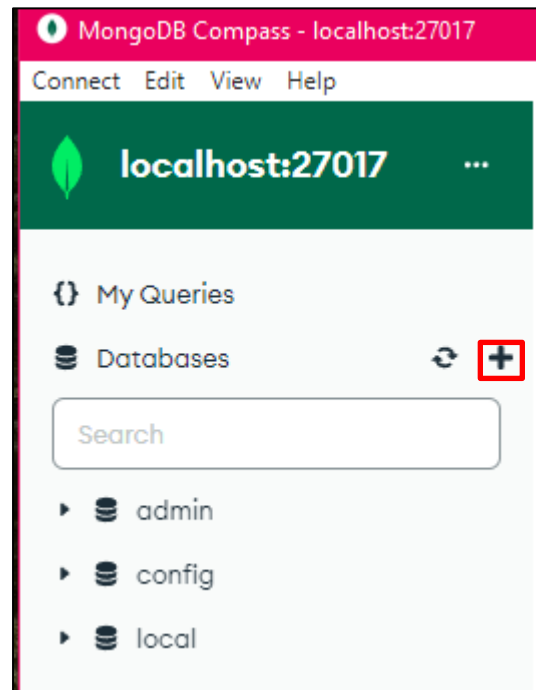
*Tạo database bằng MongoDB Compass

Bước 1: Mở phần mềm MongoDB Compass

Nhấn Connect

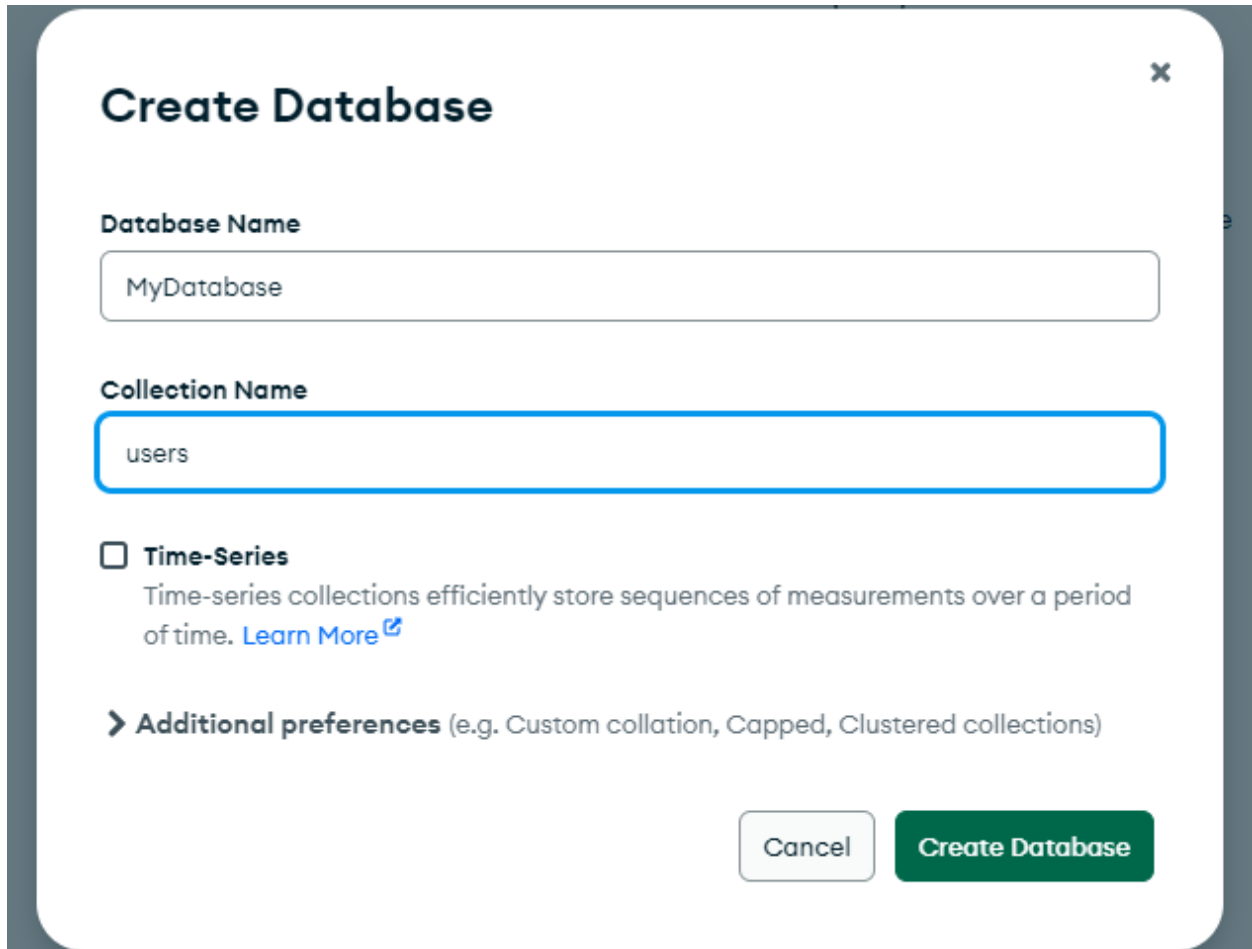


Nhấn vào biểu tượng dấu +



Điền tên Database và điền tên Collection đầu tiên. Sau đó nhấn **Create Database**

*Lưu ý: Collection Name của MongoDB nên database ở dạng số nhiều trong tiếng anh, student => students , user => users



Create Database ×

Database Name

MyDatabase

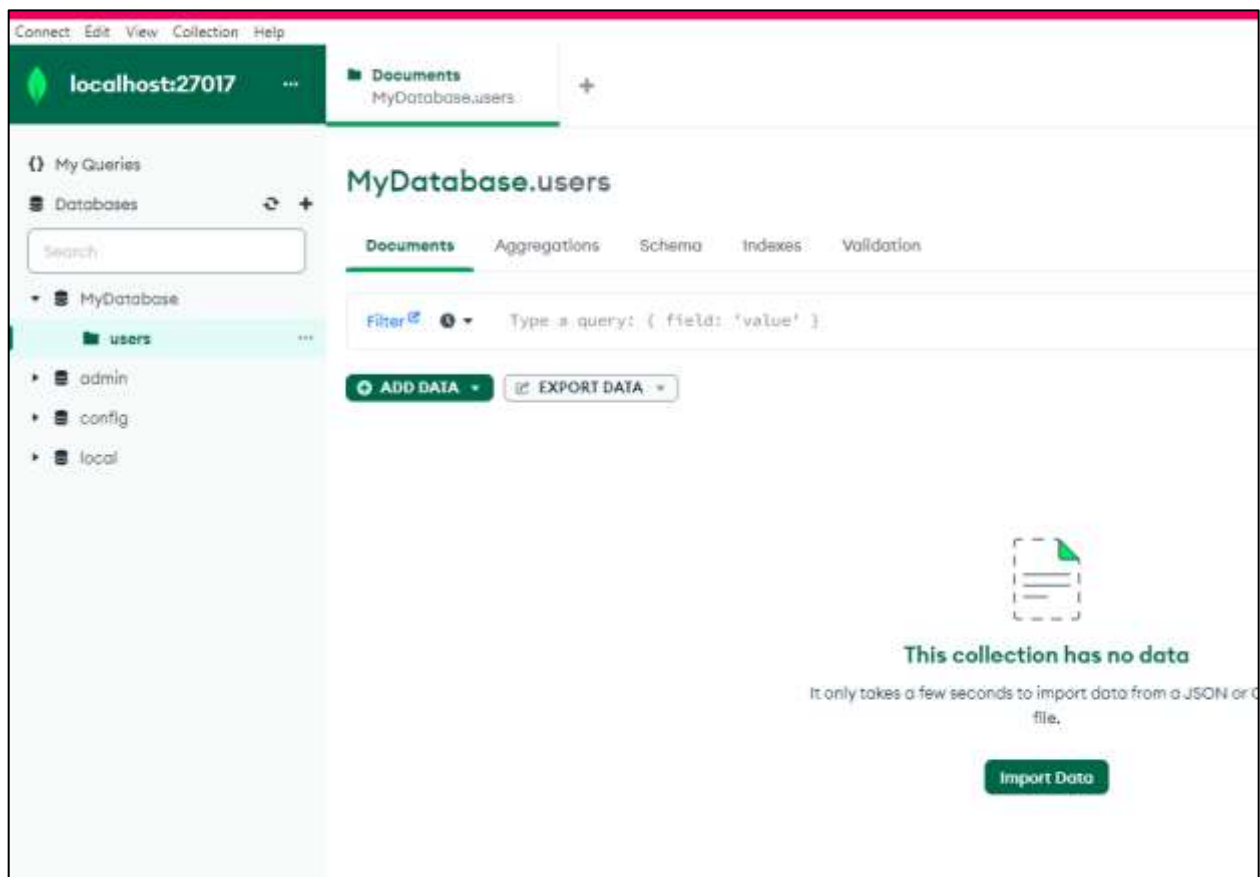
Collection Name

users

☐ **Time-Series**
Time-series collections efficiently store sequences of measurements over a period of time. [Learn More](#)

