

# Lab 1: Setting Up and Configuring the Development Environment for Building a Website

MSc. Nguyen Thai Anh

## 1 Objectives

- Set up the Flask application development environment.
- Install the code editing environment.
- Install Python and Flask.
- Connect to MongoDB Atlas (Cloud Database).
- Learn how to upload and download data from GitHub.
- Understand routing and data handling in Flask.

## 2 Setting Up the Code Editing Environment

### 2.1 Installing VSCode

1. Download the Visual Studio Code installer for Windows at: <https://code.visualstudio.com/>.
2. Run the installer, named something like `VSCodeUserSetup-{version}.exe`.
3. Click **Next** to proceed and agree to the terms of use.
4. Choose the installation location (recommended to keep the default, C drive), then click **Next**.
5. Continue clicking **Next** until completion. It is recommended to check:
  - Add “Open with Code” action to Windows Explorer file context menu.
  - Add “Open with Code” action to Windows Explorer directory context menu.

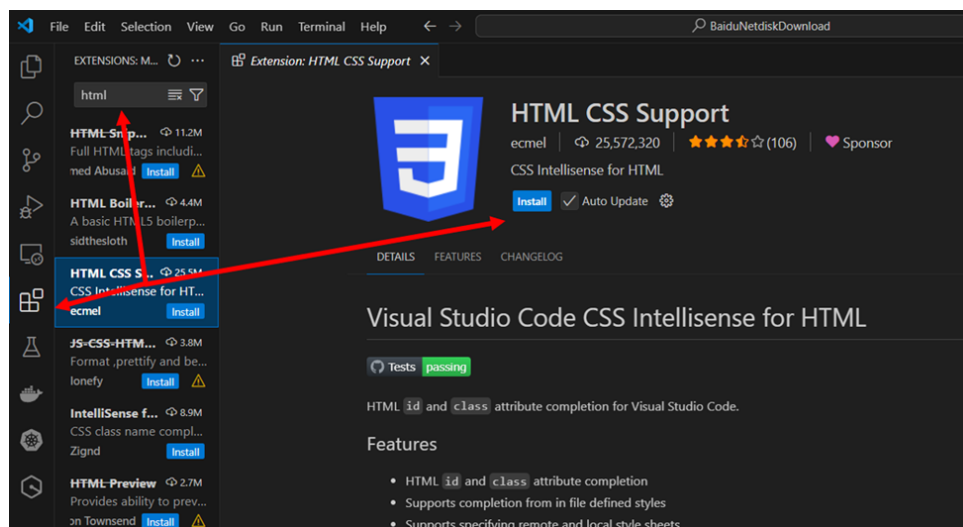
6. Installation complete, explore Visual Studio Code.

## 2.2 Extensions to Support Programming and Libraries

Recommended extensions:

- **Python:** Install Python (if not already installed) at: <https://www.python.org/downloads/>. Select the “Add Python to PATH” option during installation.
- **HTML CSS Support:** Supports writing HTML/CSS code.

Installation method: Press **Ctrl + Shift + X** or go to the Extensions icon in VS Code, search for the extension, and click **Install**.

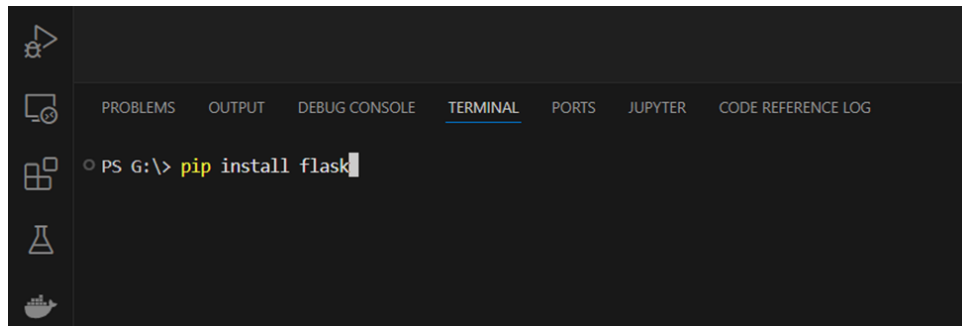


## 2.3 Required Libraries

- **flask:** Main web framework.
- **pymongo:** Library for connecting to MongoDB.
- **dnspython:** Supports Atlas connection (DNS URI).
- **python-dotenv:** Supports reading environment variables from a `.env` file.

