Sending Data from a Flask app to MongoDB Database

Last Updated : 10 May, 2023

This article covers how we can configure a MongoDB database with a Flask app and store some data in the database after configuring it. Before directly moving to the configuration phase here is a short overview of all tools and software we will use.

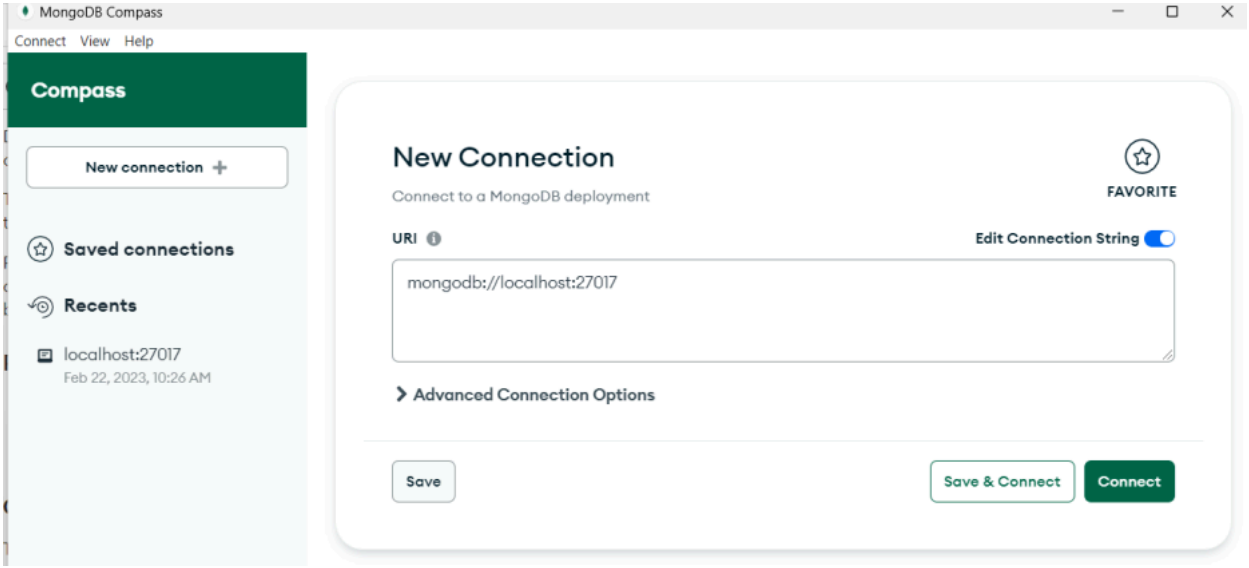
MongoDB is an open-source database that stores data in JSON-like documents. It is classified as a NoSQL database because it is based on different fundamentals as compared to a SQL relational database.

Prerequisites

- A decent understanding of Python and a machine with Python installed.
- Understanding of basic concepts of Flask.
- MongoDB is installed on your local machine if not you can refer to this article.

