



FPT POLYTECHNIC



android

www.poly.edu.vn

LẬP TRÌNH ANDROID VỚI RESTAPI

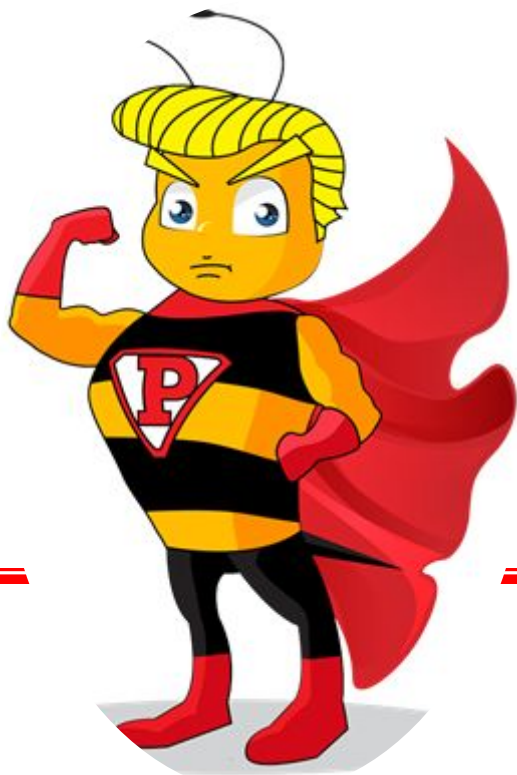
KẾT NỐI DATABASE MONGODB VÀ VIẾT API CƠ BẢN

- ☐ Kết nối dự án với database mongoDB
- ☐ Các truy vấn cơ bản trong mongoDB
- ☐ Các phương thức API
- ☐ Viết API cơ bản

MỤC TIÊU

- ◎ KẾT NỐI DỰ ÁN VỚI DATABASE MONGODB
- ◎ CÁC PHƯƠNG THỨC TRUY VẤN CƠ BẢN TRONG MONGODB
- ◎ CÁC PHƯƠNG THỨC API
- ◎ VIẾT API CƠ BẢN



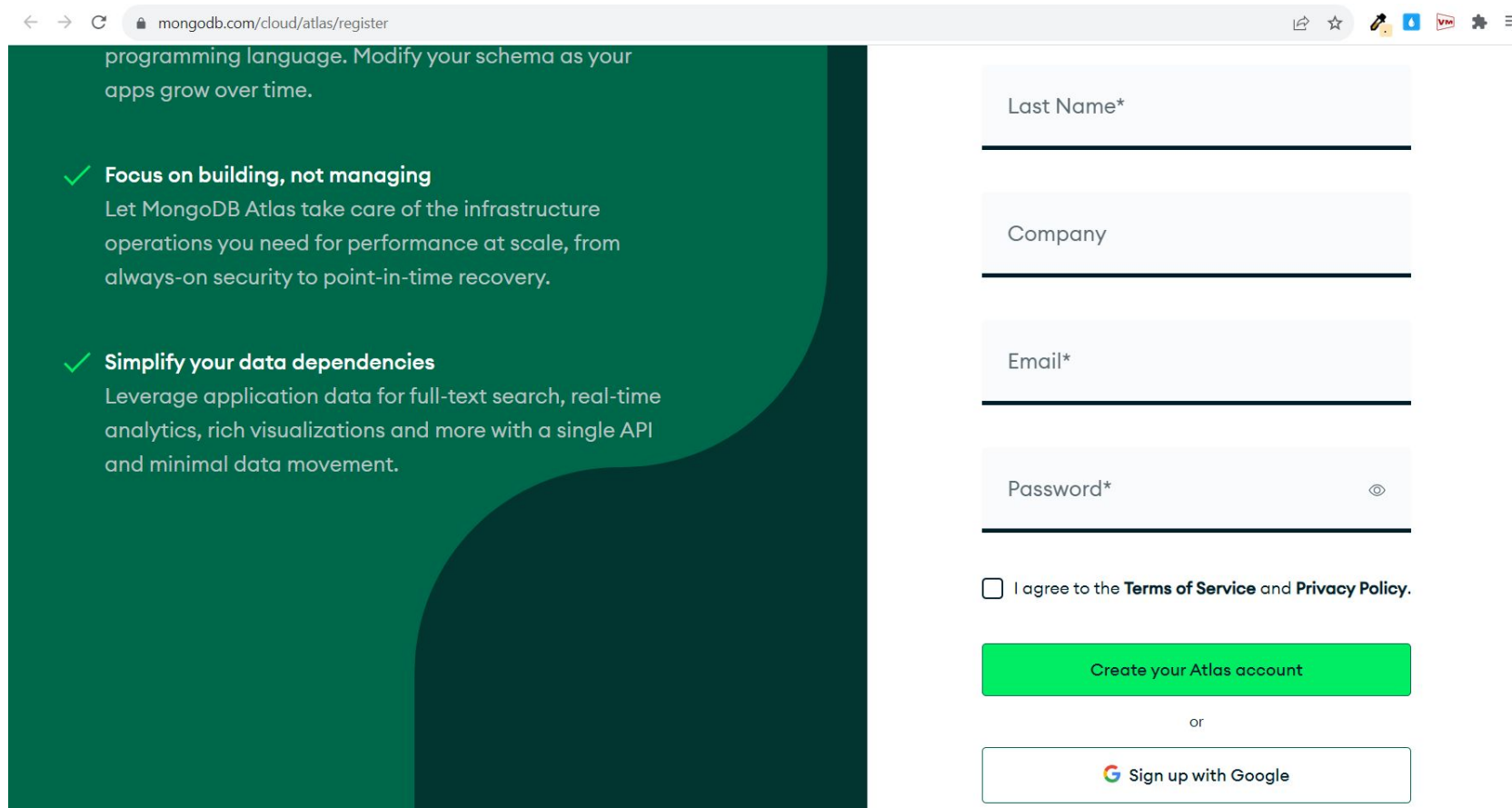


KẾT NỐI DỰ ÁN VỚI DATABASE MONGO DB

...