Installation method: Open Terminal in VS Code, select **New Terminal**, then run:

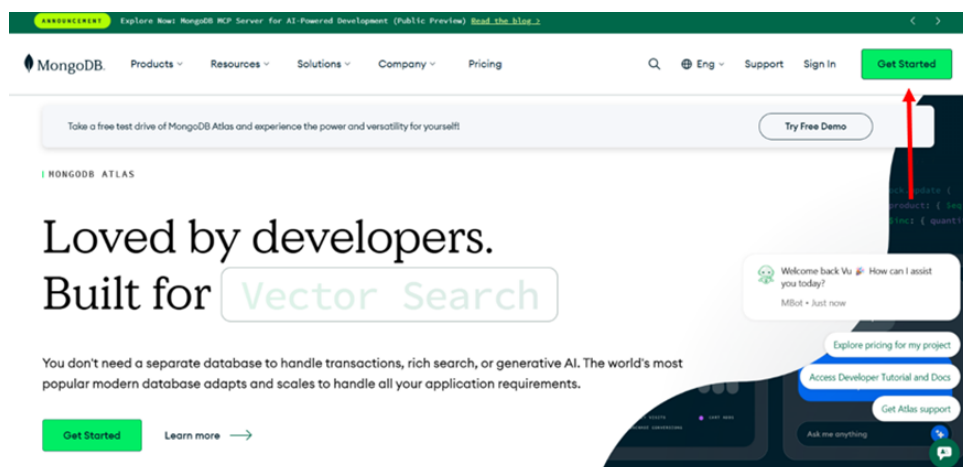
```
1 pip install flask
```



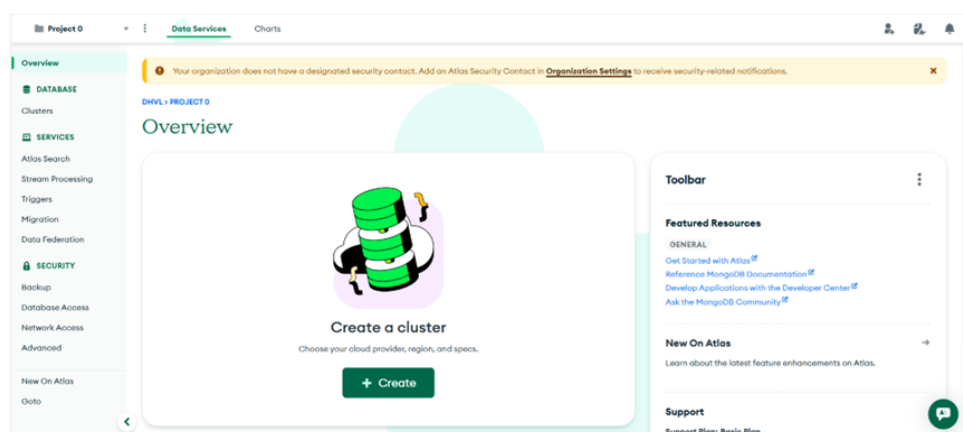
Repeat for other libraries: pymongo, dnspython, python-dotenv.

### 3 Connecting to MongoDB Atlas

1. Visit <https://www.mongodb.com/atlas> and log in.
2. Select **Get Started** to sign up using email, Google, or GitHub.



3. After signing up, access the MongoDB Atlas Dashboard.



4. Click **Create** or **Build a Database** to create a new cluster.
5. Select the **Free (Shared Cluster - M0)** plan.
6. Choose a geographic region (e.g., AWS - Singapore or Google Cloud - Tokyo).
7. Name the cluster and click **Create Deployment**.