Configuring MongoDB

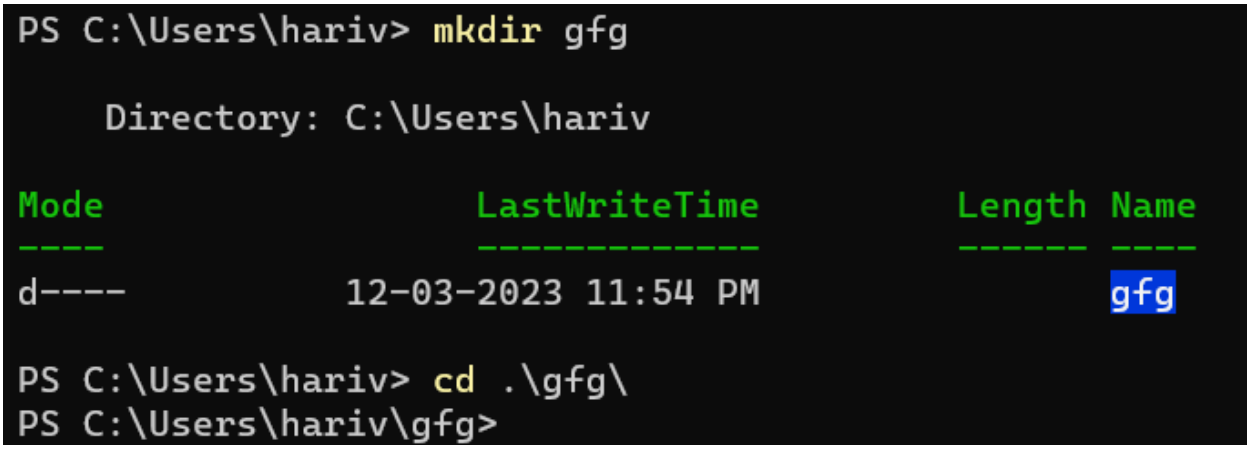
Till this step, you have a local machine with MongoDB installed on it now run the MongoDB compass and the following screen will appear. You can edit the settings or just click connect and MongoDB will be running on your local machine.



Setup a Development Environment

Let's configure the virtual environment for development, this step can be skipped but it is always recommended to use a dedicated development environment for each project to avoid dependency clash, this can be achieved using a Python virtual environment.

```
# Create gfg folder
$ mkdir gfg

# Move to gfg folder
$ cd gfg
```

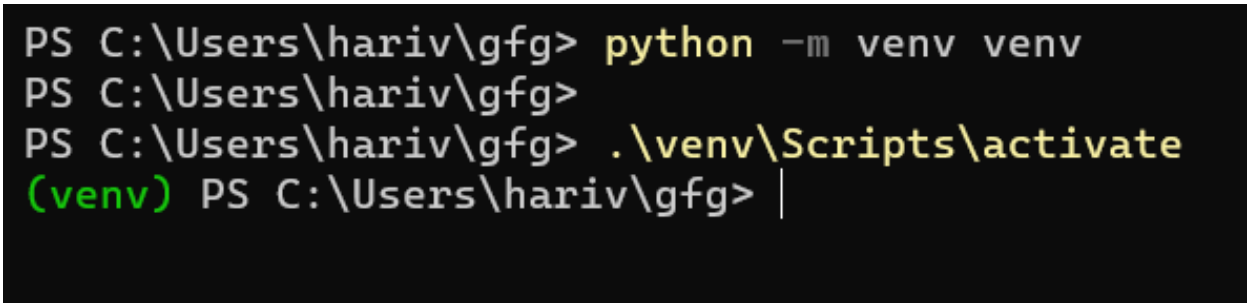


We created a folder named `gfg` for the project, you can name it anything you want and cd (change directory) to go into your newly created directory then run the following command that will create a virtual environment for your project.

```
$ python -m venv venv
```

Now to use the virtual environment we need to first activate it, this can be done by executing the activated binary file.

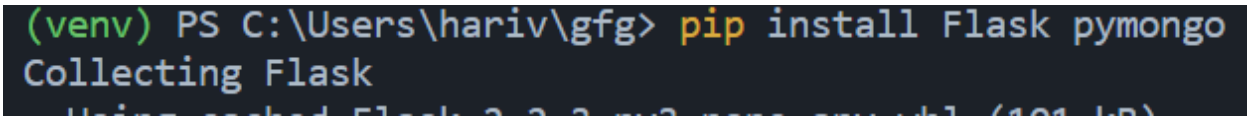
```
$ .\venv\Scripts\activate # for Windows OS
$ source venv/bin/activate # for Linux OS
```



Installing Dependencies for the Project

We are done configuring a development environment now let us install the tools that we will be using including Flask, `pymongo` which provide the interface for Python-based apps to the MongoDB database.

```
$ pip install Flask pymongo
```



Next, we'll connect a FLask app to MongoDB and send some data into it.