Truy cập trang web <https://www.mongodb.com/cloud/atlas/register>

Đăng ký tài khoản mới hoặc đăng nhập bằng tài khoản google



← → ↻ mongodb.com/cloud/atlas/register


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.

Last Name*

Company


Email*

Password* 

☐ I agree to the **Terms of Service** and **Privacy Policy**.

Create your Atlas account

or

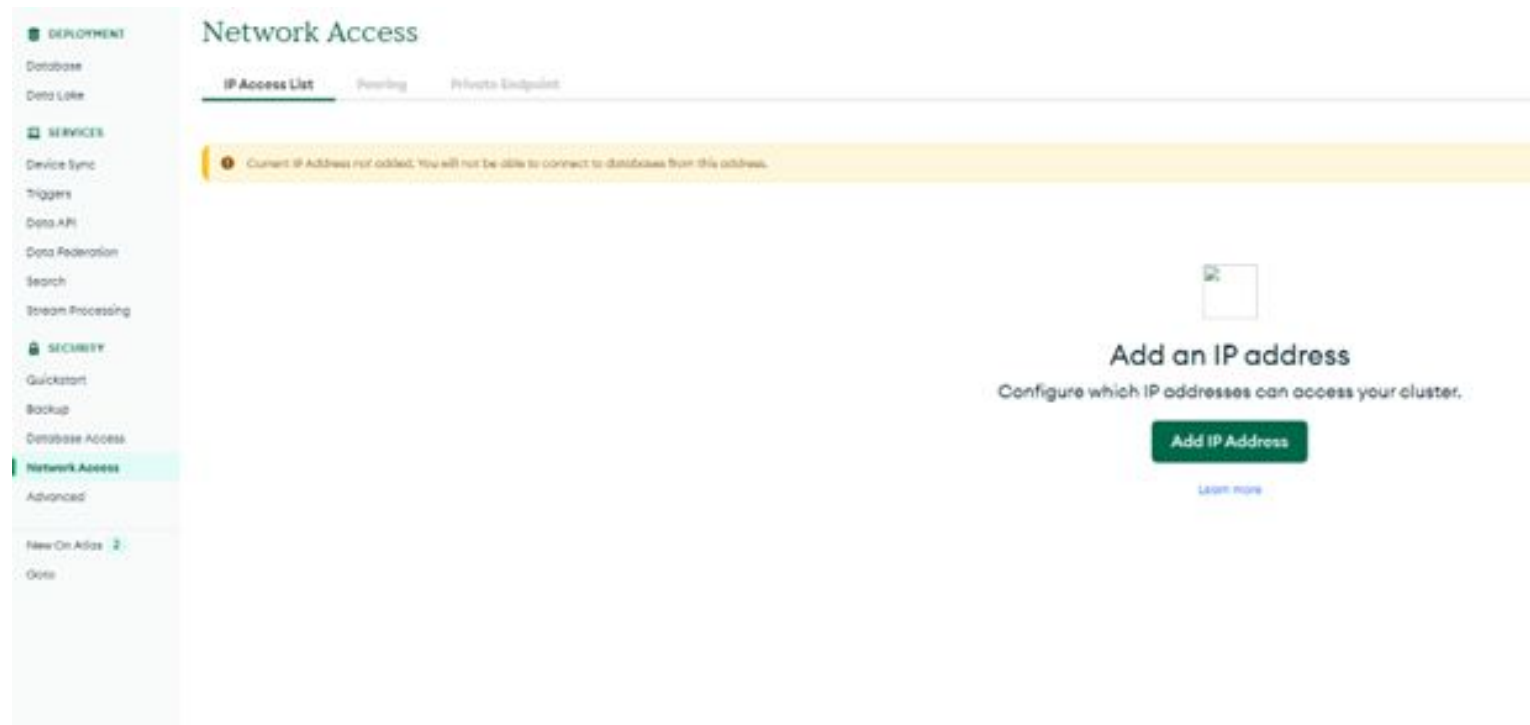
 Sign up with Google

- ❑ Sau khi đăng nhập sẽ có giao diện như hình dưới đây

The screenshot shows the MongoDB Atlas web interface. At the top, there's a navigation bar with the Atlas logo, a dropdown menu for 'Phan's Org - ...', and links for 'Access Manager' and 'Billing'. Below this is a secondary navigation bar with 'Project 0', 'Data Services' (highlighted), 'App Services', and 'Charts'. The left sidebar contains a menu with sections: 'Overview', 'DEPLOYMENT', 'Database' (highlighted), 'Data Lake', 'SERVICES', 'Device Sync', 'Triggers', 'Data API', 'Data Federation', 'Search', 'Stream Processing', 'SECURITY', 'Quickstart', 'Backup', 'Database Access', 'Network Access', 'Advanced', 'New On Atlas 2', and 'Goto'. The main content area has a yellow warning banner at the top: 'Current IP Address not added. You will not be able to connect to databases from this address.' Below this is the breadcrumb 'PHAN'S ORG - 2023-08-15 > PROJECT 0' and the title 'Database Deployments'. There's a search bar 'Find a database deployment...'. A green box with a downward arrow icon says 'Load sample datasets to Cluster0.' with the text 'Atlas provides sample data you can load into your Atlas clusters. You can use this data to quickly'. Below this is a section for 'Cluster0' with buttons 'Connect', 'View Monitoring', 'Browse Collections', and a three-dot menu. Further down is a 'Visualize Your Data' section with the text 'Build dashboards and charts, and embed them in your apps with MongoDB Charts.' and buttons 'Dismiss' and 'Explore Charts'. At the bottom, there's a table with columns: VERSION, REGION, CLUSTER TIER, TYPE, and BACKUPS. The table has one row: 6.0.8, AWS / N. Virginia (us-east-1), M0 Sandbox (General), Replica Set - 3 nodes, Inactive. There's a '+ Add Tag' button below the table.

