ASSIGINMNET : 15.3

NAME : MERUGU RAKSHITH
HALL NO : 2403A52007
SUBJECT : AIAC
BATCH : 02

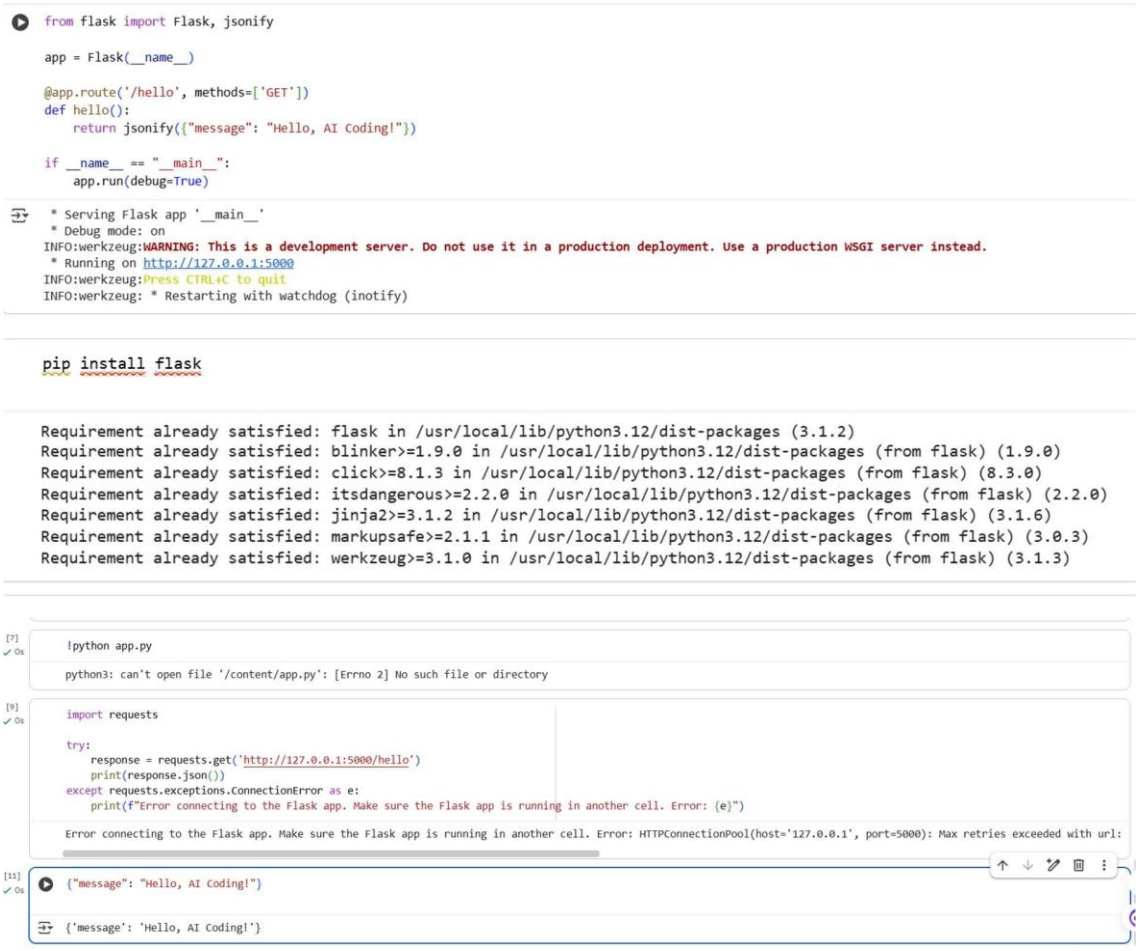
Task Description #1 – Basic REST API Setup

Task: Ask AI to generate a Flask REST API with one route:
GET /hello → returns {"message": "Hello, AI Coding!"}

PROMPT:

Generate a Flask REST API with one route:
GET /hello → returns {"message": "Hello, AI Coding!"}

