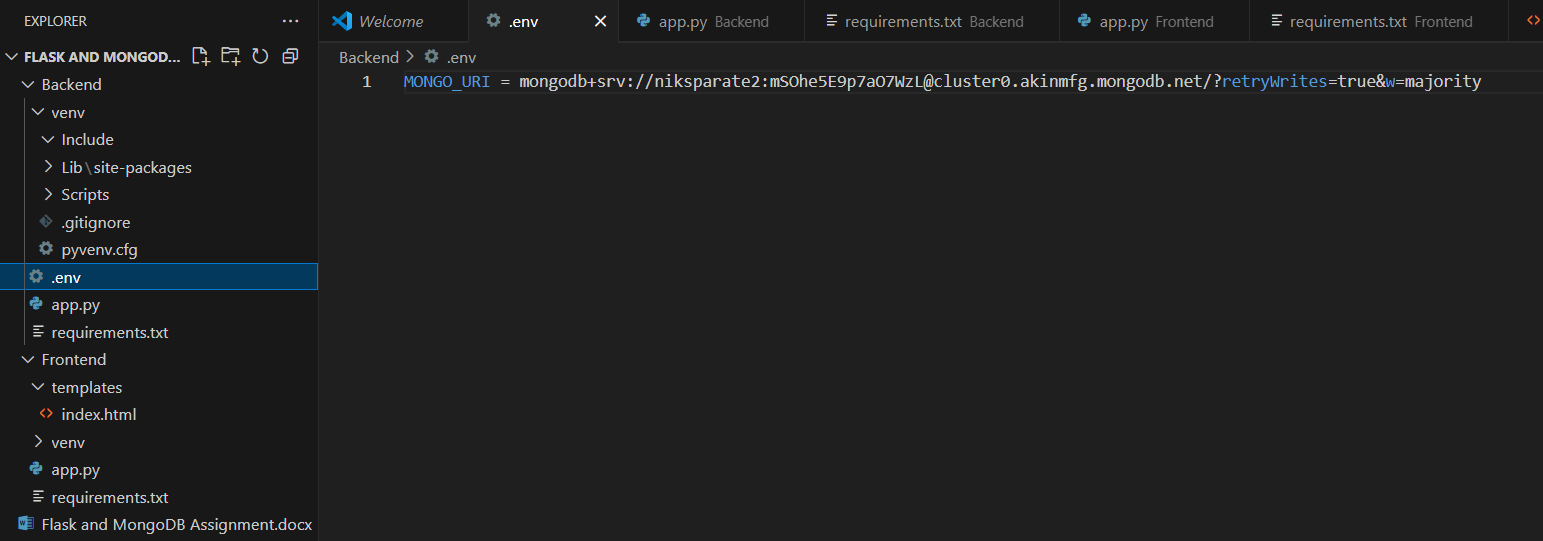
**Flask and MongoDB Assignment**

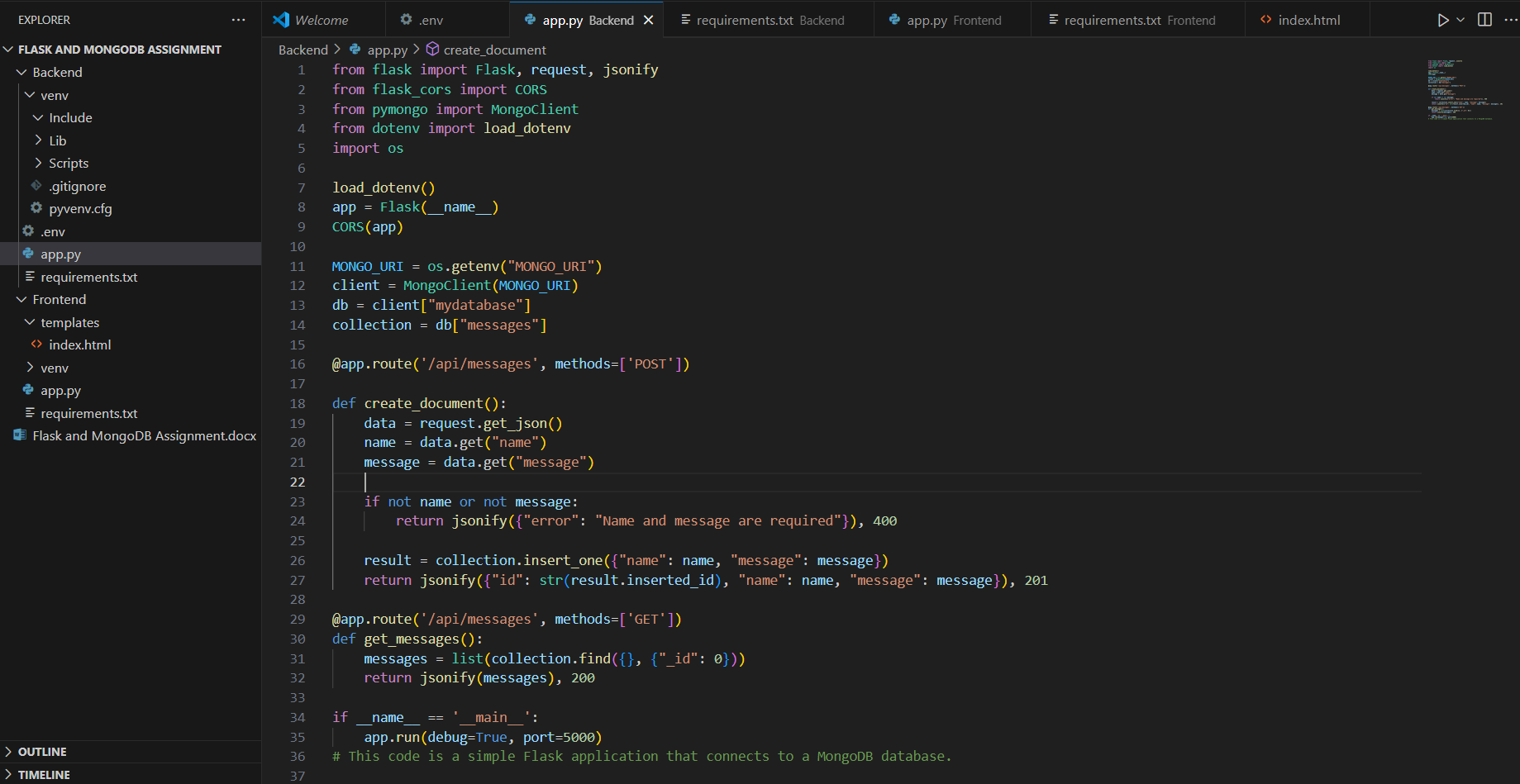
* **Backend:**

1. **.env  
     
   **

**Explanation:** .env file is used to keep sensitive data like DB passwords and API keys out of code.

* **MONGO\_URI** - This is the name of the environment variable (we access it using os.getenv("MONGO\_URI")).
* **mongodb+srv://** - Protocol used to connect to a MongoDB Atlas SRV (DNS-based) **cluster**.
* **@cluster0.akinmfg.mongodb.net/** - MongoDB cluster address — cluster0 is the cluster name, and akinmfg is project ID.
* **?retryWrites=true** - Enables automatic retries of write operations that fail due to network errors.
* **&w=majority** - Ensures write operations are acknowledged by the majority of replica set members.