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

- ❑ Tiếp theo, chọn **Network Access** để cấp quyền cho tất cả các IP được thao tác với database
- ❑ Tại đây chọn **Add IP Address**



- ❑ Tiếp tục chọn **ALLOW ACCESS FROM ANYWHERE**, sau đó nhấn **Confirm**

Add IP Access List Entry ✕

Atlas only allows client connections to a cluster from entries in the project's IP Access List. Each entry should either be a single IP address or a CIDR-notated range of addresses. [Learn more.](#)

ADD CURRENT IP ADDRESS **ALLOW ACCESS FROM ANYWHERE**

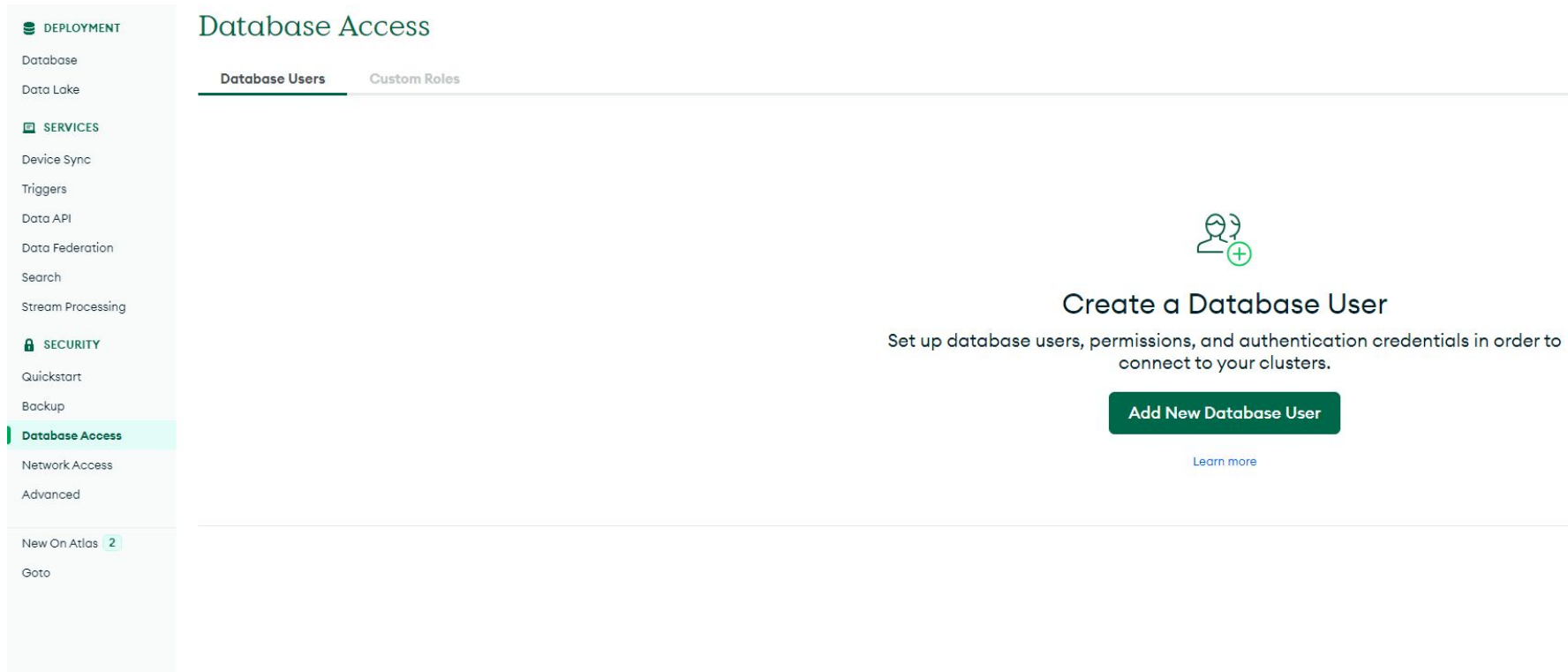
Access List Entry:

Comment:

☐ This entry is temporary and will be deleted in 6 hours


Cancel **Confirm**

Tại **Database Access** chọn **Add new Database User**



Database Access

Database Users Custom Roles



Create a Database User

Set up database users, permissions, and authentication credentials in order to connect to your clusters.

Add New Database User

[Learn more](#)

❑ Nhập tài khoản và mật khẩu, sau đó chọn **Add User**

Add New Database User

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

Authentication Method

Password

Certificate

AWS IAM
(MongoDB 4.4 and up)

Federated
(MongoDB 4.4 and up)

MongoDB uses [SCRAM](#) as its default authentication method.

Password Authentication

root

.....

SHOW

Autogenerate Secure Password

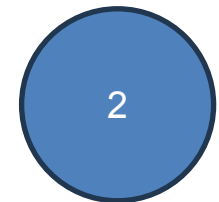
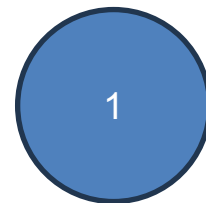
Copy