**Deploy your cluster**

Use a template below or set up advanced configuration options. You can also edit these configuration options once the cluster is created.

☐ **M10** **\$0.09/hour**  
Dedicated cluster for development environments and low-traffic applications.  

STORAGE	RAM	vCPU
10 GB	2 GB	2 vCPUs

☐ **Flex** **From \$0.011/hour**  
Up to \$30/month  
For application development and testing, with on-demand burst capacity for unpredictable traffic.  

STORAGE	RAM	vCPU
5 GB	Shared	Shared

☒ **Free**  
For learning and exploring MongoDB in a cloud environment.  

STORAGE	RAM	vCPU
512 MB	Shared	Shared

Free forever! Your free cluster is ideal for experimenting in a limited sandbox. You can upgrade to a production cluster anytime.

**Configurations**  
Name  
You cannot change the name once the cluster is created.  
  
Provider  
Region  

Hong Kong (ap-east-1)

★ Recommended ⓘ Low carbon emissions ⓘ

**Quick setup**  
☒ Automate security setup ⓘ  
☒ Preload sample dataset ⓘ

### 3.1 Creating a User and Connecting to the Cluster

- Modify the username and password, then select **Create Database User**.

1  
Set up connection security

2  
Choose a connection method

3  
Connect

You need to secure your MongoDB Atlas cluster before you can use it. Set which users and IP addresses can access your cluster now. [Read more](#)

- Add a connection IP address**

✓ Your current IP address (115.79.138.76) has been added to enable local connectivity. Only an IP address you add to your Access List will be able to connect to your project's clusters. Add more later in [Network Access](#).
- Create a database user**

This first user will have [atlasAdmin](#) permissions for this project.

We autogenerated a username and password. You can use this or create your own.

ⓘ You'll need your database user's credentials in the next step. Copy the database user password.


Username

Password


- Select **Method Drivers**.


Progress: 1. Set up connection security (checked), 2. Choose a connection method (active), 3. Connect


### Connect to your application


 **Drivers**  
Access your Atlas data using MongoDB's native drivers (e.g. Node.js, Go, etc.)

### Access your data through tools

 **Compass**  
Explore, modify, and visualize your data with MongoDB's GUI

 **Shell**  
Quickly add & update data using MongoDB's Javascript command-line interface

 **MongoDB for VS Code**  
Work with your data in MongoDB directly from your VS Code environment

 **Atlas SQL**  
Easily connect SQL tools to Atlas for data analysis and visualization

[Go Back](#) [Close](#)

- At this step, save the connection string below for later use to connect the website to MongoDB.

Progress: 1. Set up connection security (checked), 2. Choose a connection method (checked), 3. Connect (active)

### Connecting with MongoDB Driver

#### 1. Select your driver and version

We recommend installing and using the latest driver version.

Driver	Version
Node.js	6.7 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

Use this connection string in your application

☐ View full code sample ☒ Show Password ⓘ

`mongodb+srv://dhvl:ImvRyWtu5pY5Luq@vv.vlw55be.mongodb.net/?retryWrites=true&w=majority&appName=vv`

The password for **dhvl** is included in the connection string for your first time setup. This password will not be available again after exiting this connect flow.

#### RESOURCES

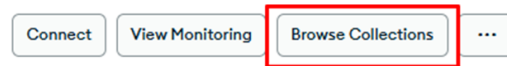
[Get started with the Node.js Driver](#) [Node.js Starter Sample App](#)  
[Access your Database Users](#) [Troubleshoot Connections](#)

[Go Back](#) [Done](#)