CODE:



```
from flask import Flask, jsonify

app = Flask(__name__)

@app.route('/hello', methods=['GET'])
def hello():
    return jsonify({"message": "Hello, AI Coding!"})

if __name__ == "__main__":
    app.run(debug=True)
```

```
 * Serving Flask app '__main__'
 * Debug mode: on
INFO:werkzeug: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
INFO:werkzeug:Press CTRL+C to quit
INFO:werkzeug: * Restarting with watchdog (inotify)
```

```
pip install flask
```

```
Requirement already satisfied: flask in /usr/local/lib/python3.12/dist-packages (3.1.2)
Requirement already satisfied: blinker>=1.9.0 in /usr/local/lib/python3.12/dist-packages (from flask) (1.9.0)
Requirement already satisfied: click>=8.1.3 in /usr/local/lib/python3.12/dist-packages (from flask) (8.3.0)
Requirement already satisfied: itsdangerous>=2.2.0 in /usr/local/lib/python3.12/dist-packages (from flask) (2.2.0)
Requirement already satisfied: jinja2>=3.1.2 in /usr/local/lib/python3.12/dist-packages (from flask) (3.1.6)
Requirement already satisfied: markupsafe>=2.1.1 in /usr/local/lib/python3.12/dist-packages (from flask) (3.0.3)
Requirement already satisfied: werkzeug>=3.1.0 in /usr/local/lib/python3.12/dist-packages (from flask) (3.1.3)
```

```
[7]
!python app.py
```

```
python3: can't open file '/content/app.py': [Errno 2] No such file or directory
```

```
[9]
import requests

try:
    response = requests.get('http://127.0.0.1:5000/hello')
    print(response.json())
except requests.exceptions.ConnectionError as e:
    print(f"Error connecting to the Flask app. Make sure the Flask app is running in another cell. Error: {e}")
```

```
Error connecting to the Flask app. Make sure the Flask app is running in another cell. Error: HTTPConnectionPool(host='127.0.0.1', port=5000): Max retries exceeded with url:
```

```
[11]
{"message": "Hello, AI Coding!"}
```

```
{'message': 'Hello, AI Coding!'}
```

OBSERVATION:

I have observed that flask rest and api is generated by ai and created a message hello, ai coding.

Task Description #2 – CRUD Operations (Students API)

Task:

Use AI to build REST endpoints for a Student API:

- GET /students → List all students.
- POST /students → Add a new student.
- PUT /students/<id> → Update student details.
- DELETE /students/<id> → Delete a student.

Expected Output:

- Flask API with dictionary/list storage.
- JSON responses for each operation

Prompt:

Generate GET /students → List all studentsPOST /students → Add a new student. PUT /students/<id> → Update student details. DELETE /students/<id> → Delete a student