> Additional preferences (e.g. Custom collation, Capped, Clustered collections)

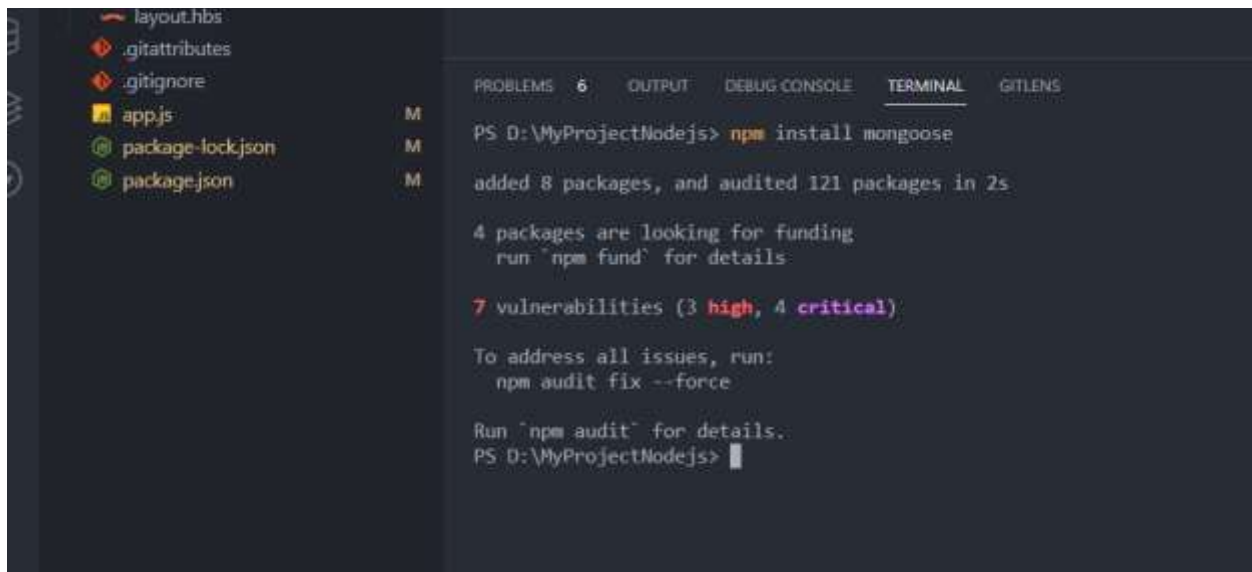
Cancel Create Database



*Kết nối database

Bước 1: Thêm thư viện mongdb vào project bằng cách mở **terminal**

Nhập dòng lệnh: **npm install mongoose**



```
layout.hbs
.gitattributes
.gitignore
app.js
package-lock.json
package.json

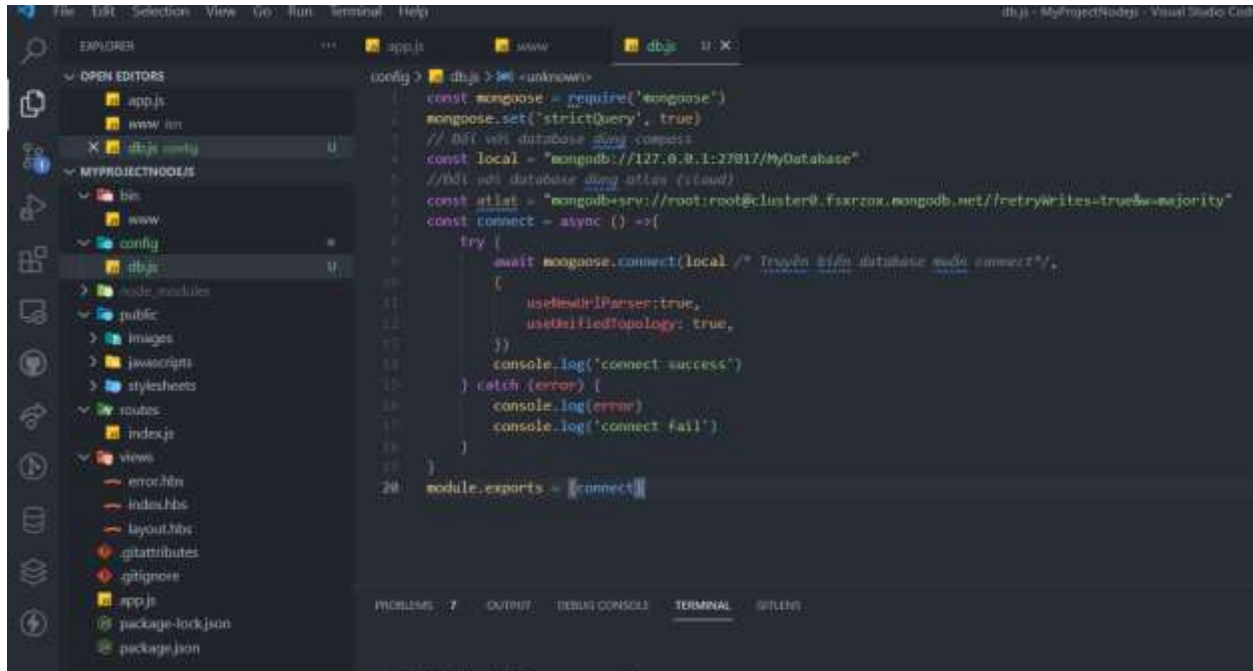
PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL GITLENS
PS D:\MyProjectNodejs> npm install mongoose
added 8 packages, and audited 121 packages in 2s

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

7 vulnerabilities (3 high, 4 critical)
To address all issues, run:
  npm audit fix --force

Run `npm audit` for details.
PS D:\MyProjectNodejs>
```

Bước 2: Tại thư mục gốc tạo thêm một folder đặt tên là **config**, tại folder **config** tạo một file **db.js (config/db.js)**



