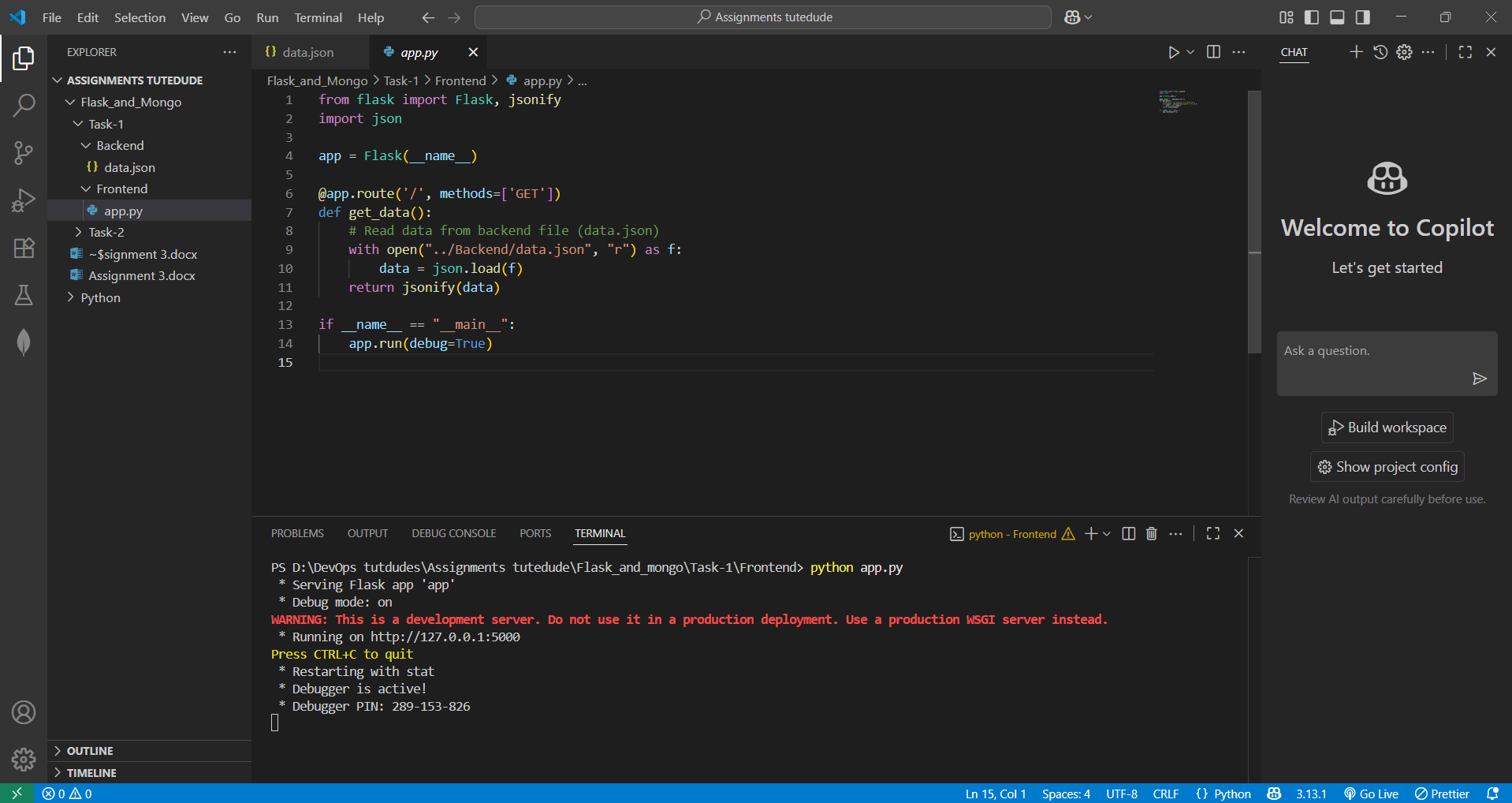