Database User Privileges

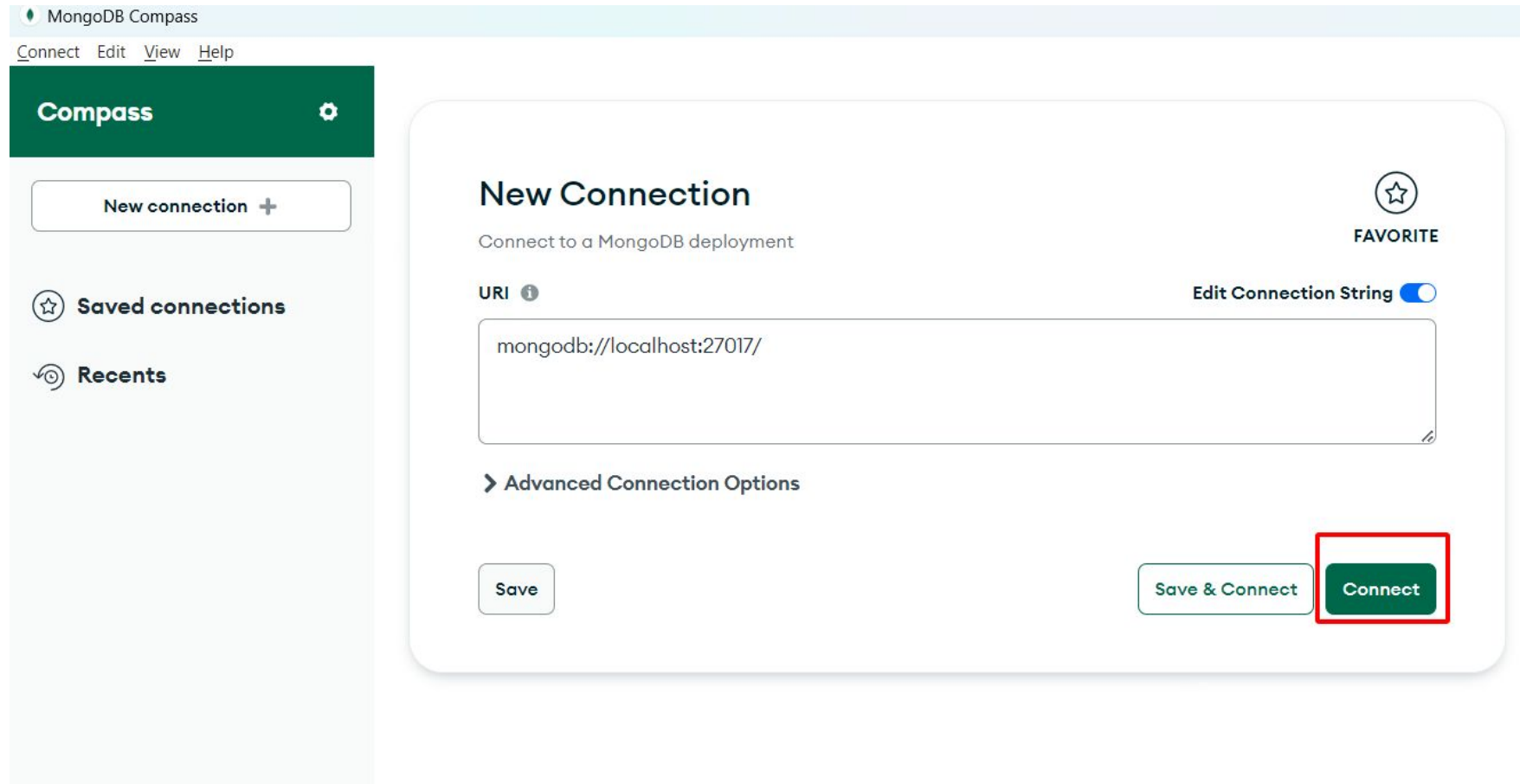
Configure role based access control by assigning database \$user a mix of one built-in role, multiple custom roles, an

Cancel

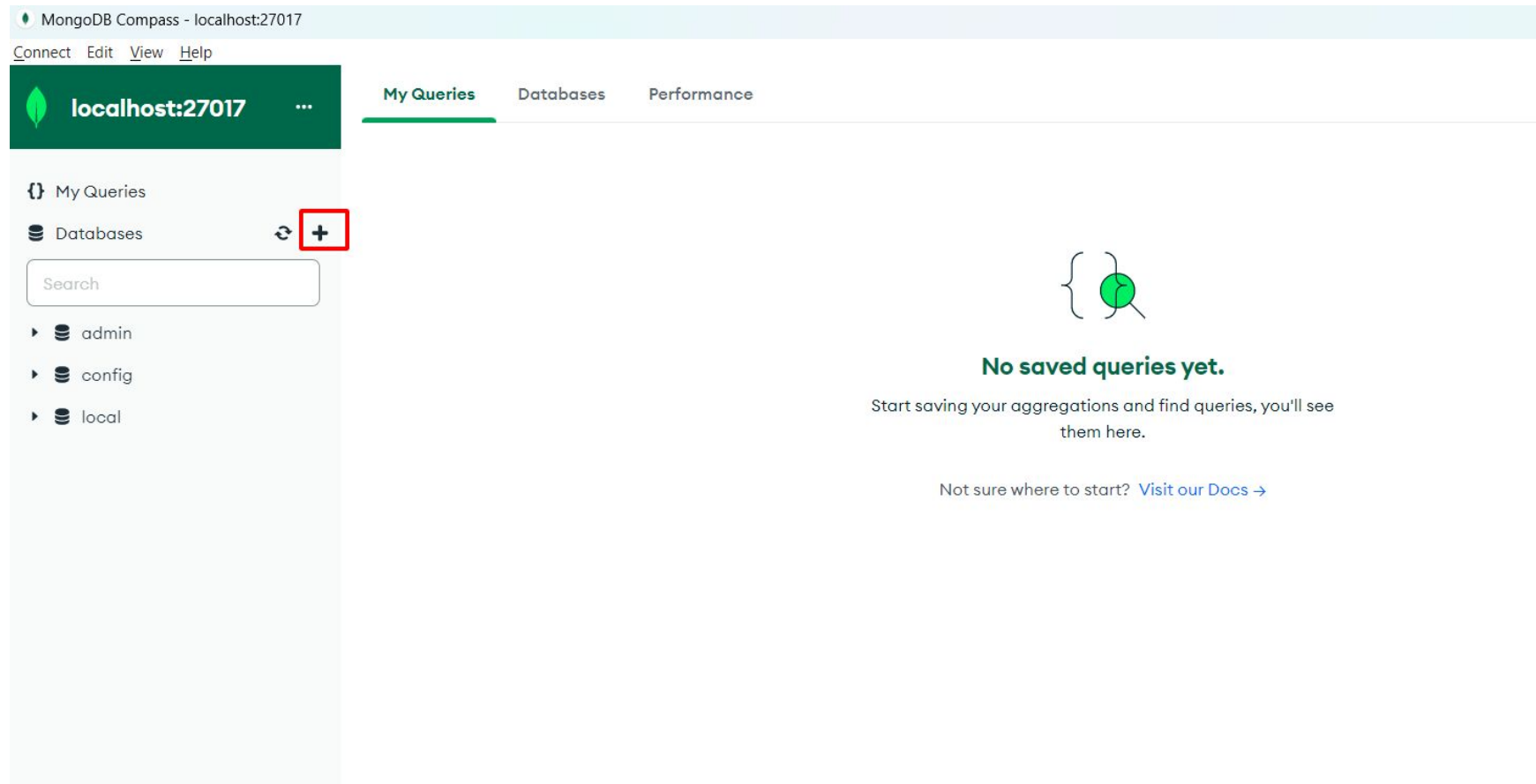
Add User



❑ Mở MongoDB Compass, chọn **Connect**



□ Tiếp theo nhấn vào biểu tượng + để thêm database mới



- ❑ Nhập **Database Name** và **Collection Name**, sau đó nhấn **Create Database**