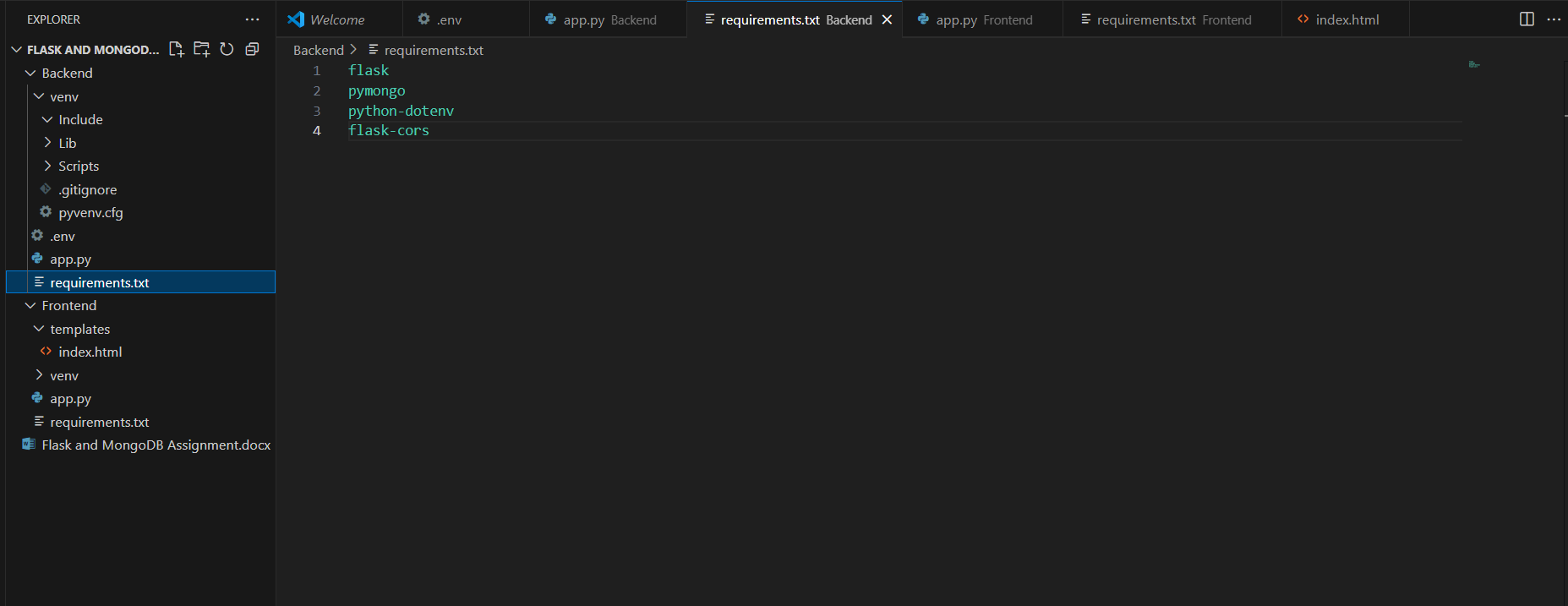
1. **app.py**

****

**Explanation:**

* **Flask**: Web framework.
* **request**: To get data from HTTP requests (e.g., POST).
* **jsonify**: To return JSON responses.
* **CORS**: Allows frontend (on a different port/domain) to communicate with backend.
* **from pymongo import MongoClient**: Connects to MongoDB using Python.
* Loads environment variables from a .env file.
* os.getenv("MONGO\_URI") fetches MongoDB URI securely.
* **MONGO\_URI**: Gets connection string from .env.
* **MongoClient(...):** Connects to your MongoDB cluster.
* **db["mydatabase"]**: Uses or creates a database named mydatabase.
* **collection = db["messages"]:** Uses or creates a collection named messages.
* **POST /api/messages**: Adds a new message to MongoDB
* **GET /api/messages**: Retrieves all messages from MongoDB