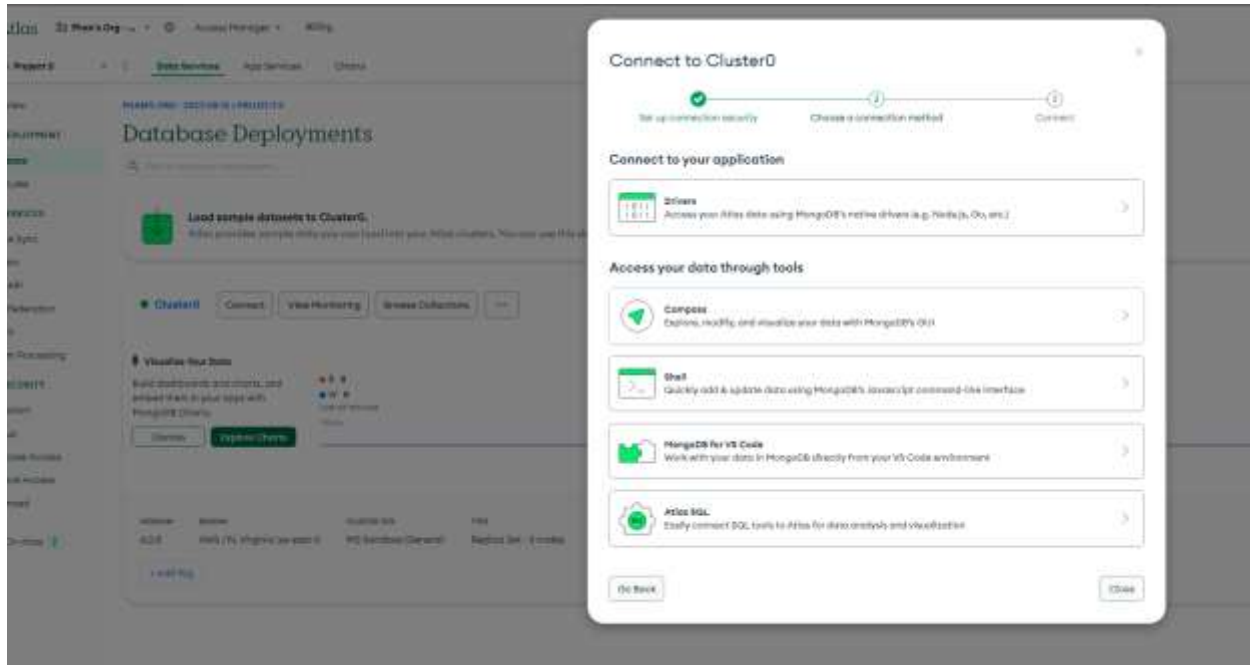
```
const mongoose = require('mongoose')
mongoose.set('strictQuery', true)
// Kết nối với database dùng compass
const local = "mongodb://127.0.0.1:27017/MyDatabase"
// Kết nối với database dùng atlas (cloud)
const atlas = "mongodb+srv://root:root@cluster0.faxrxox.mongodb.net/?retryWrites=true&w=majority"
const connect = async () =>{
  try {
    await mongoose.connect(local /* Thay tên database muốn connect */, {
      useNewUrlParser: true,
      useUnifiedTopology: true,
    })
    console.log('connect success')
  } catch (error) {
    console.log(error)
    console.log('connect fail')
  }
}
module.exports = {connect}
```

***Lưu ý:**

- Đối với connect bằng database compass ta sử dụng:

```
mongodb://127.0.0.1:27017/Tên_database_của_bạn
```

- Đối với connect bằng atlas, ta mở atlas lên chọn **database** rồi nhấn **connect**



Chọn **Drivers** . Copy đoạn khung đỏ bên hình dưới

Connect to Cluster0

✓

✓

3

Set up connection securityChoose a connection methodConnect

Connecting with MongoDB Driver

1. Select your driver and version

We recommend installing and using the latest driver version.

| Driver | Version |
|---------|--------------|
| Node.js | 5.5 or later |

2. Install your driver

Run the following on the command line

```
npm install mongodb
```

[View MongoDB Node.js Driver installation instructions.](#)

3. Add your connection string into your application code

☐ View full code sample

```
mongodb+srv://root:<password>@cluster0.fsxrzox.mongodb.net/?retryWrites=true&w=majority
```

Replace **<password>** with the password for the **root** user. Ensure any option params are [URL encoded](#).

RESOURCES

[Get started with the Node.js Driver](#)
[Access your Database Users](#)

[Node.js Starter Sample App](#)
[Troubleshoot Connections](#)

Go Back

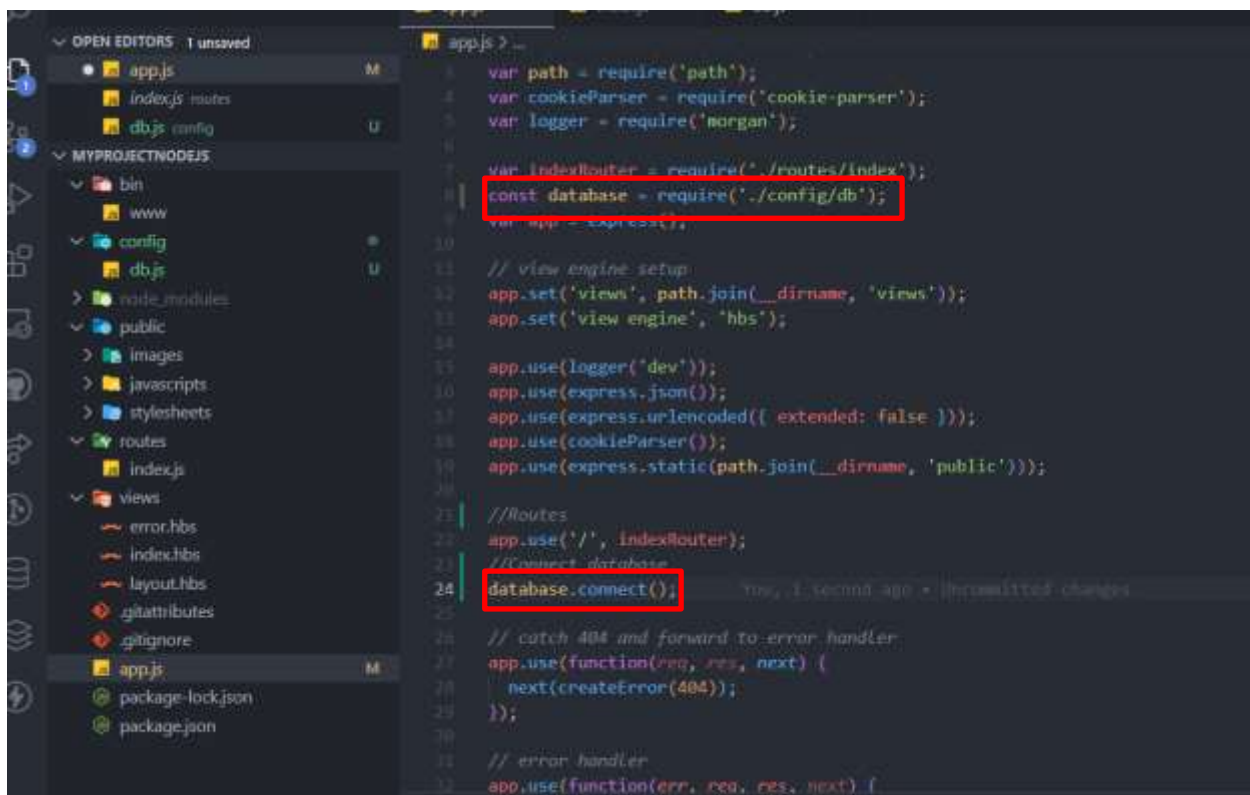
Close

Thay phần username, và password tạo ở bước trên.

Ví dụ:

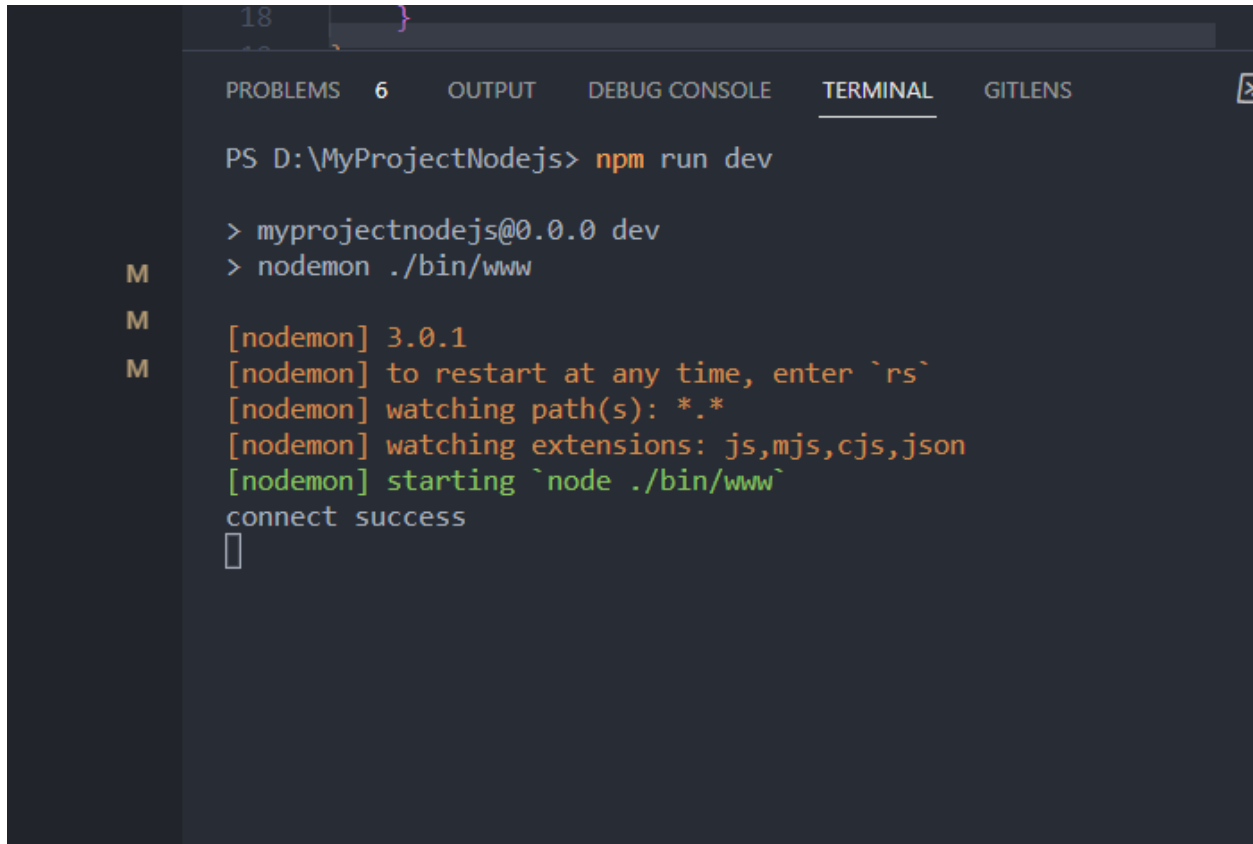
```
const atlas =  
"mongodb+srv://dinhnt24:RestAPI@123@cluster0.fsxrzox.mongodb.net/?retryWrites=true&w=majority"
```