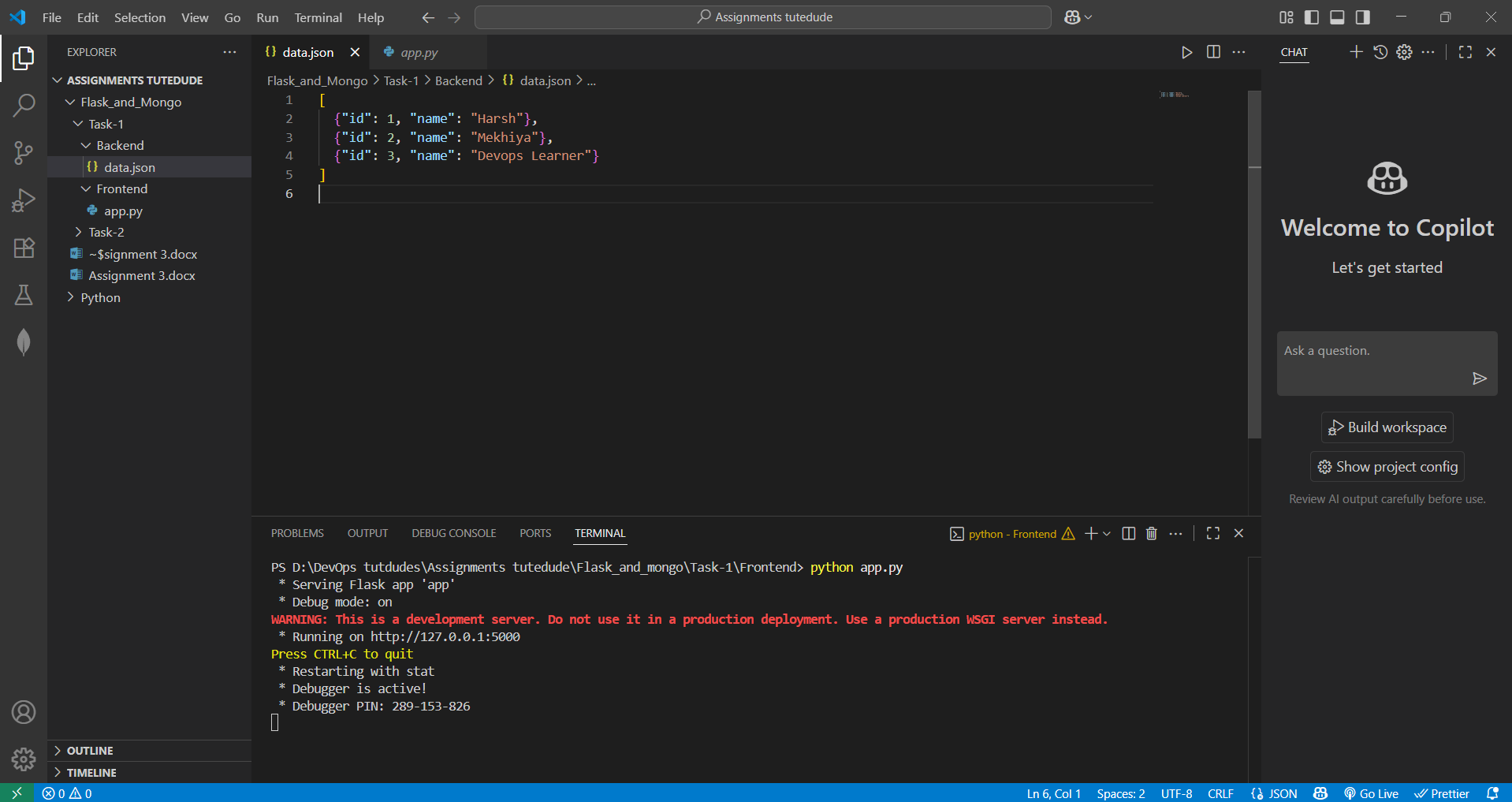
**Assignment: 3 Flask & Mongodb**