Create Database ×

Database Name

MyDatabase

Collection Name

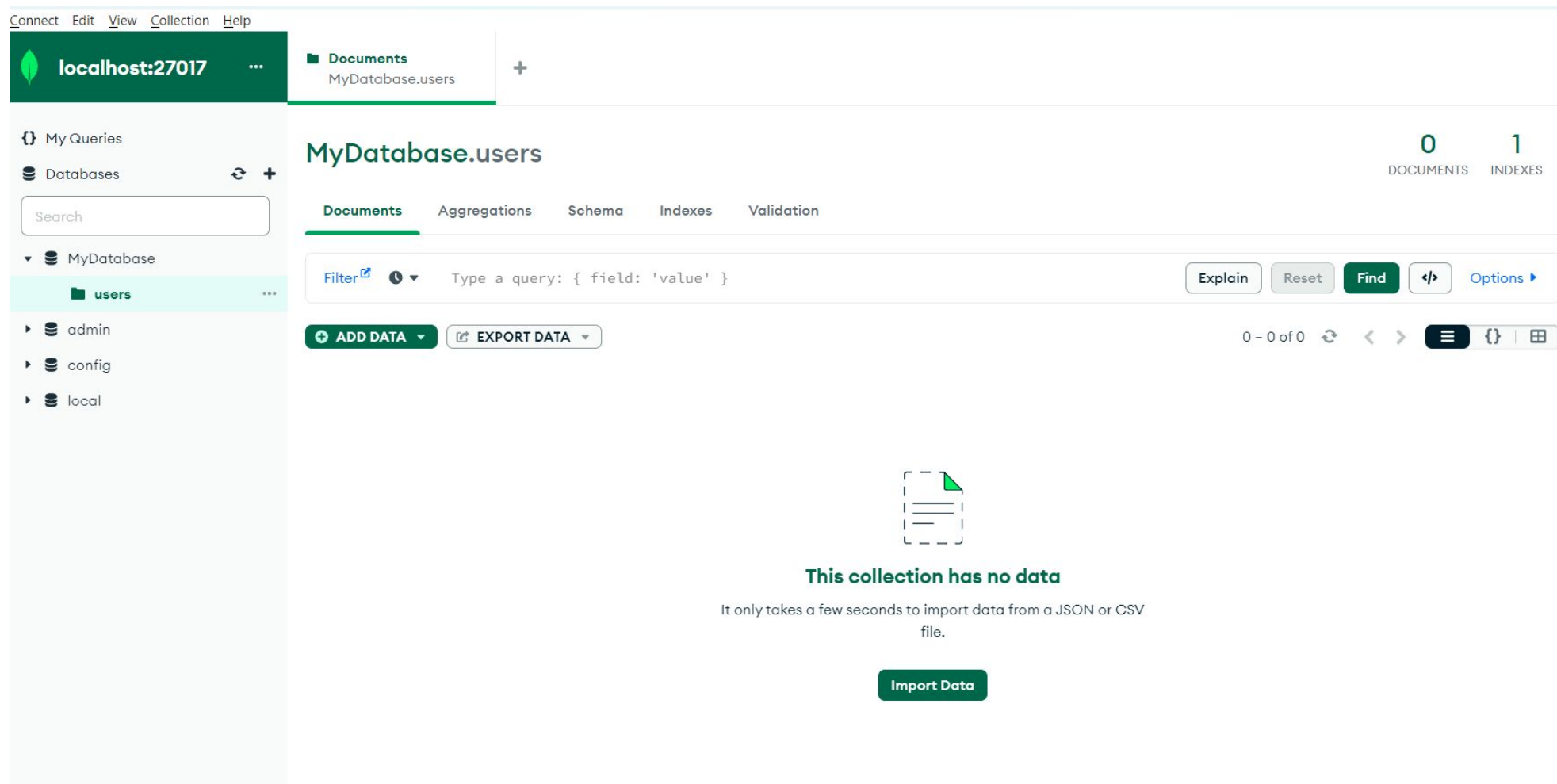
users

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

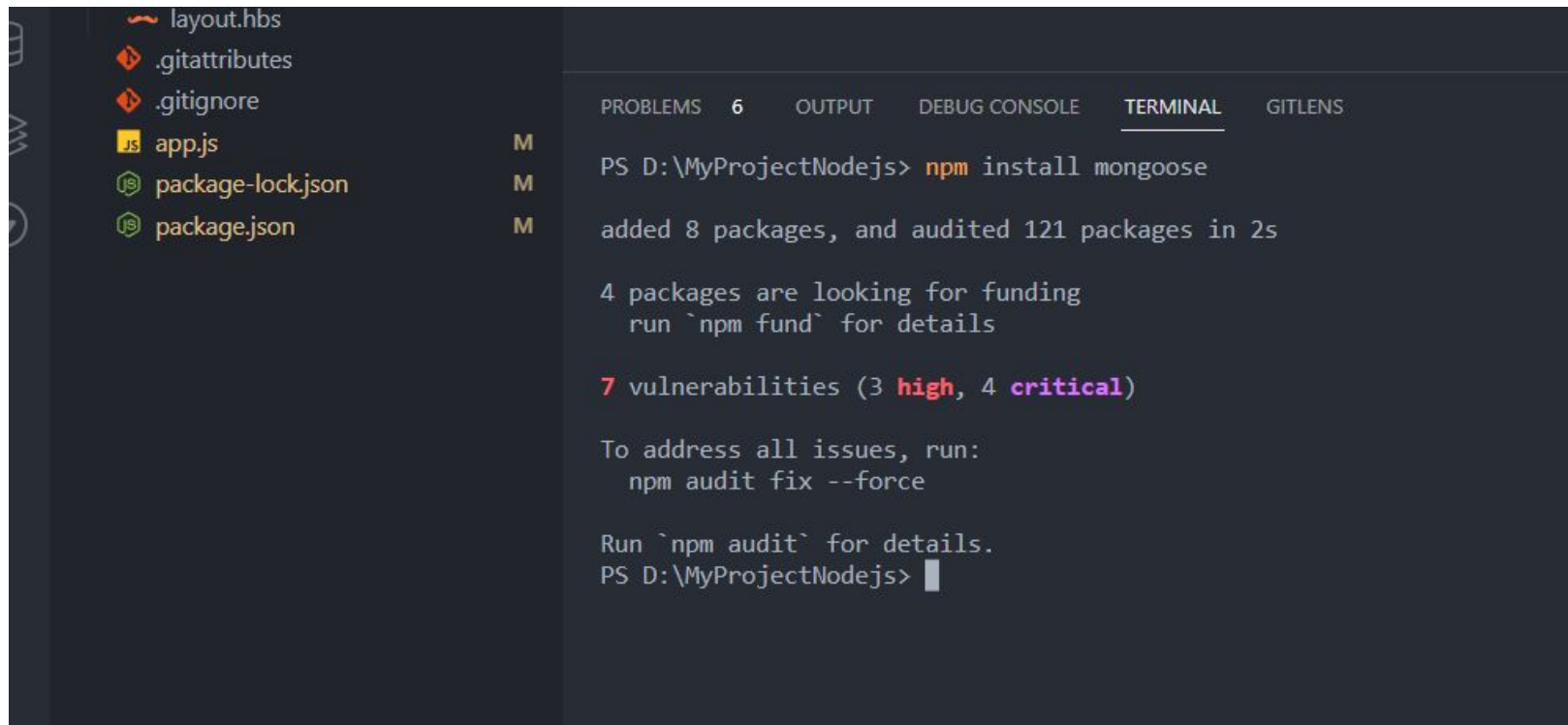
> Additional preferences (e.g. Custom collation, Capped, Clustered collections)

Cancel **Create Database**

❑ Sau khi tạo thành công, sẽ có giao diện như dưới đây



- ❑ Mở folder chứa dự án Nodejs, mở Terminal và gõ lệnh **npm i mongoose** để cài đặt thư viện **mongoose** (*thư viện này dùng để tạo kết nối dự án với database mongodb*)



```
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>
```

- ❑ Tại thư mục gốc tạo thêm folder đặt tên là **config**, tại folder **config** tạo file **db.js** và xây dựng code như hình dưới đây

```
JS db.js  X
config > JS db.js > ...
1  const mongoose = require('mongoose')
2  mongoose.set('strictQuery', true)