Bước 3: Sau đó truy cập vào file **app.js**. Import **database** và **connect**



```
app.js > ...  
1  var path = require('path');  
2  var cookieParser = require('cookie-parser');  
3  var logger = require('morgan');  
4  
5  var indexRouter = require('./routes/index');  
6  const database = require('./config/db');  
7  var app = express();  
8  
9  
10  
11 // view engine setup  
12 app.set('views', path.join(__dirname, 'views'));  
13 app.set('view engine', 'hbs');  
14  
15 app.use(logger('dev'));  
16 app.use(express.json());  
17 app.use(express.urlencoded({ extended: false }));  
18 app.use(cookieParser());  
19 app.use(express.static(path.join(__dirname, 'public')));  
20  
21 // Routes  
22 app.use('/', indexRouter);  
23 // Connect database  
24 database.connect();  
25  
26 // catch 404 and forward to error handler  
27 app.use(function(req, res, next) {  
28   next(createError(404));  
29 });  
30  
31 // error handler  
32 app.use(function(err, req, res, next) {
```

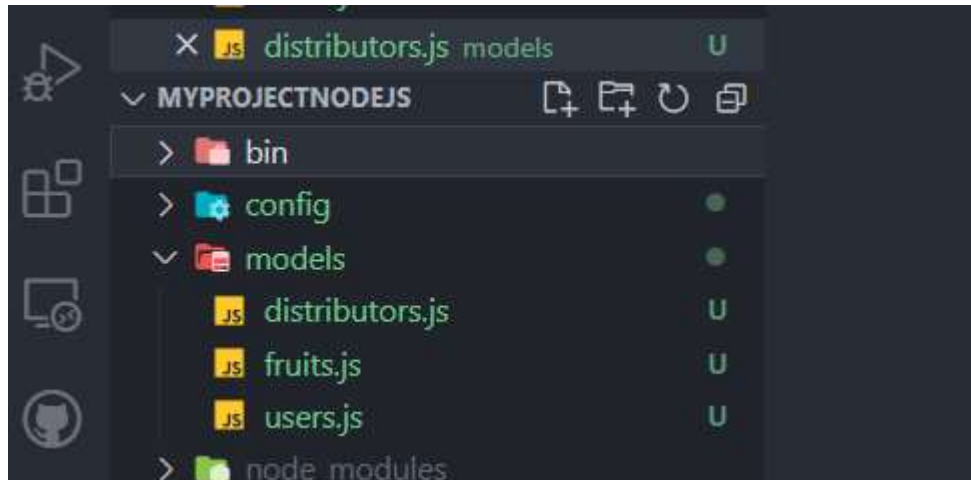
Bước 4: Chạy project và kiểm tra. Khi connect thành công sẽ có thông báo connect thành công.



```
18 }  
PROBLEMS 6 OUTPUT DEBUG CONSOLE TERMINAL GITLENS  
PS D:\MyProjectNodejs> npm run dev  
  
> myprojectnodejs@0.0.0 dev  
> nodemon ./bin/www  
  
[nodemon] 3.0.1  
[nodemon] to restart at any time, enter `rs`  
[nodemon] watching path(s): *.*  
[nodemon] watching extensions: js,mjs,cjs,json  
[nodemon] starting `node ./bin/www`  
connect success  
█
```


BÀI 2: TẠO MODEL

Tại folder gốc của project tạo một folder đặt tên là **models**



*Tạo model users

Mở file **users.js** (models/users.js). Tạo model user

```
JS users.js U X
models > JS users.js > ...
1  const mongoose = require('mongoose');
2  const Schema = mongoose.Schema;
3
4  const Users = new Schema({
5    username: {type: String, unique: true, maxLength: 255},
6    password : {type: String, maxLength: 255},
7    email: {type: String, unique: true},
8    name: {type: String},
9    avatar: {type: String},
10   available : {type: Boolean, default: false},
11
12  }, {
13    timestamps: true
14  })
15
16  module.exports = mongoose.model('user', Users)
17  /*
18   mongoose.model('user', User)
19   đặt tên collection, đặt ở dạng số ít
20   thư viện mongoose sẽ tự động tạo ra tên collection
21   số nhiều (user => users)
22  */
23  /*
24   Type: String, Boolean => kiểu dữ liệu
25   unique: true => không được trùng
26   maxLength: 255 => tối đa ký tự được nhập
27   default: false => giá trị mặc định là false
28   timestamps => Tạo ra 2 trường createdAt và updatedAt
29  */
30
```

*Tạo model distributors

Mở file **distributors.js** (models/distributors.js). Tạo model distributors

```
JS distributors.js U X
models > JS distributors.js > ...
1  const mongoose = require('mongoose');
2  const Schema = mongoose.Schema;
3
4  const Distributors = new Schema({
5    name: {type: String},
6
7  }, {
8    timestamps: true
9  })
10
11 module.exports = mongoose.model('distributor', Distributors)
```