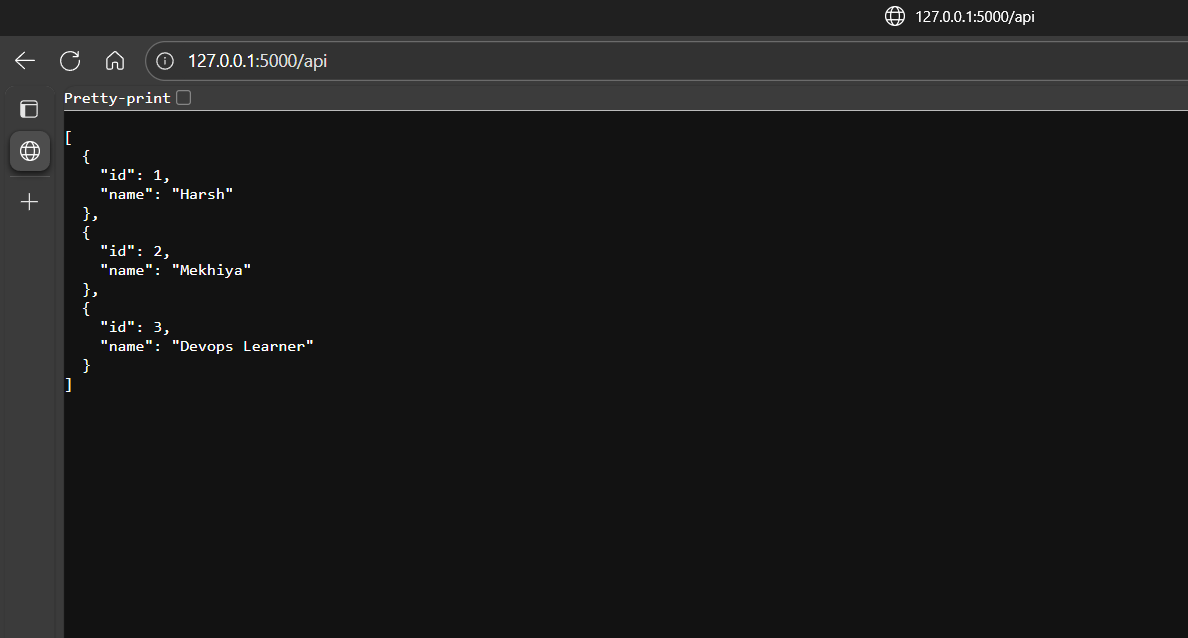
Name: Harsh M. Mekhiya   
email: [harshmekhiya191@gmail.com](mailto:harshmekhiya191@gmail.com)