Creating a Flask App

We are done with the database setup and installing the required libraries now we will create a Flask app and connect it to the MongoDB we created and

insert some user data into it. First, let's create a Flask app as follows.

Create a `main.py` file in the project directory and add the following code to the file.

Python3

Flask Templates Jinja2 Flask-REST API Python SQLAlchemy Flask Bcrypt Flask Cookies Json Postman

```
app = Flask(__name__)


@app.route('/')
def hello_world():
    return 'Hello, World!'


if __name__ == '__main__':
    app.run()
```

Over here we have created a starter Flask App with a single route that returns a string "Hello, World!", Now test this thing out by running the app as shown below:

Output:

```
(venv) PS C:\Users\hariv\gfg> python main.py
 * Serving Flask app 'main'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
```

The app is up and running now let us test the output at `http://127.0.0.1:5000`,

Hello, World!

Connecting Flask App to Database

We have got a running starter Flask App with some basic code let's connect it to the MongoDB database using the following script.

Python3

```
from flask import Flask, request
from pymongo import MongoClient

# Flask app object
app = Flask(__name__)

# Set up MongoDB connection
client = MongoClient('mongodb://localhost:27017/')
db = client['demo']
collection = db['data']
```

The above script will connect to the MongoDB database that we configured earlier now we need a post route to add some data in the database which can be done as follow:

Python3

```
@app.route('/add_data', methods=['POST'])
def add_data():
    # Get data from request
```

```
    data = request.json

    # Insert data into MongoDB
    collection.insert_one(data)

    return 'Data added to MongoDB'
```

Here we created a route named `/add_data` which when invoked with a post request reads the JSON data from the body and inserts it into the database.

For reference here is the overall code that I used for the demo.

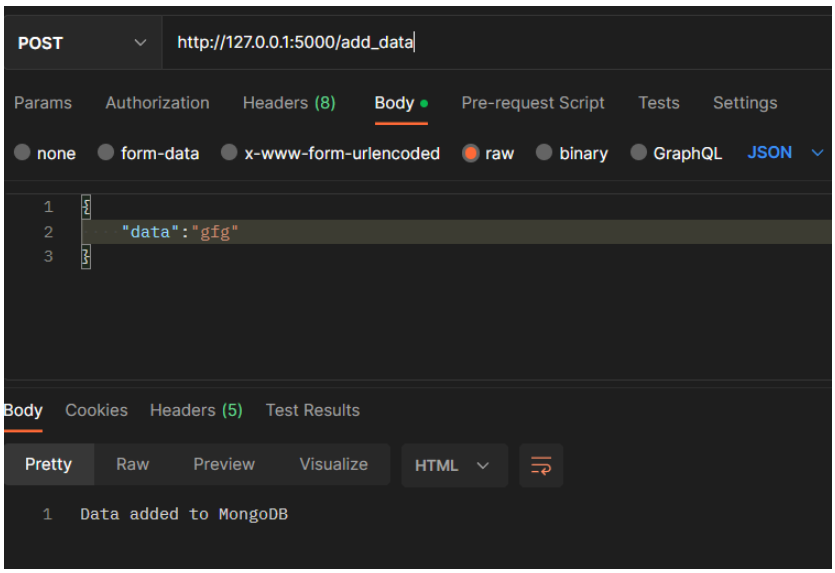
Python3

```
from flask import Flask, request
from pymongo import MongoClient

app = Flask(__name__)

# root route
@app.route('/')
def hello_world():
    return 'Hello, World!'

# Set up MongoDB connection and collection
client = MongoClient('mongodb://localhost:27017/')

# Create database named demo if they don't exist already
db = client['demo']

# Create collection named data if it doesn't exist already
collection = db['data']

# Add data to MongoDB route
@app.route('/add_data', methods=['POST'])
def add_data():
    # Get data from request
    data = request.json

    # Insert data into MongoDB
    collection.insert_one(data)

    return 'Data added to MongoDB'


if __name__ == '__main__':
```