*Tạo model fruits

Mở file **fruits.js** (models/fruits.js). Tạo model fruits

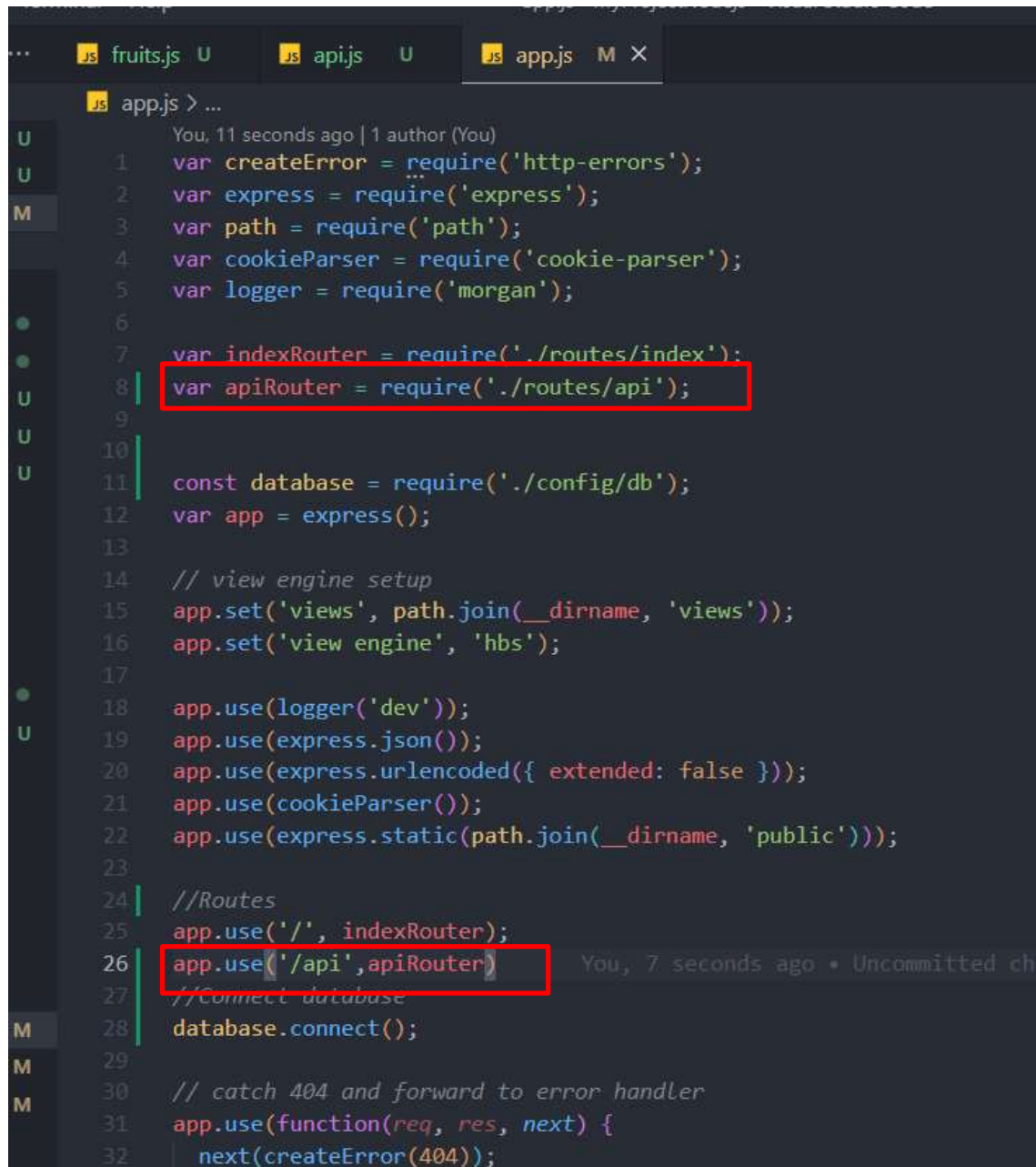
```
fruits.js U X
models > JS fruits.js > ...
1  const mongoose = require('mongoose');
2  const Schema = mongoose.Schema;
3
4  const Fruits = new Schema({
5    name: {type: String},
6    quantity: {type: Number},
7    price : {type : Number},
8    status : {type: Number}, // status = 1 => Còn hàng, 0 => Hết hàng, -1 => Ngừng kinh doanh,
9    image : {type : Array}, // Kiểu dữ liệu danh sách
10   description : {type: String},
11   id_distributor : {type: Schema.Types.ObjectId, ref: 'distributor'},
12
13 },{
14   timestamps: true
15 })
16 module.exports = mongoose.model('fruit', Fruits)
17 /*
18   type: Schema.Types.ObjectId => Kiểu dữ liệu id của mongodb
19   ref : khóa ngoại
20  */
```

BÀI 3: THÊM DISTRIBUTOR VÀ fruit (API POST)

*Setup routes api



Tại folder **routes** thêm một file đặt tên là **api.js** (routes/api.js)

Thêm route vào file **app.js**



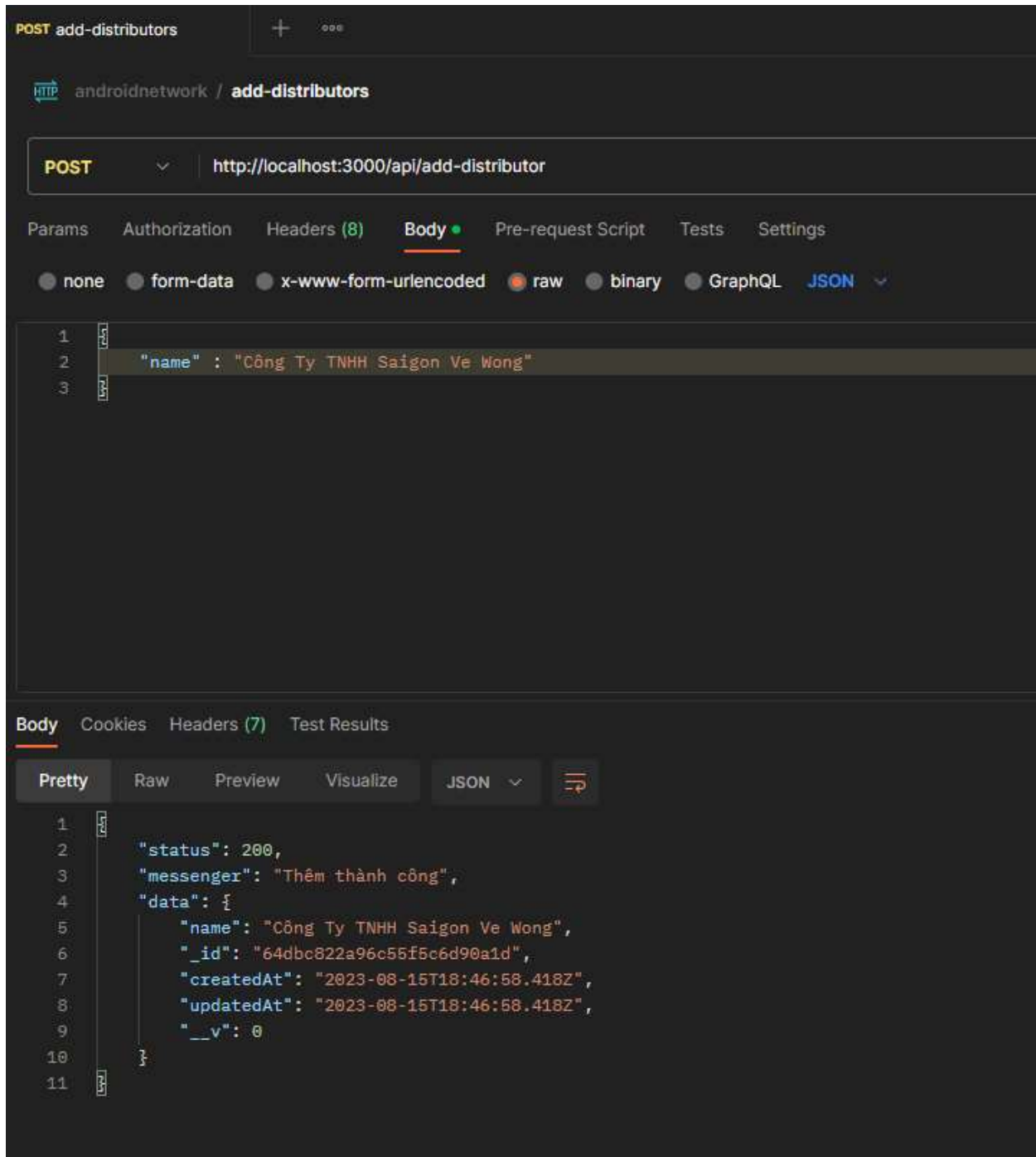
```
...  JS fruits.js  U    JS api.js  U    JS app.js  M X
JS app.js > ...
You, 11 seconds ago | 1 author (You)
U  1  var createError = require('http-errors');
U  2  var express = require('express');
M  3  var path = require('path');
  4  var cookieParser = require('cookie-parser');
  5  var logger = require('morgan');
  6
  7  var indexRouter = require('./routes/index');
  8  var apiRouter = require('./routes/api');
  9
 10
 11  const database = require('./config/db');
 12  var app = express();
 13
 14  // view engine setup
 15  app.set('views', path.join(__dirname, 'views'));
 16  app.set('view engine', 'hbs');
 17
 18  app.use(logger('dev'));
 19  app.use(express.json());
 20  app.use(express.urlencoded({ extended: false }));
 21  app.use(cookieParser());
 22  app.use(express.static(path.join(__dirname, 'public')));
 23
 24  //Routes
 25  app.use('/', indexRouter);
 26  app.use('/api', apiRouter);
 27  //Connect database
 28  database.connect();
 29
 30  // catch 404 and forward to error handler
 31  app.use(function(req, res, next) {
 32    next(createError(404));
```