**Tasks**

1. **Create a Flask application with an /api route. When this route is accessed, it should return a JSON list. The data should be stored in a backend file, read from it, and sent as a response.**







Explanation:

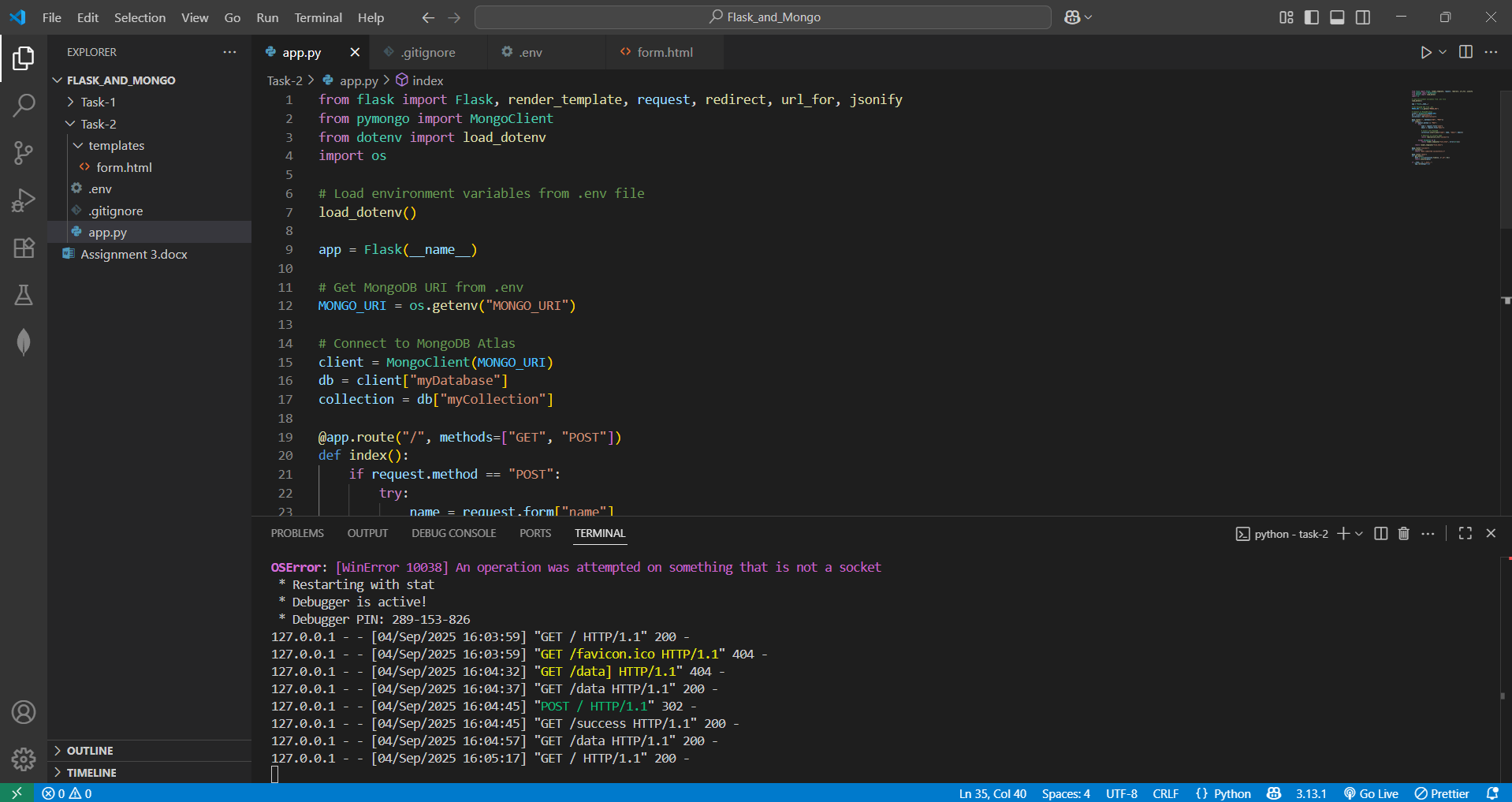
First of all, I created a Flask application by importing Flask and jsonify from the Flask library and also importing Python’s built-in json module. I initialized the Flask app using app = Flask(\_\_name\_\_).