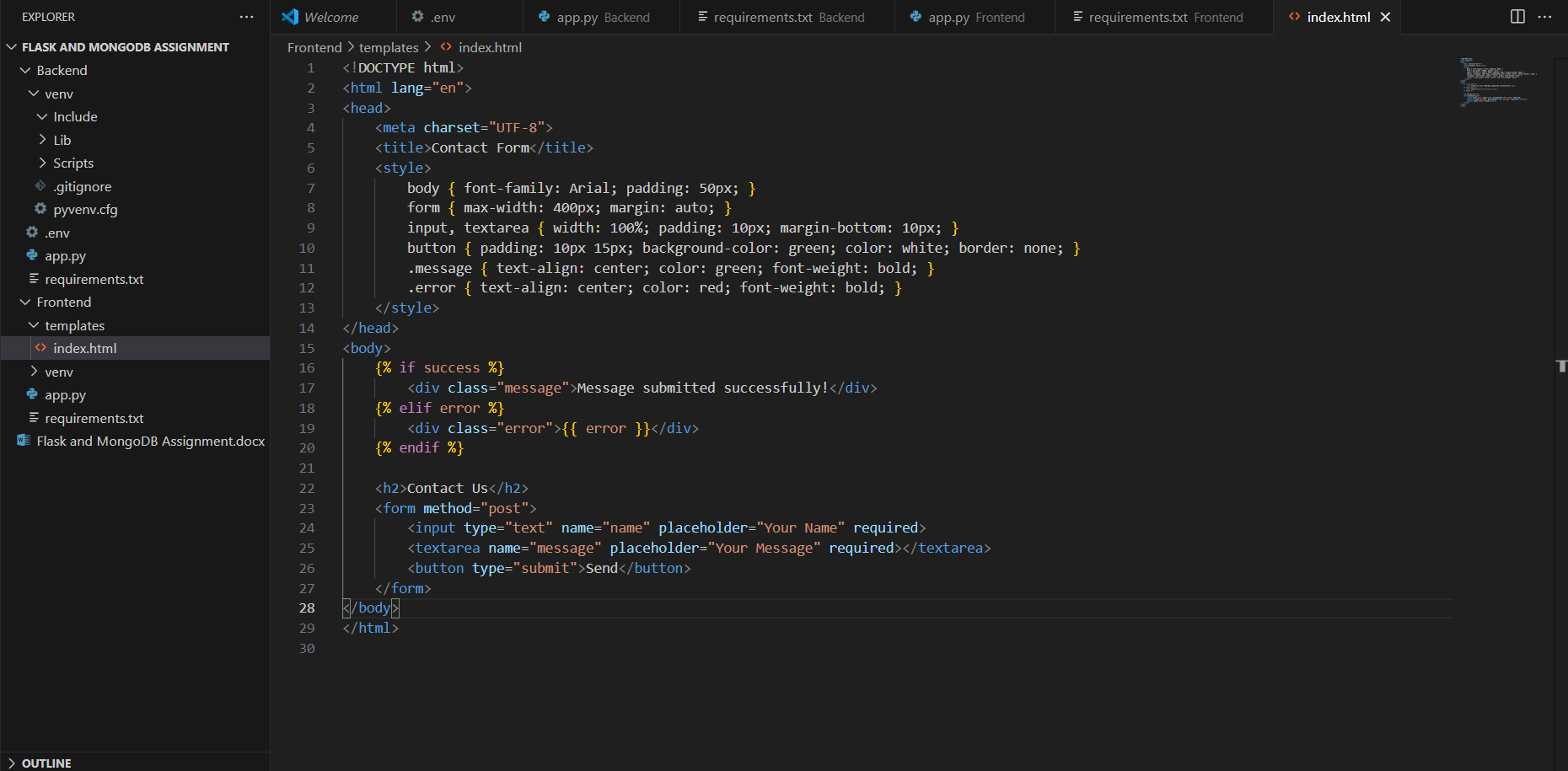
1. **requirements.txt**

****

**Explanation:**

* This file lists all the Python packages your project needs.
* **flask**: A lightweight web framework for building web applications and APIs.
* **pymongo**: Official MongoDB driver for Python.
* **python-dotenv**: Loads environment variables from a .env file into your Python app.