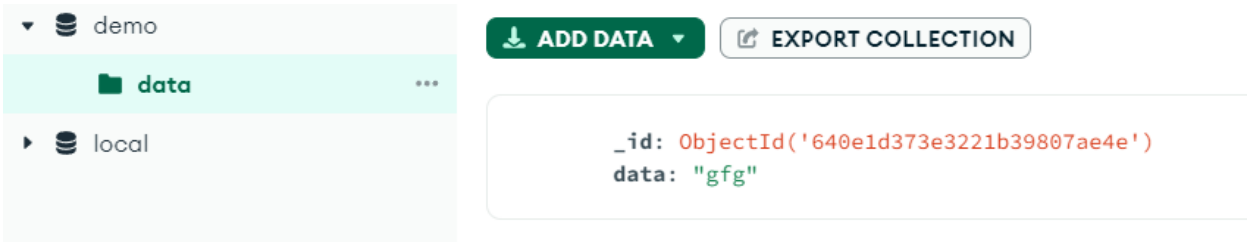
```
app.run()
```

Sending Data from Flask to MongoDB

Now let us test the entire script and if we can insert some data into the database using it, First run the Flask App shown before then make a POST request to `/add_data` a route using a tool like Postman.



The response above looks fine let us check the MongoDB database if there is any data inserted or not.



As you can see a database named demo is created with a collection of `data` with a single document that we just inserted using Flask.

Looking to dive into the world of programming or sharpen your Python skills? Our Master Python: Complete Beginner to Advanced Course is your ultimate guide to becoming proficient in Python. This course covers everything you need

to build a solid foundation from fundamental programming concepts to advanced techniques. With **hands-on projects**, real-world examples, and expert guidance, you'll gain the confidence to tackle complex **coding challenges**. Whether you're starting from scratch or aiming to enhance your skills, this course is the perfect fit. Enroll now and master Python, the language of the future!

Previous Article

How to Build a Web App using Flask and SQLite in Python

Next Article

Making a Flask app using a PostgreSQL database

Similar Reads

Sending data from a Flask app to PostgreSQL Database

A database is used to store and maintain persistent data that can be retrieved and manipulated efficiently. we usually need a database, an organized collection...

6 min read

Sending Emails Using API in Flask-Mail

Python, being a powerful language don't need any external library to import and offers a native library to send emails- "SMTP lib". "smtplib" creates a Simple Mai...

3 min read

Making a Flask app using a PostgreSQL database

The Postgres database can be accessed via one of two methods in Python. Installing PgAdmin4 is the first step because it offers a user interface for...

4 min read

Documenting Flask Endpoint using Flask-Autodoc

Documentation of endpoints is an essential task in web development and being able to apply it in different frameworks is always a utility. This article discusses...

4 min read

How to use Flask-Session in Python Flask ?

Flask Session - Flask-Session is an extension for Flask that supports Server-side Session to your application.The Session is the time between the client logs in to...

4 min read

How to Integrate Flask-Admin and Flask-Login

In order to merge the admin and login pages, we can utilize a short form or any other login method that only requires the username and password. This is know...

8 min read

Minify HTML in Flask using Flask-Minify

Flask offers HTML rendering as output, it is usually desired that the output HTML should be concise and it serves the purpose as well. In this article, we would...

12 min read

Flask URL Helper Function - Flask url_for()

In this article, we are going to learn about the flask url_for() function of the flask URL helper in Python. Flask is a straightforward, speedy, scalable library, used f...

11 min read

A Guide to Sending Data Using Cache in Django

In this article, we aim to comprehensively explore the functionality of sending data using cache in Django by providing a well-structured example...

6 min read

How to Connect Mongodb Compass to Flask

An effective GUI tool for managing and visualizing MongoDB data is MongoDB Compass. On the other hand, a well-liked Python web framework for creating...

2 min read

Article Tags : Django Python MongoDB Python Flask

Practice Tags : python