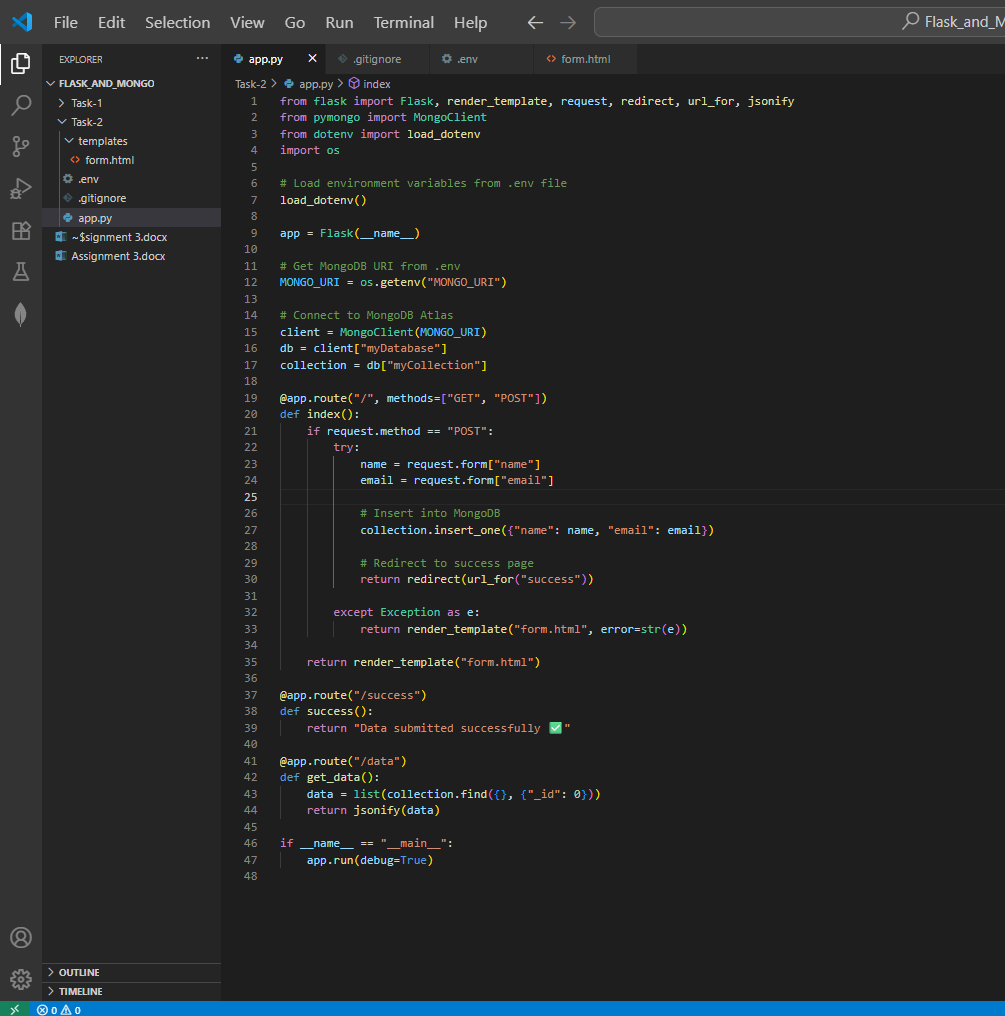
Then, I created an API route /api using the @app.route('/api', methods=['GET']) decorator. Inside this function, I opened a backend file named data.json (which I placed inside the Backend folder). The file contents were read using json.load(f) which converts the JSON data into a Python object. Finally, I returned this data as a **JSON response** using the jsonify() function, so that when the API is called, the browser or client receives properly formatted JSON.