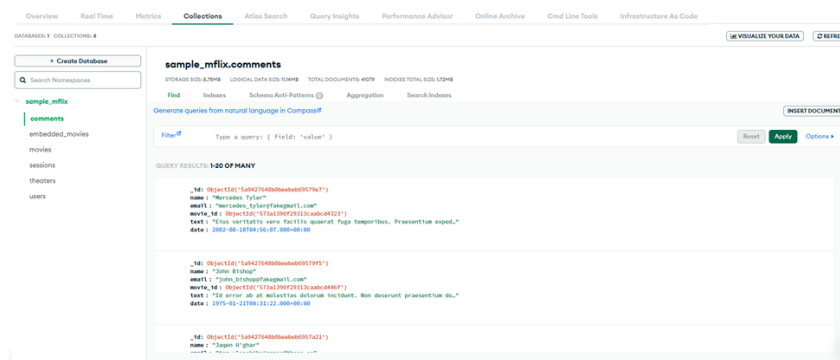
- Click **Done**.

## 3.2 Creating a Database

1. In the Dashboard, select **Browse Collections**.



2. Click **Create Database**.
3. Enter:
  - **Database Name:** For example, myDatabase.
  - **Collection Name:** For example, users.
4. Click **Create**.



### Create Database

Database name ?

Collection name ?

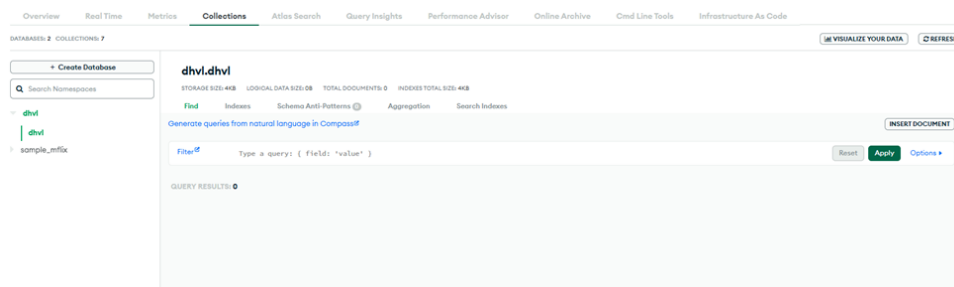
Additional Preferences

Select

Cancel

Create

5. After successfully creating the database, students can input data into the database here:



## 4 Uploading and Downloading Data from GitHub

### 4.1 Downloading Data from GitHub

1. Go to GitHub, open the desired repository, click the **Code** button, and copy the HTTPS link (e.g., <https://github.com/username/repository.git>).
2. Open Terminal or Git Bash and run:

```
1 git clone https://github.com/username/repository.git
```

```
● PS D:\> git clone https://github.com/vuviet1207/vidu.git
Cloning into 'vidu'...
```

3. The folder will be downloaded to your machine.

### 4.2 Uploading a Project to GitHub

1. Create a new repository on GitHub: Click **New Repository**, name it, and click **Create Repository**.
2. In the terminal, navigate to the project folder and initialize Git:

```
1 cd /path/to/project_folder
2 git init
```

```
PS D:\> cd bt
PS D:\bt> git init
Initialized empty Git repository in D:/bt/.git/
```

3. Connect to the repository:

```
1 git remote add origin https://github.com/username/  
    repository.git
```

```
PS D:\bt> git remote add origin https://github.com/vuviet1207/vidu.git  
>>
```

4. Add, commit, and push:

```
1 git add .  
2 git commit -m "First commit"  
3 git push -u origin master
```

5. The folder or files will be uploaded to the specified GitHub repository.

## 5 Developing Your First Flask Application

1. Create a folder named Lab1.
2. Open VS Code, select File/Add Folder, and choose the Lab1 folder.
3. Right-click the Lab1 folder in VS Code, select New File, and create the following structure:

```
Lab1/  
  app.py  
  templates/  
    hello.html
```

4. Open app.py to write the application.

```
from flask import Flask, render_template  
  
app = Flask(__name__)  
  
@app.route('/')  
def hello():  
    return render_template('hello.html')  
  
if __name__ == '__main__':  
    app.run(debug=True)
```