* **flask-cors**: A Flask extension that adds Cross-Origin Resource Sharing (CORS) support.
* **Frontend:**

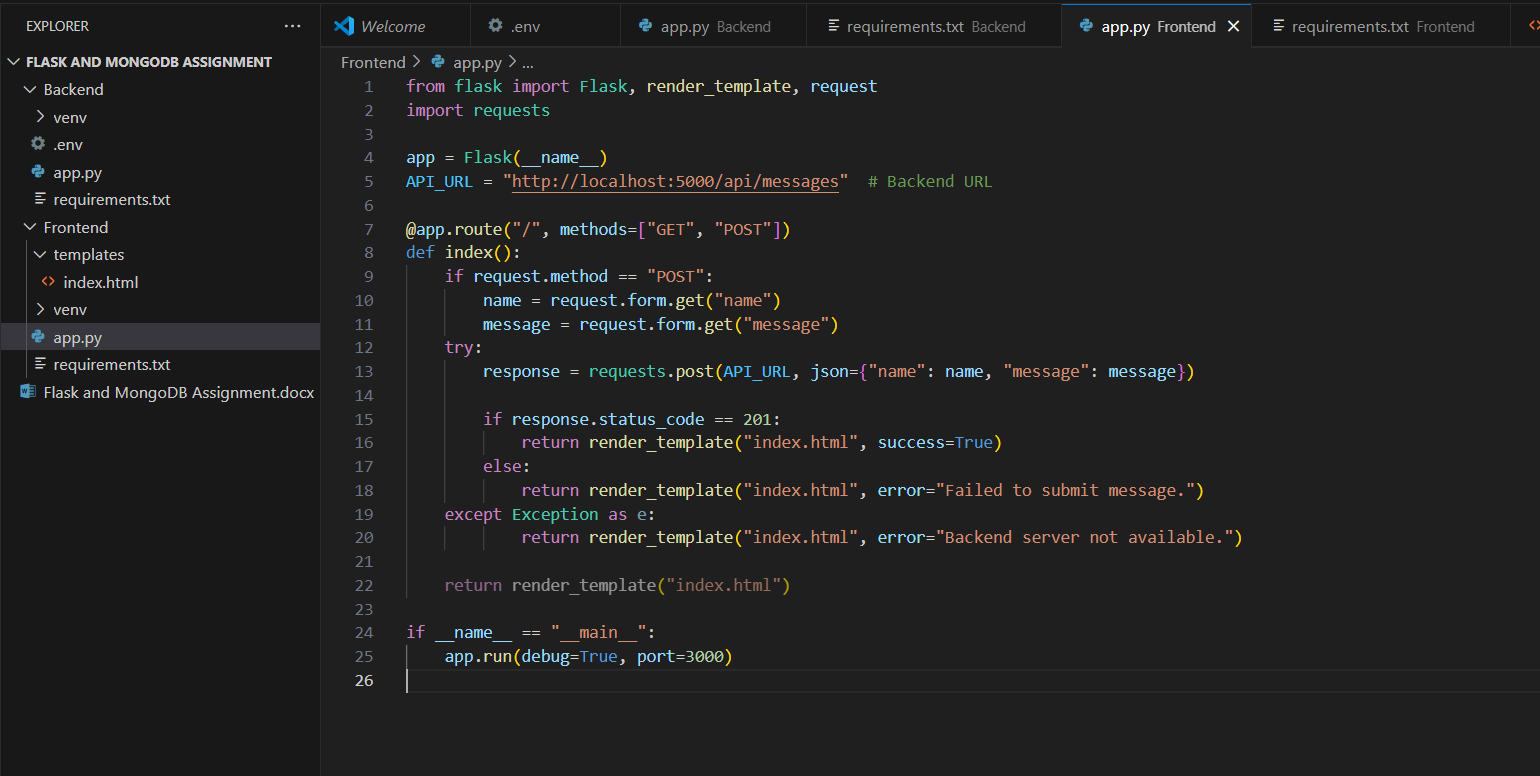
1. **index.html**

****

**Explanation:**

* Provides a **user interface** to submit a name and message.
* Shows a **success or error message** based on the server response.