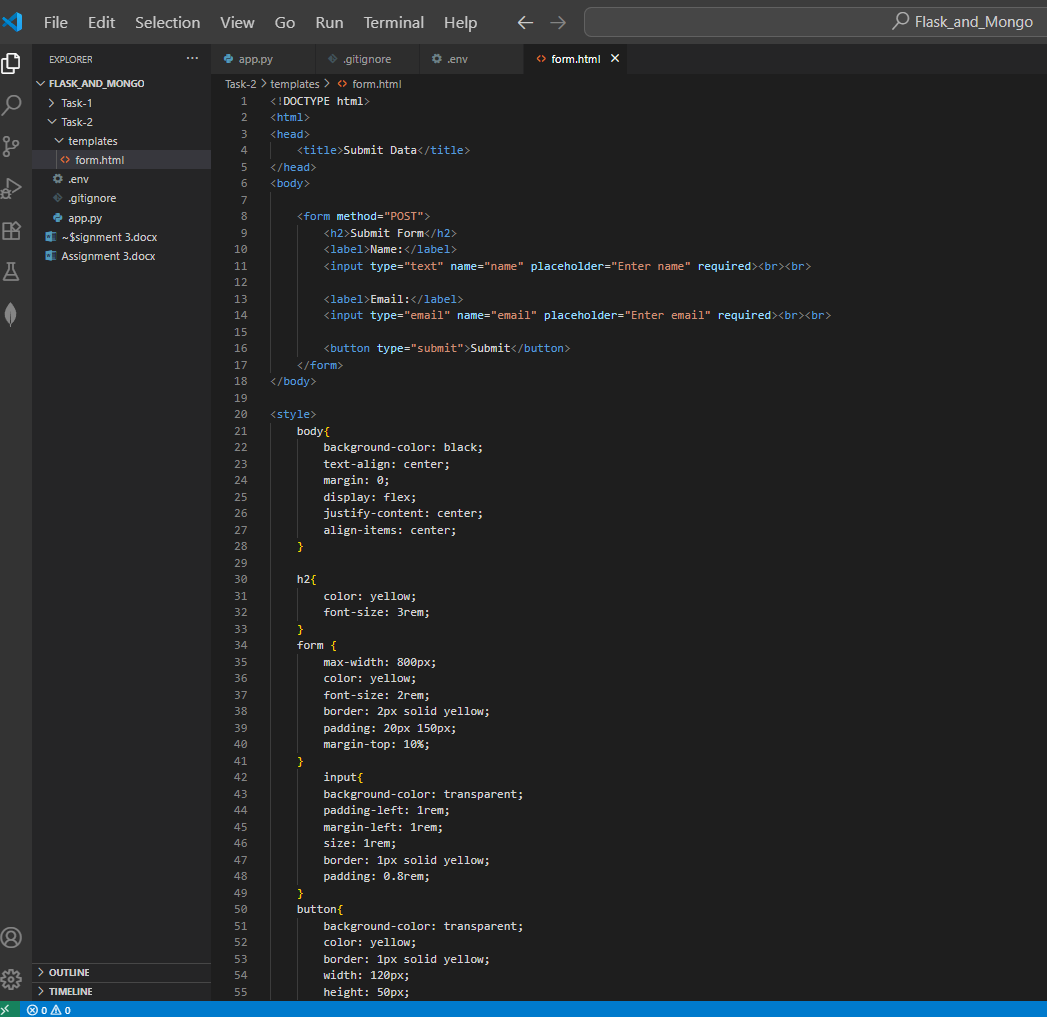
At the end, I used if \_\_name\_\_ == "\_\_main\_\_": app.run(debug=True) to run the application in debug mode. This allows me to test the API locally at http://127.0.0.1:5000/api, and every time I update the file data.json, the API will automatically return the updated JSON list.