5. Open `hello.html` to add HTML content.

```
<!DOCTYPE html>
<html>
<head>
|   <title>Hello</title>
</head>
<body>
|   <h1>Hello, World!</h1>
</body>
</html>
```

6. Run `app.py`; the terminal will display a link (e.g., `http://127.0.0.1:5000`).

```
PS D:\> & C:/Python313/python.exe d:/Lab1/app.py
* Serving Flask app 'app'
* Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
* Restarting with watchdog (windowsapi)
* Debugger is active!
* Debugger PIN: 541-036-351
```

7. Click the link to access the website.



## 6 Practical Exercises

### 6.1 Exercise 1: Create a Website Displaying a Linear Equation

Create a website that displays the linear equation:  $y = 5x + 7$  with  $x = -2$ .

**Requirements:**

- Display the equation on the web interface.
- Calculate and display the value of  $y$ .
- Style the interface simply using CSS.

### 6.2 Exercise 2: Create a Website Displaying Student Information

Create a website that displays a student's personal information, including:

- Name.
- Student ID.
- Academic Year.
- Major.
- Hobbies.

**Requirements:**

- Present the information neatly using basic HTML tags.
- Use either `<table>` or `<ul>` to display the information.

### 6.3 Exercise 3: Develop a Website Using Flask Following the Provided Code

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4    <meta charset="UTF-8" />
5    <title>Sample Form</title>
6  </head>
7  <body>
8
9    <h2>User Information Form</h2>
10
11    <form action="#" method="post">
12
13      <label for="username">Username:</label><br>
14      <input type="text" id="username" name="username" /><br><br>
15
16      <label for="password">Password:</label><br>
17      <input type="password" id="password" name="password" /><br><br>
18
19      Gender:<br>
20      <input type="radio" id="male" name="gender" value="male" />
21      <label for="male">Male</label><br>
22
23      <input type="radio" id="female" name="gender" value="female" />
24      <label for="female">Female</label><br>
25
26      <input type="radio" id="other" name="gender" value="other" />
27      <label for="other">Other</label><br><br>
28
29      <label for="membership">Membership Level:</label><br>
30      <input list="Options" id="membership" name="membership" />
31      <datalist id="Options">
32        <option value="Thành viên Bạc"></option>
33        <option value="Thành viên Vàng"></option>
34        <option value="Thành viên Kim Cương"></option>
35      </datalist><br><br>
36
37      <label for="favcolor">Choose your favorite color:</label><br>
38      <input type="color" id="favcolor" name="favcolor" /><br><br>
39
40      <input type="checkbox" id="correct" name="correct" value="correct" />
41      <label for="correct">Correct</label><br><br>
42
43      <input type="submit" value="Submit" />
44
45    </form>
46
47  </body>
48 </html>
49
```

#### Requirements:

- Create a Flask website based on the provided code structure (shown in the image above).
- Modify the "User Information Form" in the code to "Student Information Form," personalized with the student's name (e.g., "Viet Vu Information Form").
- Use appropriate HTML tags to structure the content as specified in the code.
- Apply CSS for styling to match the design outlined in the provided code.

## 6.4 General Requirements

- Use Flask to create the server and handle routing.
- Use HTML to build the interface.
- Use basic CSS for styling (inline or separate CSS file).
- Apply basic HTML tags such as: `<h1>`, `<p>`, `<div>`, `<span>`, `<ul>`, `<li>`, `<table>`, etc.
- Folder structure:

```
Lab1/  
  app.py  
  templates/  
    index.html  
  static/  
    style.css
```

## 6.5 Submission

- After completing the exercises, combine all folders into a folder named: `LAB1_StudentID_FullName`.
- Create a `.docx` file, paste screenshots of your exercise results into it, and include this file in the `LAB1_StudentID_FullName` folder.
- Upload the folder to GitHub following the instructions in the section on uploading and downloading data from GitHub.
- Deadline 23h59 29/05/2025