*Thêm Distributor

routes >  api.js >  router.post('/add-distributor') callback

```
1
2  var express = require('express');
3  var router = express.Router();
4
5  //Thêm model
6  const Distributors = require('../models/distributors')
7  const Fruits = require('../models/fruits')
8  //Api thêm distributor
9  router.post('/add-distributor', async (req, res) => {
10    try {
11      const data = req.body; // Lấy dữ liệu từ body
12      const newDistributors = new Distributors({
13        name: data.name
14      }); //Tạo một đối tượng mới
15      const result = await newDistributors.save(); //Thêm vào database
16      if(result)
17      {
18        // Nếu thêm thành công result !null trả về dữ liệu
19        res.json({
20          "status" : 200,
21          "messenger" : "Thêm thành công",
22          "data" : result
23        })
24      }else
25      {
26        // Nếu thêm không thành công result null, thông báo không thành công
27        res.json({
28          "status" : 400 ,
29          "messenger" : "Lỗi, thêm không thành công",
30          "data" : []
31        })
32      }
33    } catch (error) {
34      console.log(error);
35    }
36  });
37
38
39  module.exports = router;
```

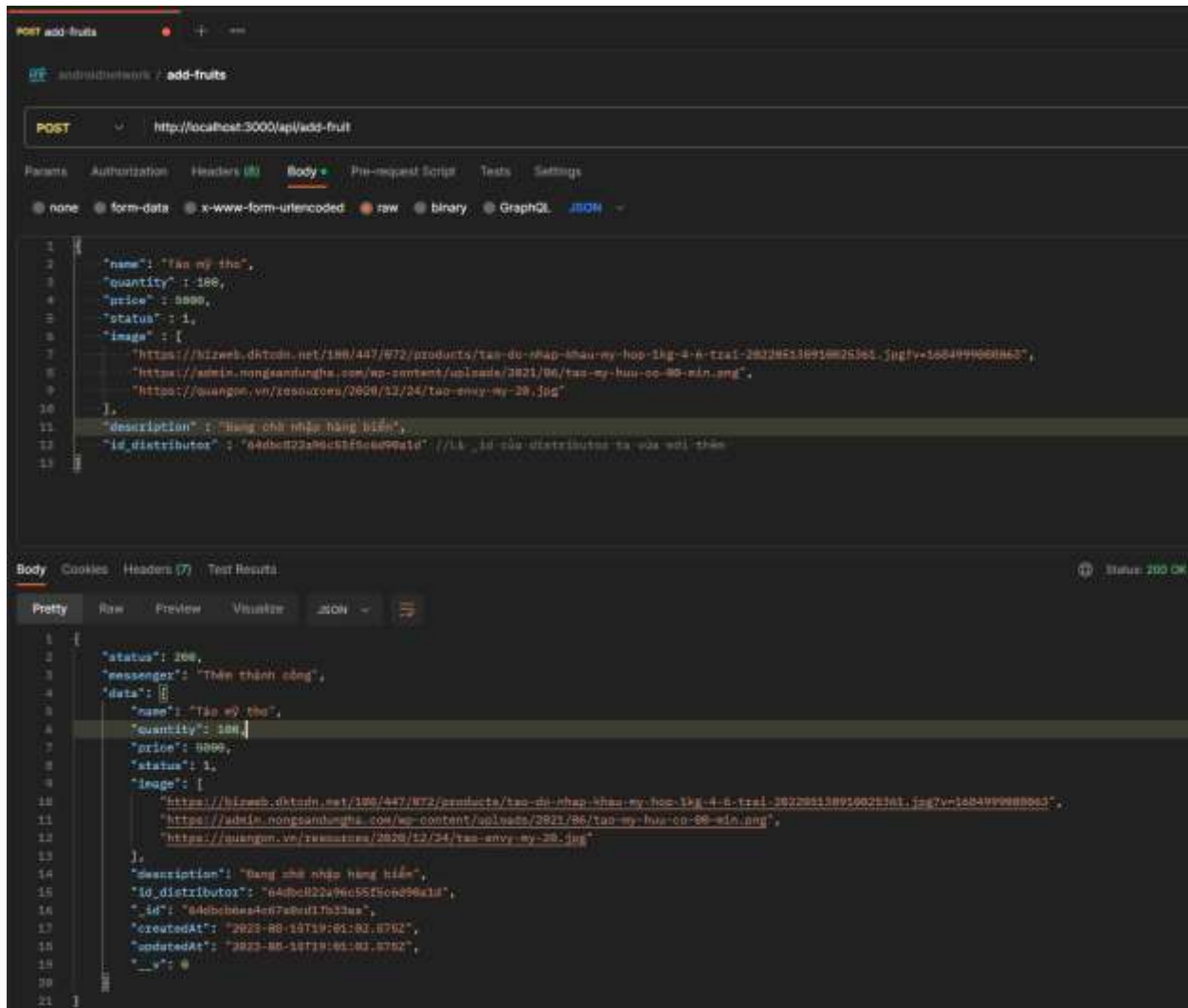
Sử dụng postman để test lại API mới hoàn thành tại bước trên



*Thêm fruits.js

```
//Api thêm fruit
router.post('/add-fruit', async (req, res) => {
  try {
    const data = req.body; // Lấy dữ liệu từ body
    const newfruit = new Fruits({
      name: data.name,
      quantity : data.quantity,
      price : data.price,
      status : data.status,
      image : data.image,
      description : data.description,
      id_distributor : data.id_distributor
    }); //Tạo một đối tượng mới
    const result = await newfruit.save(); //Thêm vào database
    if(result)
    {
      // Nếu thêm thành công result !null trả về dữ liệu
      res.json({
        "status" : 200,
        "messenger" : "Thêm thành công",
        "data" : result
      })
    }
  } else
  {
    // Nếu thêm không thành công result null, thông báo không thành công
    res.json({
      "status" : 400 ,
      "messenger" : "Lỗi, thêm không thành công",
      "data" : []
    })
  }
} catch (error) {
  console.log(error);
}
});
```