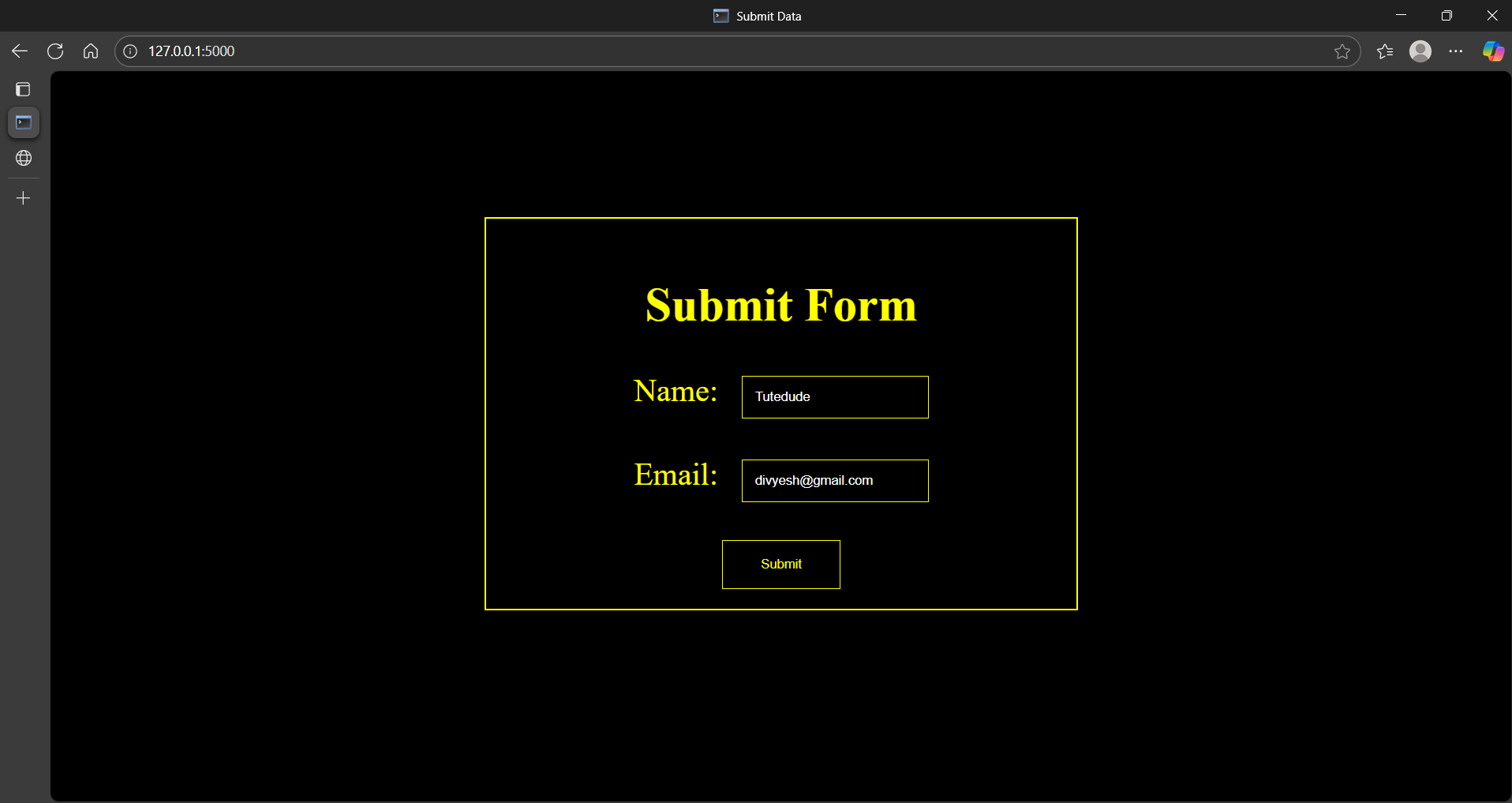
**Tasks**

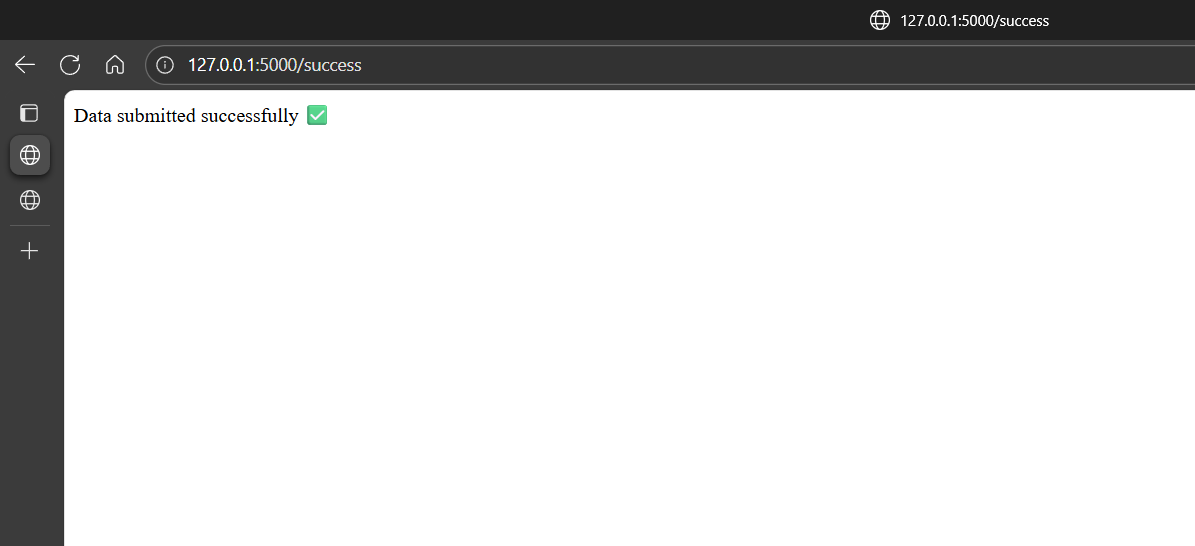
1. **Create a form on the frontend that, when submitted, inserts data into MongoDB Atlas. Upon successful submission, the user should be redirected to another page displaying the message "Data submitted successfully". If there's an error during submission, display the error on the same page without redirection.**