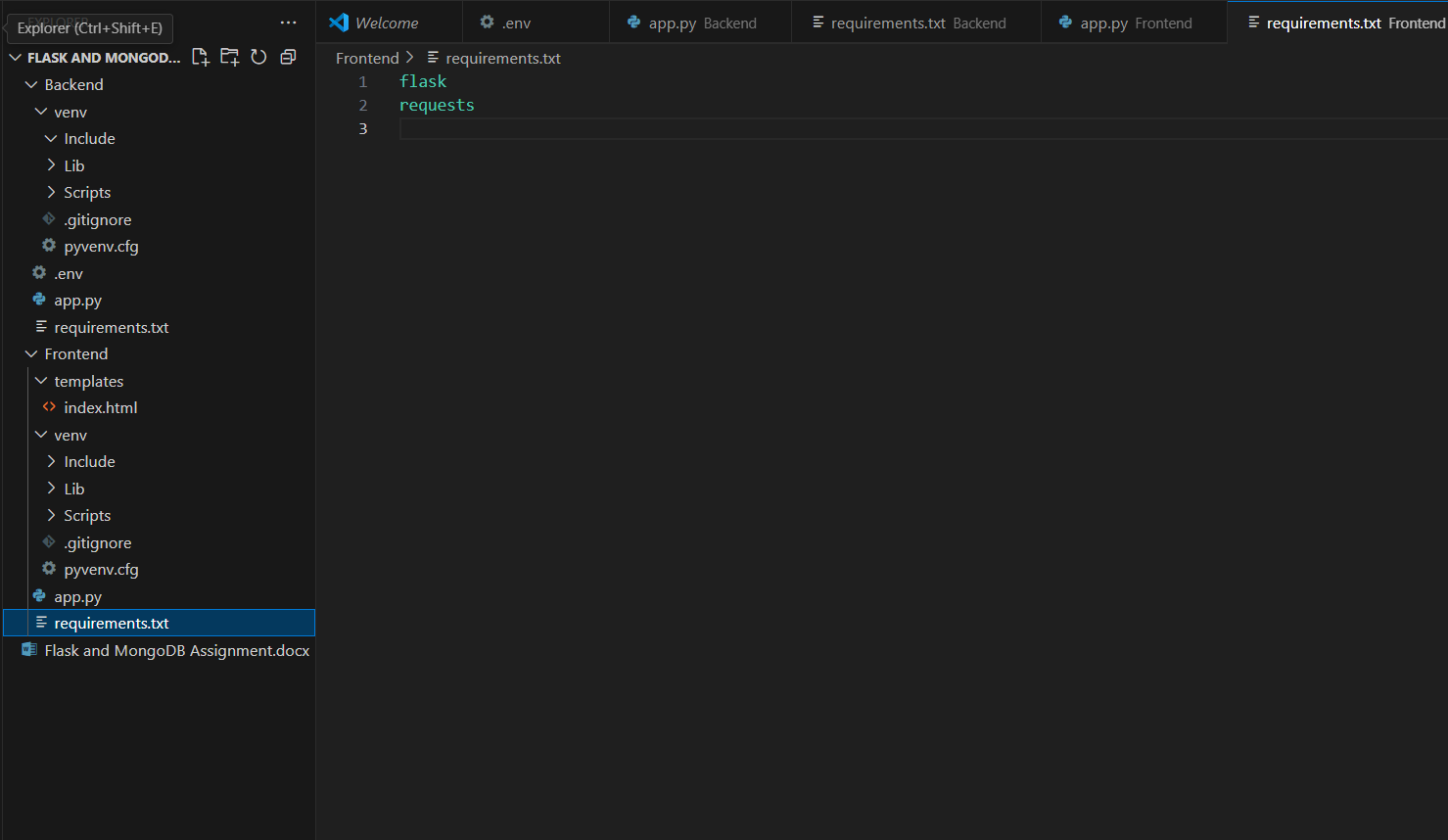
1. **app.py**

****

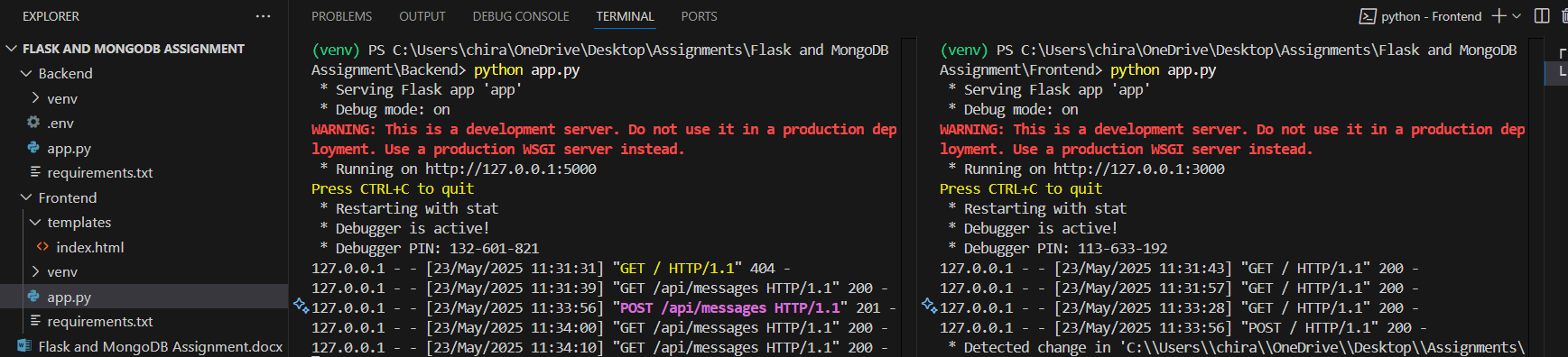
**Explanation:**

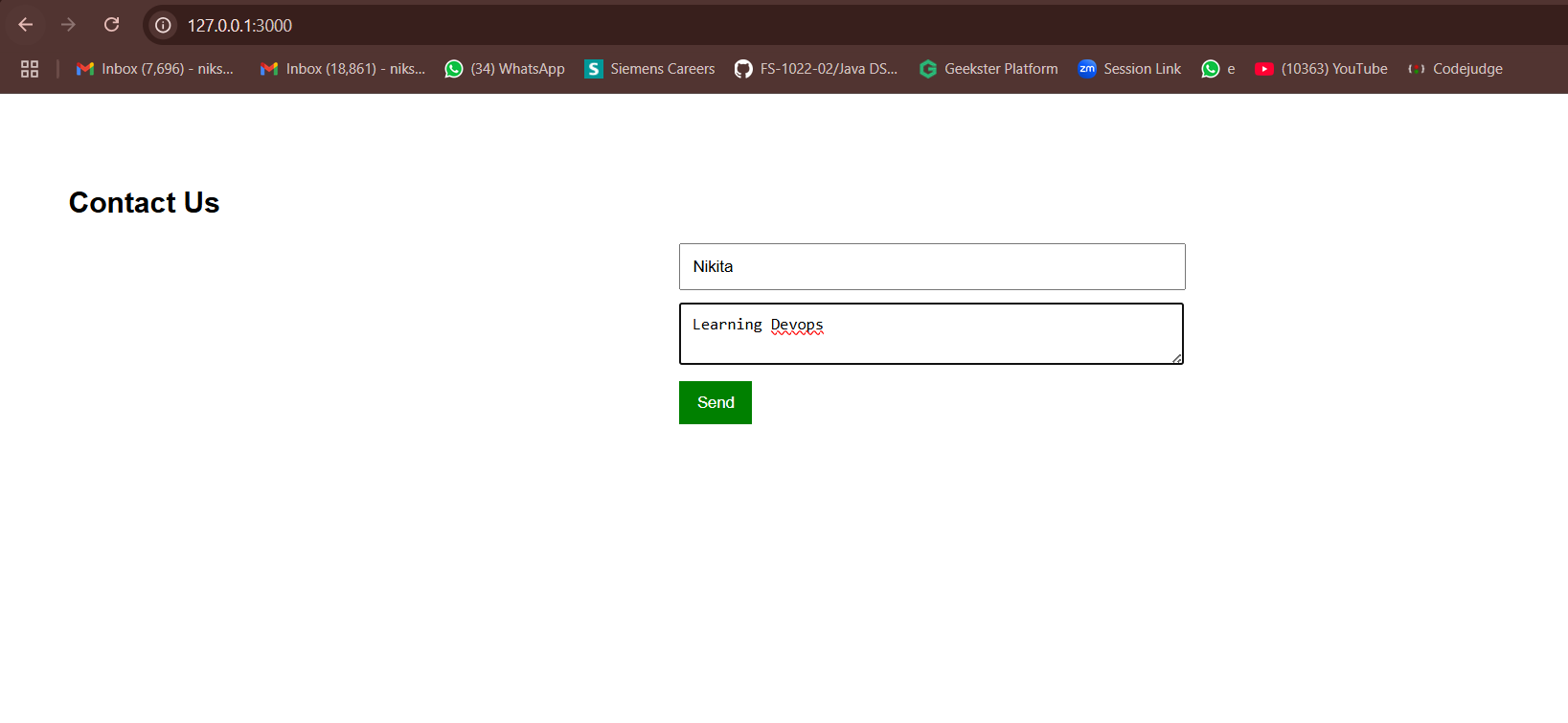
* **Flask:** Creates the frontend web app.
* **render\_template:** Renders index.html.
* **request:** Gets form data from the browser.
* **requests:** Sends HTTP requests to the backend API (like POST).
* **app = Flask(\_\_name\_\_):** Creates a new Flask app.
* **API\_URL:** Points to backend that receives the message
* **@app.route("/"):** Main route for showing and submitting the form
* **success=True / error=...:** Used by Jinja in HTML to show status messages