3  // Đối với database dùng compass
4  const local = "mongodb://127.0.0.1:27017/MyDatabase"
5  //Đối với database dùng atlas (cloud)
6  const atlas = "mongodb+srv://root:root@cluster0.fsxrzox.mongodb.net/?retryWrites=true&w=majority"
7  const connect = async () =>{
8      try {
9          await mongoose.connect(local /* Truyền biến database muốn connect*/,
10              {
11                  useNewUrlParser:true,
12                  useUnifiedTopology: true,
13              })
14          console.log(message: 'connect success')
15      } catch (error) {
16          console.log(message: error)
17          console.log(message: 'connect fail')
18      }
19  }
20  module.exports = {connect}
```

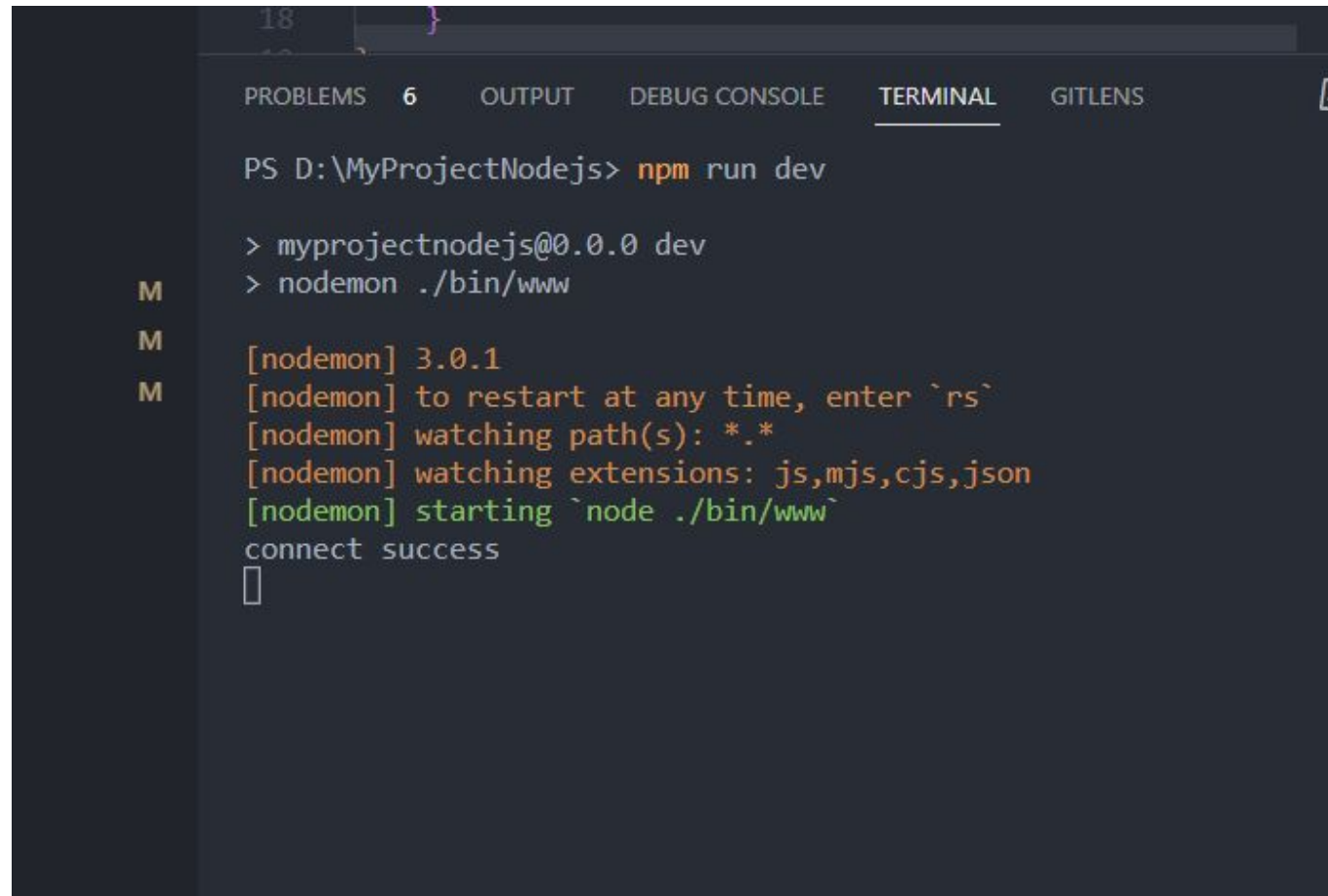
Lưu ý: **"mongodb://127.0.0.1:27017/MyDatabase"** là đường dẫn đến database trên database trên mongodb compass