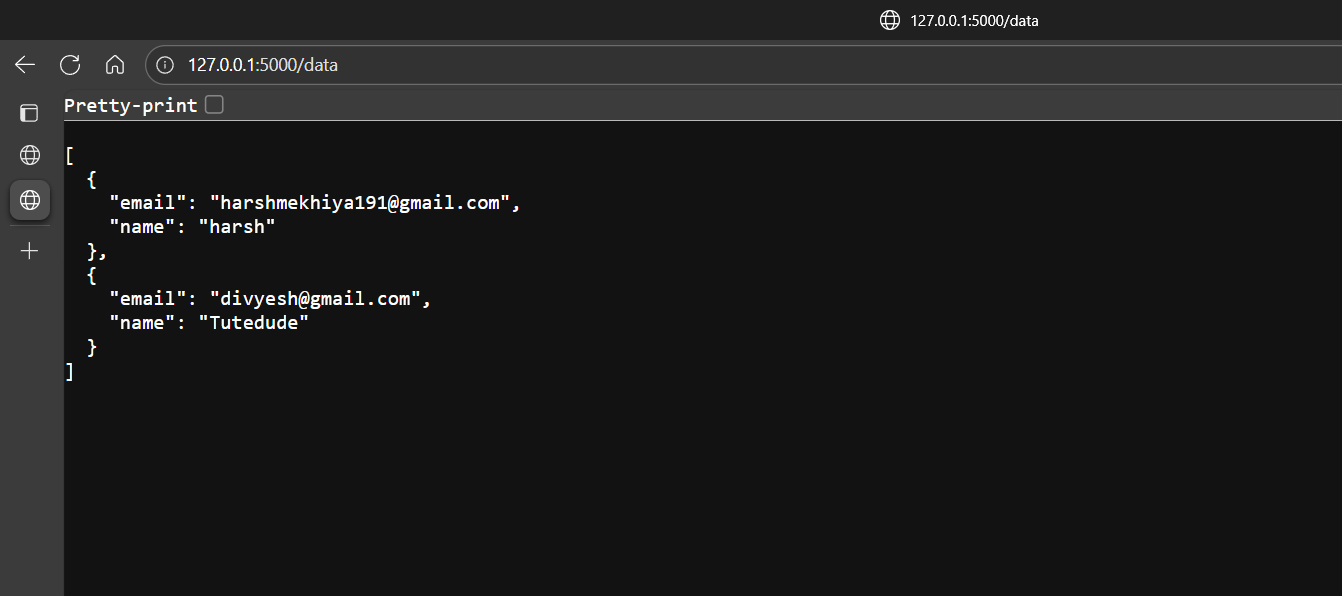
****

****

****

****

****

****

**Explanation:**

First of all, I created a Flask project where I connected my application to **MongoDB Atlas** using the pymongo library. Instead of directly putting the MongoDB URI inside the code, I stored it securely inside an **.env file** and loaded it with python-dotenv. This is a safer way to manage database credentials.

Next, I designed a frontend form (form.html) inside the **templates folder**. The form allowed users to enter their **name** and **email**. When the form was submitted, the data was sent to Flask using a **POST request**. Flask then inserted this data into the MongoDB Atlas collection.

If the insertion was successful, the user was redirected to a new route /success where the message *"Data submitted successfully "* was displayed. On the other hand, if there was any error (for example, connection issues with MongoDB), the same form page reloaded with the error message displayed in red, without redirecting.

Finally, I created another route /data that fetched all the documents from MongoDB Atlas and displayed them as a JSON response in the browser. This way, I could confirm that the submitted form entries were actually stored in the database.

**Summary:**

I learned how to build Flask applications with routes that return JSON data and handle form submissions. I created a frontend form that collects user input and stores it in MongoDB Atlas using pymongo. I also learned how to securely manage database credentials with .env files, handle errors gracefully, and create routes to fetch and display stored data. These tasks gave me hands-on experience in backend development, API creation, and database integration with Flask and MongoDB.