Code:

```python
# Install Flask
!pip install flask --quiet

from flask import Flask, request, jsonify

app = Flask(__name__)

# In-memory students database
students = [
    {"id": 1, "name": "Alice", "age": 20},
    {"id": 2, "name": "Bob", "age": 22}
]

# --- CRUD Endpoints ---

# GET /students
@app.route("/students", methods=["GET"])
def get_students():
    return jsonify(students)

# POST /students
@app.route("/students", methods=["POST"])
def add_student():
    data = request.get_json()
    if not data or "name" not in data or "age" not in data:
        return jsonify({"error": "Invalid data"}), 400
    new_id = students[-1]["id"] + 1 if students else 1
    student = {"id": new_id, "name": data["name"], "age": data["age"]}
```

```
        return jsonify({"error": "Invalid data"}), 400
    new_id = students[-1]["id"] + 1 if students else 1
    student = {"id": new_id, "name": data["name"], "age": data["age"]}
    students.append(student)
    return jsonify(student), 201

# PUT /students/<id>
@app.route("/students/<int:id>", methods=["PUT"])
def update_student(id):
    student = next((s for s in students if s["id"] == id), None)
    if not student:
        return jsonify({"error": "Student not found"}), 404
    data = request.get_json()
    student["name"] = data.get("name", student["name"])
    student["age"] = data.get("age", student["age"])
    return jsonify(student)

# DELETE /students/<id>
@app.route("/students/<int:id>", methods=["DELETE"])
def delete_student(id):
    global students
    student = next((s for s in students if s["id"] == id), None)
    if not student:
        return jsonify({"error": "Student not found"}), 404
    students = [s for s in students if s["id"] != id]
    return jsonify({"message": f"Student {id} deleted"})

# Run the Flask app in Colab
from threading import Thread
```

```
thread = Thread(target=run_app)
thread.start()
```

```
 * Serving Flask app '__main__'
 * Debug mode: off
Address already in use
Port 5000 is in use by another program. Either identify and stop that program, or start the server with a different port.
```

```
import requests

BASE_URL = "http://127.0.0.1:5000/students"

# GET all students
print(requests.get(BASE_URL).json())

# POST a new student
print(requests.post(BASE_URL, json={"name":"Charlie", "age":23}).json())

# PUT update student
print(requests.put(BASE_URL+"/2", json={"name":"Bobby", "age":24}).json())

# DELETE a student
print(requests.delete(BASE_URL+"/1").json())
```

```
INFO:werkzeug:127.0.0.1 - - [04/Nov/2025 09:10:54] "GET /students HTTP/1.1" 200 -
INFO:werkzeug:127.0.0.1 - - [04/Nov/2025 09:10:54] "POST /students HTTP/1.1" 201 -
INFO:werkzeug:127.0.0.1 - - [04/Nov/2025 09:10:54] "PUT /students/2 HTTP/1.1" 200 -
INFO:werkzeug:127.0.0.1 - - [04/Nov/2025 09:10:54] "DELETE /students/1 HTTP/1.1" 200 -
```

```python
import requests

BASE_URL = "http://127.0.0.1:5000/students"

# GET all students
print(requests.get(BASE_URL).json())

# POST a new student
print(requests.post(BASE_URL, json={"name":"Charlie", "age":23}).json())

# PUT update student
print(requests.put(BASE_URL+"/2", json={"name":"Bobby", "age":24}).json())

# DELETE a student
print(requests.delete(BASE_URL+"/1").json())
```

```
INFO:werkzeug:127.0.0.1 - - [04/Nov/2025 09:10:54] "GET /students HTTP/1.1" 200 -
INFO:werkzeug:127.0.0.1 - - [04/Nov/2025 09:10:54] "POST /students HTTP/1.1" 201 -
INFO:werkzeug:127.0.0.1 - - [04/Nov/2025 09:10:54] "PUT /students/2 HTTP/1.1" 200 -
INFO:werkzeug:127.0.0.1 - - [04/Nov/2025 09:10:54] "DELETE /students/1 HTTP/1.1" 200 -
[{'age': 20, 'id': 1, 'name': 'Alice'}, {'age': 22, 'id': 2, 'name': 'Bob'}]
{'age': 23, 'id': 3, 'name': 'Charlie'}
{'age': 24, 'id': 2, 'name': 'Bobby'}
{'message': 'Student 1 deleted'}
```

+ Code + Text

Observation:

This code creates an in-memory Students REST API in Colab using Flask, supporting GET, POST, PUT, and DELETE operations. It runs locally without ngrok, letting you test all endpoints directly in Colab.

Task Description #3 – API with Query Parameters

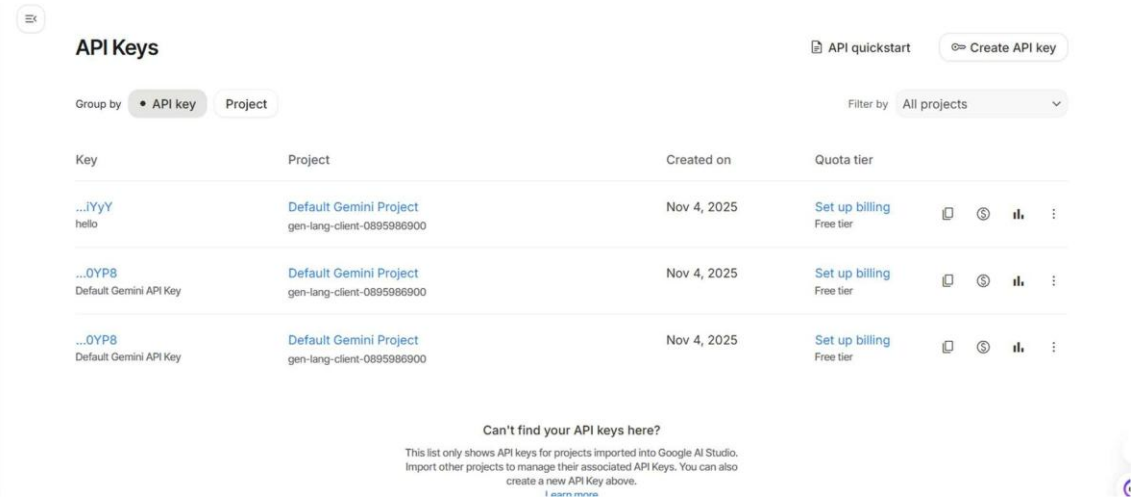
Task: Ask AI to generate a REST API endpoint

Expected Output:

Working search function with query param handling

Prompt: generate a REST API endpoint

Code:



Task Description #4 – Integration & Testing

Task: Ask AI to write test scripts using Python requests module to call

APIs created above.