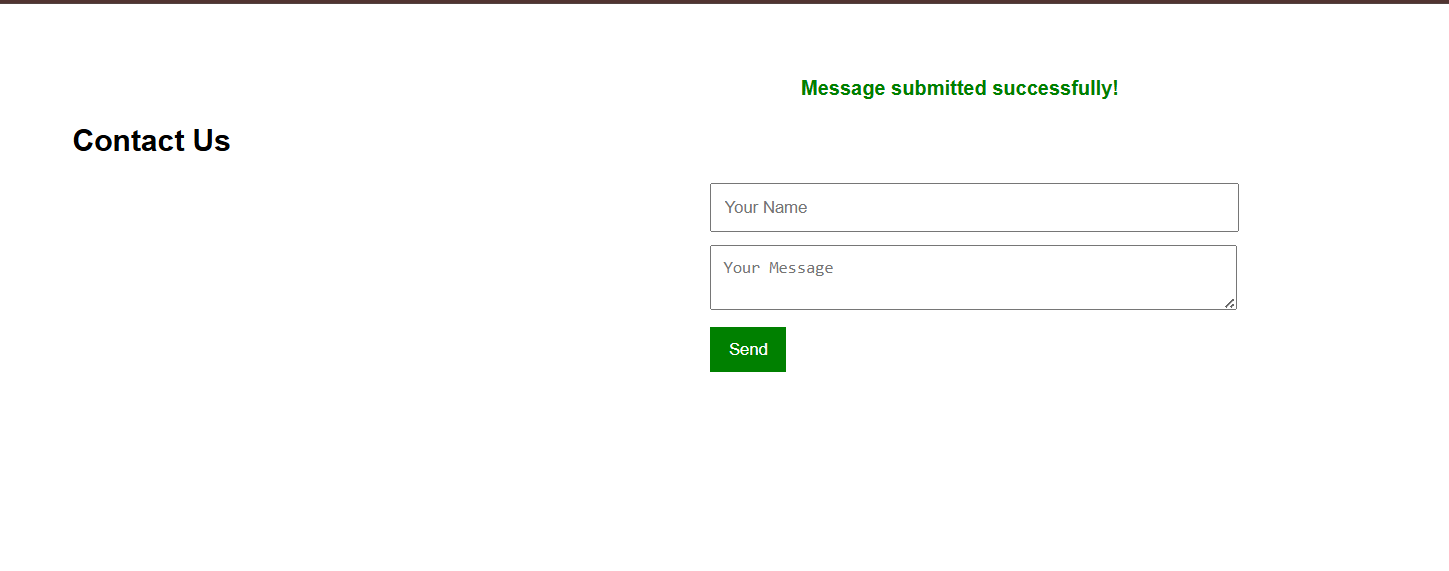
1. **requirements.txt**

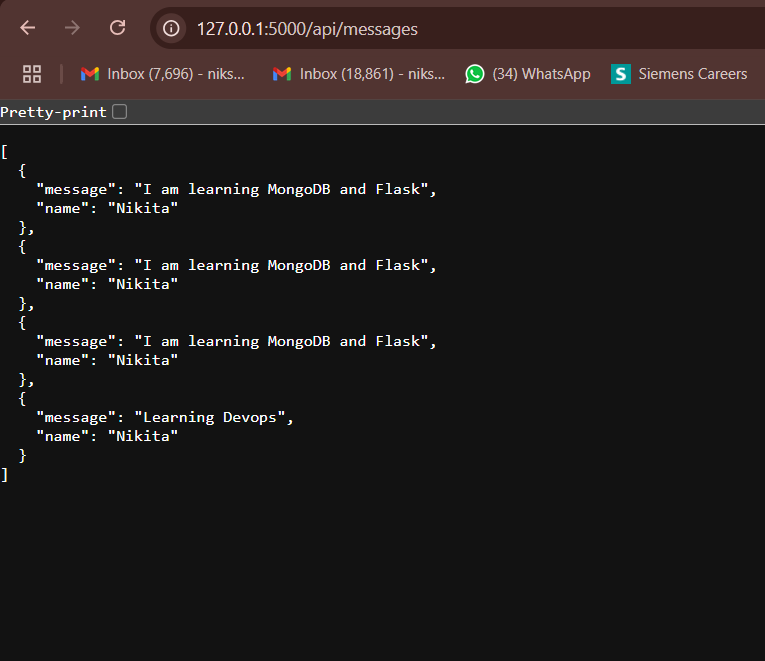
****

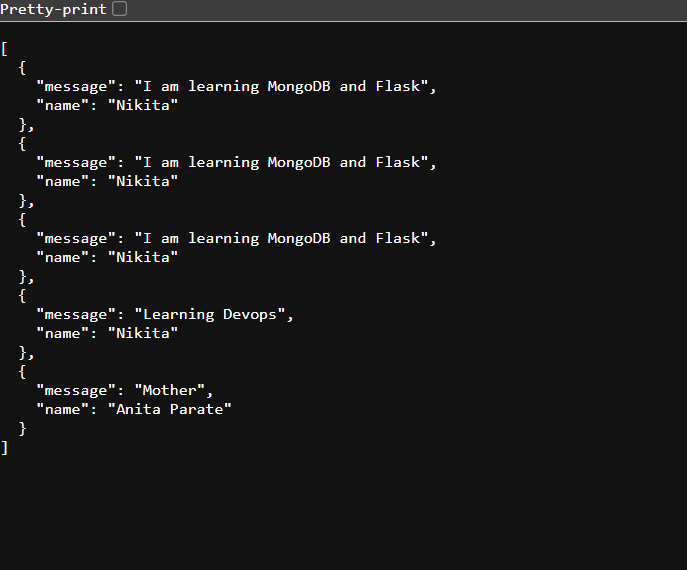
**Explanation:**