- Tại **app.js** gọi file **db.js** để kết nối đến database



```
9
10
11 const database = require(id: './config/db');
12 var app = express();
13
14 // view engine setup
15 app.set(setting: 'views', val: path.join(...paths: __dirname, 'views'));
16 app.set(setting: 'view engine', val: 'hbs');
17
18 app.use(...handlers: logger(format: 'dev'));
19 app.use(...handlers: express.json());
20 app.use(...handlers: express.urlencoded(options: { extended: false }));
21 app.use(...handlers: cookieParser());
22 app.use(...handlers: express.static(root: path.join(...paths: __dirname, 'public')));
23
24 //Routes
25 app.use(path: '/', ...handlers: indexRouter);
26
27 database.connect();
28
```

- ❑ Chạy dự án nếu thấy log **connect success** là đã kết nối 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  
█
```



CÁC PHƯƠNG THỨC TRUY VẤN

...

❑ **find()** dùng để lấy dữ liệu từ collection, tùy chỉnh của phương thức này:

❖ Lấy hết document trong collection:

find({})

❖ Chọn document cần hiển thị:

find({<tên field cần hiển thị>: 1})

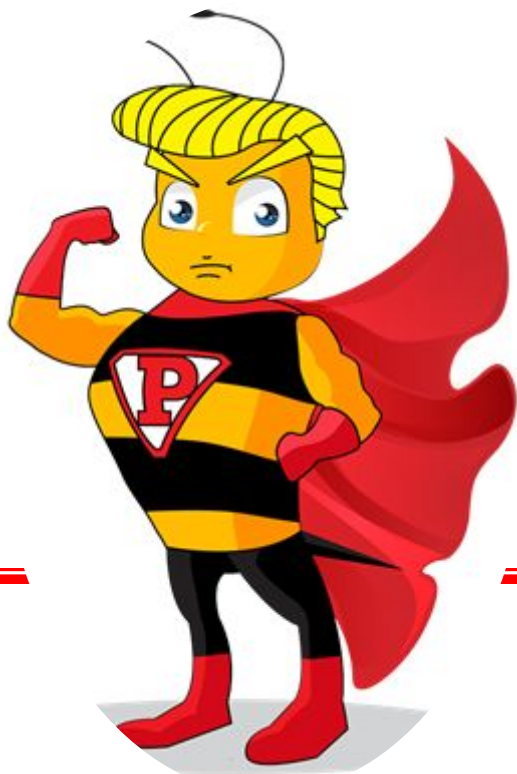
**Lưu ý: giá trị 1 để hiển thị, 0 để ẩn*

❖ Lấy field theo điều kiện:

find({<tên field cần lấy>: "Điều kiện"})

- ❑ **findById():** dùng để lấy các document theo `_id`
- ❑ **findByIdAndUpdate():** dùng để tìm document theo `_id` và update collection đó
- ❑ **findOne():** dùng để tìm kiếm 1 document
- ❑ **sort():** dùng để lấy các document thoả mãn điều kiện, thường kết hợp với phương thức **find()**
- ❑ **deleteOne():** dùng để xóa 1 document



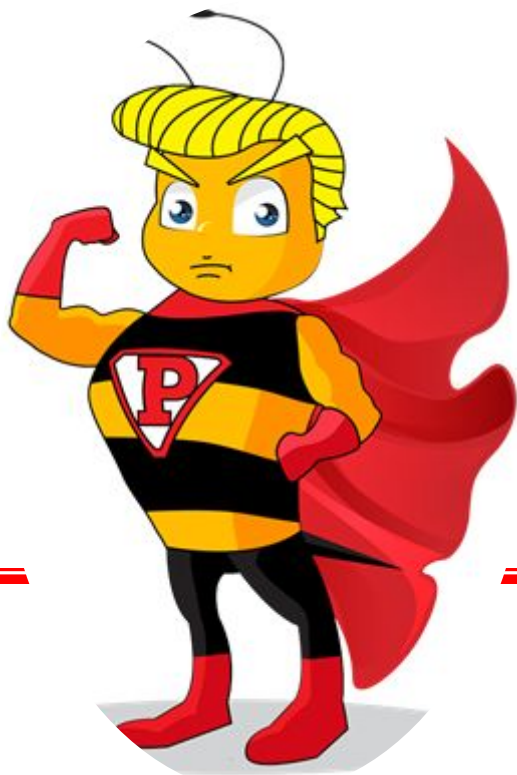


CÁC PHƯƠNG THỨC API

...

- ❑ **get:** dùng để lấy dữ liệu
- ❑ **post:** dùng để đẩy dữ liệu
- ❑ **put:** dùng để update dữ liệu
- ❑ **delete:** dùng để xóa dữ liệu
- ❑ Các phương thức trên có thể dùng có params hoặc không cần **params** (ngoại trừ mục đích update dữ liệu)





VIẾT API CƠ BẢN

...

- ❑ Tại thư mục gốc, tạo folder **models**, trong folder này tạo file **distributors.js**