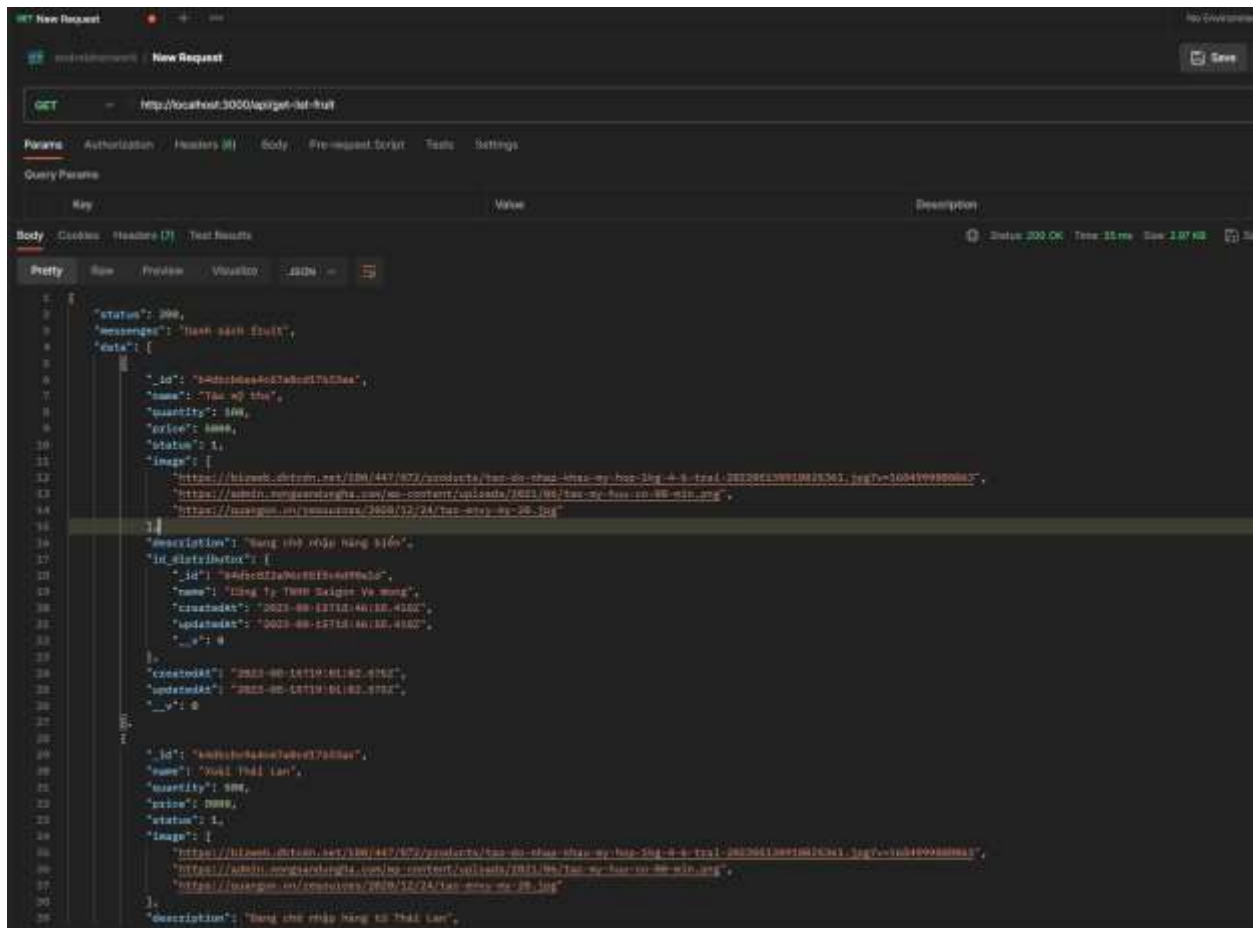

Sử dụng postman để test lại API mới hoàn thành tại bước trên



BÀI 4: TRUY VẤN DANH SÁCH FRUITS

*Get danh sách Fruits

```
router.get('/get-list-fruit', async (req, res) => {
  try {
    const data = await Fruits.find().populate('id_distributor');
    res.json({
      "status" : 200,
      "messenger" : "Danh sách fruit",
      "data" : data
    })
  } catch (error) {
    console.log(error);
  }
});
```



*Get chi tiết Fruits (truyền param id)

```

✓ router.get('/get-fruit-by-id/:id', async (req, res) => {
  //:id param
  ✓ try {
    const {id} = req.params // lấy dữ liệu thông qua :id trên url gọi là param
    const data = await Fruits.findById(id).populate('id_distributor');
  ✓ res.json({
    "status": 200,
    "messenger": "Danh sách fruit",
    "data": data
  })
  ✓ } catch (error) {
    console.log(error);
  }
});

```

GET get-fruit-by-id

androidnetwork / get-fruit-by-id

GET http://localhost:3000/api/get-fruit-by-id/64dbcb6ea4c67a0cd17b33aa

Params Authorization Headers (6) Body Pre-request Script Tests Settings

Body Cookies Headers (7) Test Results

Pretty Raw Preview Visualize JSON

```

1 {
2   "status": 200,
3   "messenger": "Danh sách fruit",
4   "data": [
5     {
6       "_id": "64dbcb6ea4c67a0cd17b33aa",
7       "name": "Táo Mỹ đỏ",
8       "quantity": 100,
9       "price": 5000,
10      "status": 1,
11      "image": [
12        "https://bizweb.dktcdn.net/100/447/872/products/tao-du-nhap-khau-my-hop-1kg-4-0-txai-2022051309100253h1.jpg?w=1654&h=866&3",
13        "https://admin.nongsandungha.co.th/wp-content/uploads/2021/06/tao-my-huu-so-60-min.png",
14        "https://quangtin.vn/resources/2020/12/24/tao-envy-my-20.jpg"
15      ],
16      "description": "Quả chín nhập hàng biển",
17      "id_distributor": {
18        "_id": "64dbcb22a96c55f5c5d9801d",
19        "name": "Công Ty TNHH Saigon Ve Wong",
20        "createdAt": "2023-08-15T10:46:58.418Z",
21        "updatedAt": "2023-08-15T10:46:58.418Z",
22        "__v": 0
23      },
24      "createdAt": "2023-08-15T19:01:02.575Z",
25      "updatedAt": "2023-08-15T10:01:02.575Z",
26      "__v": 0
27    }
28  ]
29 }

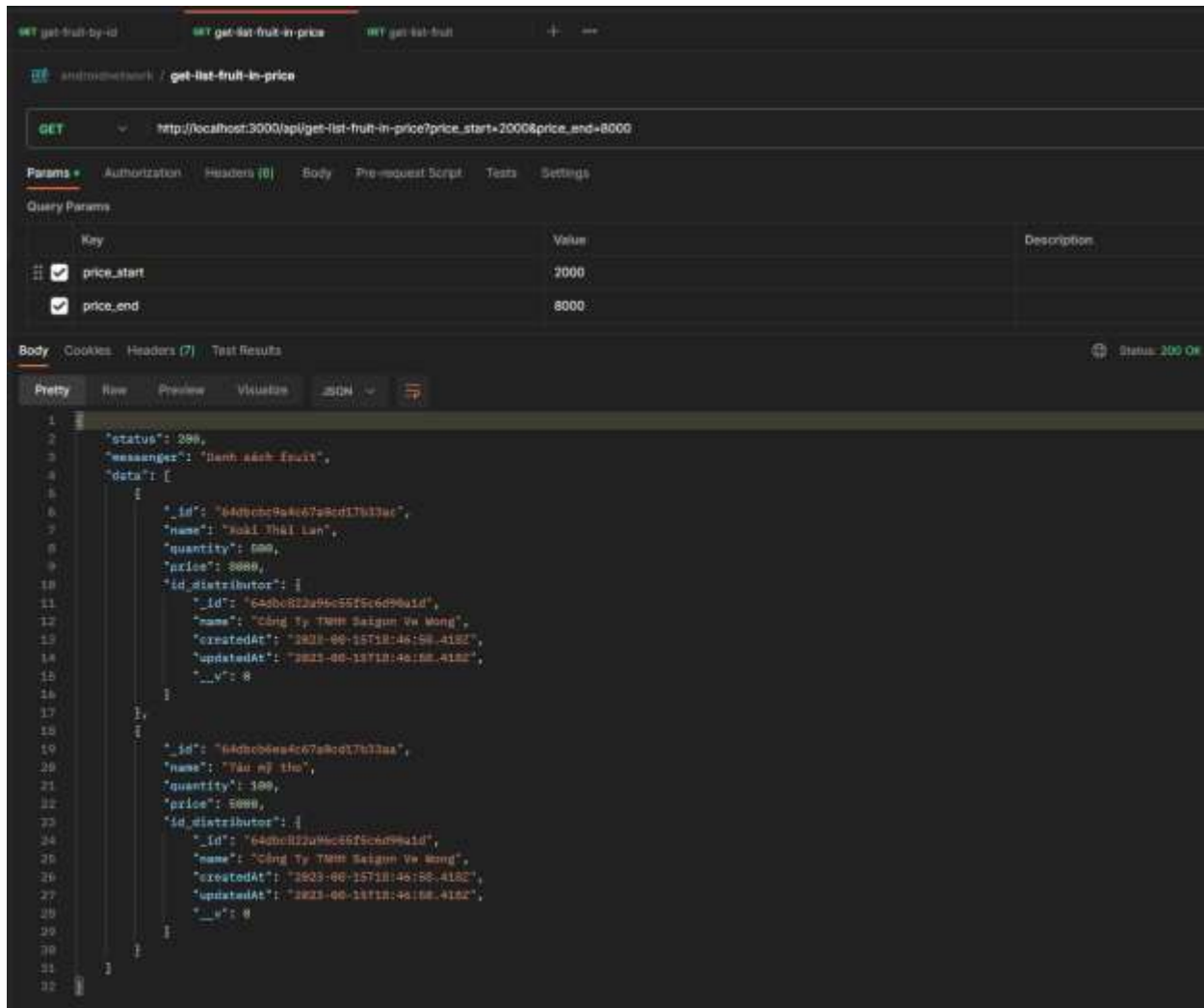
```