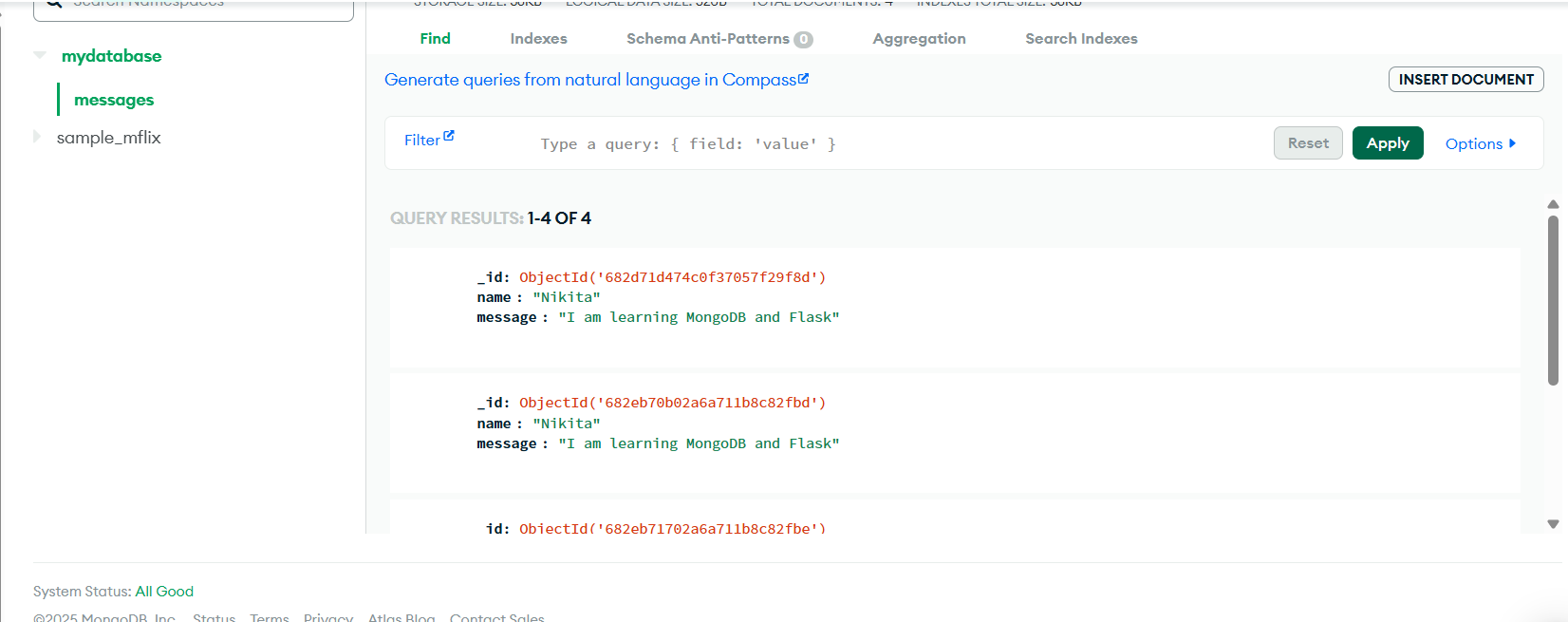
* **requests**: A powerful HTTP library to send requests from Python to other services (like APIs).
* **Result:  
    
  **

****

****

****

****

****