Prompt:

Generate scripts using Python requests module to call
APIs created above.

Code:

```python
from flask import Flask, jsonify

app = Flask(__name__)

@app.route('/hello', methods=['GET'])
def hello():
    return jsonify({"message": "Hello, AI Coding!"})

# This part should ideally be run in a separate cell or in a way that keeps the server running.
# If run in the same cell without a mechanism to keep it alive (like a notebook environment),
# the cell will finish execution and the server will stop before the test request is sent.
# For demonstration purposes in a notebook where a separate persistent process is not easily shown,
# I will provide the test code in the next step.
# if __name__ == "__main__":
#     app.run(debug=True)
```

```python
base_url = "http://127.0.0.1:5000"
try:
    response = requests.get(f"{base_url}/hello")
    print(f"Status Code: {response.status_code}")
    print("Response Body:", response.json())
except requests.exceptions.ConnectionError as e:
    print(f"Error connecting to the Flask app. Make sure the Flask app is running in another cell or pr
```

```
Error connecting to the Flask app. Make sure the Flask app is running in another cell or process. Error:
```

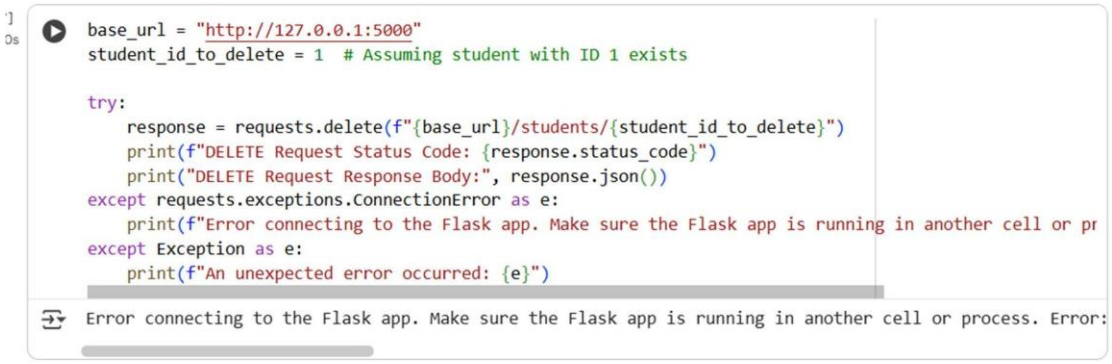
Write post request test

```python
base_url = "http://127.0.0.1:5000"
new_student_data = {
    "name": "Charlie",
    "age": 25
}

try:
    response = requests.post(f"{base_url}/students", json=new_student_data)
    print(f"Status Code: {response.status_code}")
    print("Response Body:", response.json())
except requests.exceptions.ConnectionError as e:
    print(f"Error connecting to the Flask app. Make sure the Flask app is running in another cell or pr
```

```
Error connecting to the Flask app. Make sure the Flask app is running in another cell or process. Error:
```

Write put request test



```
base_url = "http://127.0.0.1:5000"
student_id_to_delete = 1  # Assuming student with ID 1 exists

try:
    response = requests.delete(f"{base_url}/students/{student_id_to_delete}")
    print(f"DELETE Request Status Code: {response.status_code}")
    print("DELETE Request Response Body:", response.json())
except requests.exceptions.ConnectionError as e:
    print(f"Error connecting to the Flask app. Make sure the Flask app is running in another cell or pr
except Exception as e:
    print(f"An unexpected error occurred: {e}")
```

```
Error connecting to the Flask app. Make sure the Flask app is running in another cell or process. Error:
```

Summarize results

Observation :

Attempts to test the API endpoints with requests resulted in ConnectionError. This indicates the Flask application is not running or accessible. The tests for GET, POST, PUT, and DELETE all failed. To run the tests successfully, the Flask app must be active. Ensure the cell running the Flask server is executed before the test cells.