```
JS distributors.js ×
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)
```

*Lưu ý: tên collection để dạng số ít, vì khi tạo collection bằng code thì khi đẩy lên database để tự động là tên collection ở dạng số nhiều.

Ví dụ: trong code: **distributor** khi đẩy lên database sẽ là **distributors**

❑ Trong folder **models**, tiếp tục tạo file **fruits.js**

```
JS fruits.js 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 */
```

- ❑ Trong thư mục **routes**, tạo file **api.js** và import 2 **model** vừa tạo

```
JS api.js M X
routes > JS api.js > ...
1
2 var express = require(id: 'express');
3 var router = express.Router();
4
5 //Thêm model
6 const Distributors = require(id: '../models/distributors')
7 const Fruits = require(id: '../models/fruits')
8
9 module.exports = router;
10
11
12 |
```

Viết api để tạo 1 document trong collection **distributors**

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

❑ Viết api để tạo 1 document trong collection **fruits**

```
//Api thêm fruit
router.post(path: '/add-fruit',...handlers: 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(body: {
        "status" : 200,
        "messenger" : "Thêm thành công",
        "data" : result
      })
    }
    }else
    {
      // Nếu thêm không thành công result null, thông báo không thành công
      res.json(body: {
        "status" : 400 ,
        "messenger" : "Lỗi, thêm không thành công",
        "data" : []
      })
    }
  }
} catch (error) {
  console.log(message: error);
}
});
```

- ❑ Viết api để lấy danh sách document trong collection **fruits**

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


Viết api để cập nhật 1 collection **fruits** thông qua **_id**

```
//Api cập nhật fruit
router.put(path: '/update-fruit-by-id/:id',...handlers: 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(body: {
        "status" : 200,
        "messenger" : "Cập nhật thành công",
        "data" : result })
    }else{
      // Nếu thêm không thành công result null, thông báo không thành công
      res.json(body: {
        "status" : 400 ,
        "messenger" : "Lỗi, Cập nhật không thành công",
        "data" : [] })
    }
  } catch (error) {
    console.log(message: error);
  }
});
```

- Tiếp theo mở file **app.js**, import **route** vừa tạo

```
var apiRouter = require(id: './routes/api');
```

```
app.use(path: '/api', ...handlers: apiRouter)
```

- Dùng **Postman** để kiểm tra xem api đã hoạt động đúng hay chưa.
Truy cập trang web <https://www.postman.com/downloads/> download và cài đặt phần mềm

API phương thức POST

The screenshot displays a REST client interface with the following components:

- Method:** POST (highlighted with a red box).
- URL:** http://localhost:3000/api/add-distributor (highlighted with a red box).
- Link:** link (highlighted with a yellow box).
- Body:** The request body is in JSON format, containing: `{ "name": "Công ty B" }` (highlighted with a red box).
- Response:** The response is in JSON format, showing a successful status and data: `{ "status": 200, "messenger": "Thêm thành công", "data": { "name": "Công ty B", "_id": "64e9bc4eac77b95a57450e5f", "createdAt": "2023-08-26T08:48:14.329Z", "updatedAt": "2023-08-26T08:48:14.329Z", "__v": 0 } } }` (highlighted with a red box).
- Status:** 200 OK (highlighted with a yellow box).

Other visible elements include tabs for Headers, Pre-request Script, Tests, Settings, Cookies, and Beautify. The response is also shown in a 'Pretty' view.

API phương thức GET

The screenshot displays a REST client interface with a GET request to `http://localhost:3000/api/get-list-fruit`. The response is a JSON object with a status of 200 OK, a messenger message, and a list of fruit data.

Request: GET `http://localhost:3000/api/get-list-fruit`

Query Params:

Key	Value	Description
Key	Value	Description

Response Body (JSON):

```
1 {
2   "status": 200,
3   "messenger": "Danh sách fruit",
4   "data": [
5     {
6       "_id": "64eaeab82254364fed6ffc3b",
7       "name": "Dưa Hấu Ấn Độ",
8       "quantity": 0,
9       "price": 5600,
10      "status": -1,
11      "image": [
12        "https://bizweb.dktcdn.net/100/447/072/products/tao-do-nhap-khau-my-hop-1kg-4-6-trai-202205130910025361.jpg?v=1684999008063",
```

API phương thức PUT thông qua _id

The screenshot displays a REST client interface with a PUT request configured. The URL is `http://localhost:3000/api/update-fruit-by-id/64eaeab82254364fed6ffc3b`, with the ID `64eaeab82254364fed6ffc3b` highlighted in red and labeled "truyền params". The request body is in JSON format, containing fields for name, quantity, price, description, and distributor ID. The response shows a successful status (200 OK) and a JSON object with the updated fruit details.

PUT update-fruits

androidnetwork / update-fruits

PUT `http://localhost:3000/api/update-fruit-by-id/64eaeab82254364fed6ffc3b` **truyền params** Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies Beautify

none form-data x-www-form-urlencoded raw binary GraphQL JSON

```
1 {
2   "name": "Dưa Hấu Ấn Độ",
3   "quantity": 0,
4   "price": 5600,
5   "description": "Ngũng kinh doanh trong thời gian dài",
6   "id_distributor": "64dbc822a96c55f5c6d90a1d" // Là _id của distributor ta vừa mới thêm
7 }
```

Body Cookies Headers (7) Test Results Status: 200 OK Time: 18 ms Size: 872 B Save as Example

Pretty Raw Preview Visualize JSON

```
1 {
2   "status": 200,
3   "messenger": "Cập nhật thành công",
4   "data": {
5     "_id": "64eaeab82254364fed6ffc3b",
6     "name": "Dưa Hấu Ấn Độ",
7     "quantity": 0,
8     "price": 5600,
9     "status": -1,
10    "image": [
11      "https://bizweb.dktcdn.net/100/447/072/products/tao-do-nhap-khau-my-hop-1kg-4-6-trai-202205130910025361.jpg?v=1684999008063",
12      "https://admin.nongsandungha.com/wp-content/uploads/2021/06/tao-my-huu-co-00-min.png",
13    ]
14  }
15 }
```

DEMO



FPT Education

FPT POLYTECHNIC

Thank you