*Get danh sách Fruits (danh sách trả về gồm: name, quantity, price, id_distributor) nằm trong khoảng giá (query giá cao nhất, giá thấp nhất) và sắp xếp theo quantity (giảm dần)

```
router.get('/get-list-fruit-in-price', async (req, res) => {
  //:id param
  try {
    const {price_start, price_end} = req.query // lấy dữ liệu thông qua :id trên url gọi là param

    const query = {price: { $gte: price_start, $lte: price_end } }
    // $gte lớn hơn hoặc bằng, $gt lớn hơn
    // $lte nhỏ hơn hoặc bằng, $lt nhỏ hơn
    // truyền câu điều kiện, và chỉ lấy các trường mong muốn
    const data = await Fruits.find(query, 'name quantity price id_distributor')
      .populate('id_distributor')
      .sort({quantity: -1}) // giảm dần = -1, tăng dần = 1
      .skip(0) // bỏ qua số lượng row
      .limit(2) // lấy 2 sản phẩm

    res.json({
      "status": 200,
      "messenger": "Danh sách fruit",
      "data": data
    })
  } catch (error) {
    console.log(error);
  }
});
```



The screenshot shows a REST client interface with the following details:

- Request Method:** GET
- URL:** `http://localhost:3000/api/get-list-fruit-in-price?price_start=2000&price_end=8000`
- Params:**
 - price_start:** 2000
 - price_end:** 8000
- Status:** 200 OK
- Response Body (JSON):**

```

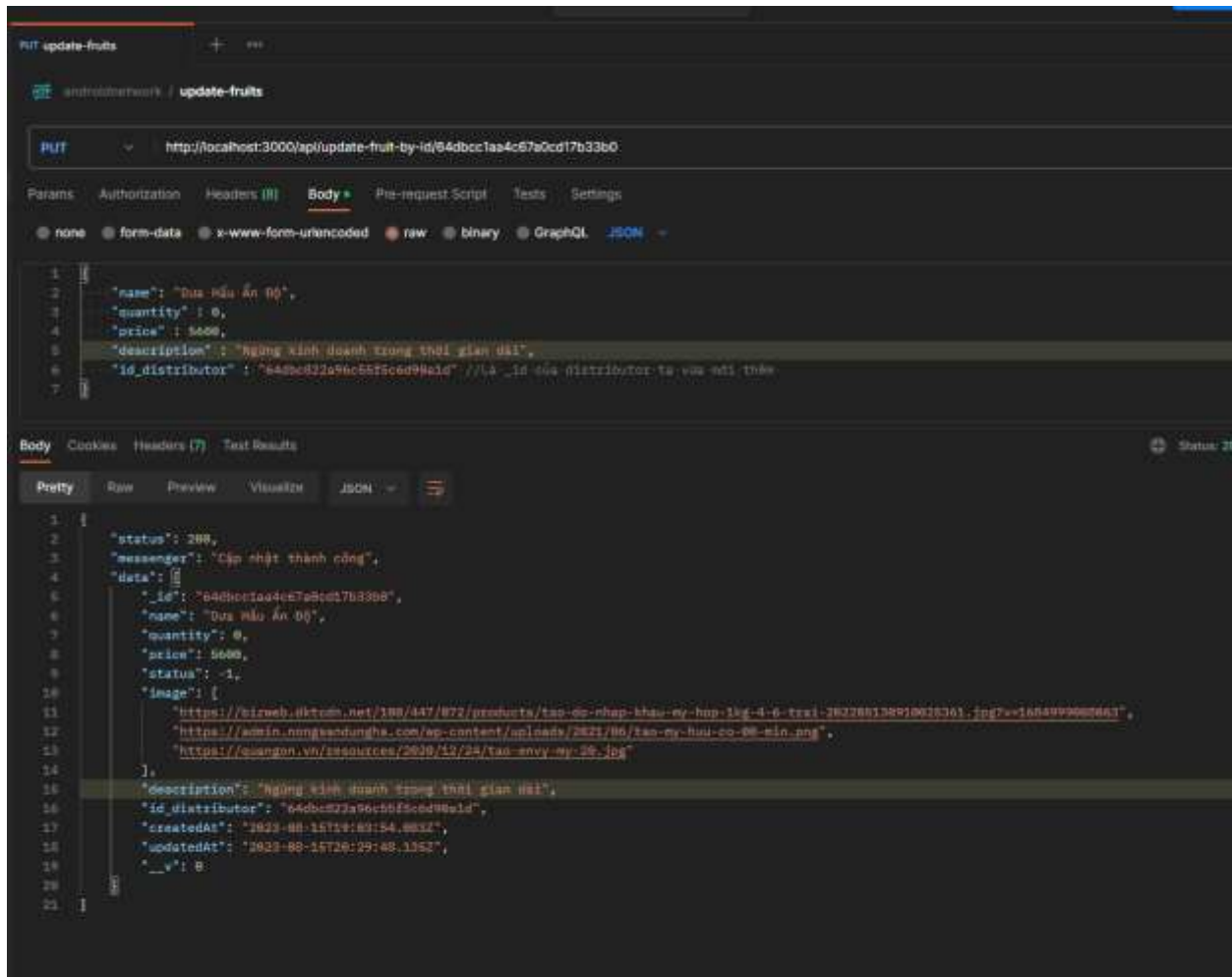
1 {
2   "status": 200,
3   "message": "Danh sách fruit",
4   "data": [
5     {
6       "_id": "64dbcb9a4c67a8cd17b33ac",
7       "name": "Khôl Thái Lan",
8       "quantity": 500,
9       "price": 8888,
10      "id_distributor": {
11        "_id": "64dbcb82a9cc55f5c6d99a1d",
12        "name": "Công Ty TNHH Saigon Vu Mong",
13        "createdAt": "2023-08-15T13:46:58.418Z",
14        "updatedAt": "2023-08-15T13:46:58.418Z",
15        "__v": 0
16      }
17    },
18    {
19      "_id": "64dbcb9a4c67a8cd17b33aa",
20      "name": "Táo Mỹ đỏ",
21      "quantity": 100,
22      "price": 5888,
23      "id_distributor": {
24        "_id": "64dbcb82a9cc55f5c6d99a1d",
25        "name": "Công Ty TNHH Saigon Vu Mong",
26        "createdAt": "2023-08-15T13:46:58.418Z",
27        "updatedAt": "2023-08-15T13:46:58.418Z",
28        "__v": 0
29      }
30    }
31  ]
32 }
```

***Get danh sách Fruits (danh sách trả về gồm: name, quantity, price, id_distributor)
có chữ cái bắt đầu tên là A hoặc X**

```
router.get('/get-list-fruit-have-name-a-or-x', async (req, res) => {  
  //:id param  
  try {  
    const query = {$or: [  
      {name: {$regex: 'T'}},  
      {name: {$regex: 'X'}},  
    ]}  
  
    // truyền câu điều kiện , và chỉ lấy các trường mong muốn  
    const data = await Fruits.find( query, 'name quantity price id_distributor')  
      .populate('id_distributor')  
  
    res.json({  
      "status" : 200,  
      "messenger" : "Danh sách fruit",  
      "data" : data  
    })  
  } catch (error) {  
    console.log(error);  
  }  
});
```

BÀI 5: CẬP NHẬT FRUITS BẰNG ID (PUT)

```
//Api cập nhật fruit
router.put('/update-fruit-by-id/:id', async (req, res) => {
  try {
    const {id} = req.params
    const data = req.body; // Lấy dữ liệu từ body
    const updatefruit = await Fruits.findById(id)
    let result = null;
    if(updatefruit)
    {
      updatefruit.name = data.name ?? updatefruit.name;
      updatefruit.quantity = data.quantity ?? updatefruit.quantity,
      updatefruit.price = data.price ?? updatefruit.price,
      updatefruit.status = data.status ?? updatefruit.status,
      updatefruit.image = data.image ?? updatefruit.image,
      updatefruit.description = data.description ?? updatefruit.description,
      updatefruit.id_distributor = data.id_distributor ?? updatefruit.id_distributor
      result = await updatefruit.save();
    }
    //Tạo một đối tượng mới
    //Thêm vào database
    if(result)
    {
      // Nếu thêm thành công result !null trả về dữ liệu
      res.json({
        "status" : 200,
        "messenger" : "Cập nhật thành công",
        "data" : result
      })
    }
    else
    {
      // Nếu thêm không thành công result null, thông báo không thành công
      res.json({
        "status" : 400 ,
        "messenger" : "Lỗi, Cập nhật không thành công",
        "data" : []
      })
    }
  } catch (error) {
    console.log(error);
  }
});
```



*** YÊU CẦU NỘP BÀI:

Sv nén file bao gồm các yêu cầu đã thực hiện trên, nộp lms đúng thời gian quy định của giảng viên. Không nộp bài coi như không có điểm.

--- Hết ---