# **Deploying a Flask Application with Nginx on Ubuntu VM**

## **1. Setting Up Python and Virtual Environment**

### **Install Python and Virtual Environment**

sudo apt update  
sudo apt install python3 python3-pip python3-venv -y

### **Create and Activate a Virtual Environment**

mkdir flask\_app  
cd flask\_app  
python3 -m venv venv  
source venv/bin/activate # Activate venv

## **2. Creating the Flask API**

### **Install Required Packages**

pip install flask neo4j

### **Create app.py File**

from flask import Flask, request, jsonify  
from neo4j import GraphDatabase  
  
# Neo4j Connection Details  
NEO4J\_URI = "bolt://YOUR\_VM\_IP:7687" # Replace with actual Neo4j IP  
NEO4J\_USER = "neo4j"  
NEO4J\_PASSWORD = "your\_password"  
  
driver = GraphDatabase.driver(NEO4J\_URI, auth=(NEO4J\_USER, NEO4J\_PASSWORD))  
  
app = Flask(\_\_name\_\_)  
  
def execute\_query(query):  
 try:  
 with driver.session() as session:  
 result = session.run(query)  
 return [record.data() for record in result]  
 except Exception as e:  
 return {"error": str(e)}  
  
@app.route("/execute", methods=["POST"])  
def execute\_cypher():  
 data = request.json  
 query = data.get("query")  
 if not query or "DELETE" in query.upper(): # Prevent DELETE queries  
 return jsonify({"error": "Invalid query or DELETE queries are not allowed"}), 400  
 return jsonify(execute\_query(query))  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 app.run(host="0.0.0.0", port=5000, debug=True)

### **Test the Flask Application Locally**

python app.py

Then, in another terminal, test with:

curl -X POST <http://127.0.0.1:5000/execute> -H "Content-Type: application/json" -d '{"query": "MATCH (n) RETURN n LIMIT 1"}'

## **3. Running Flask in the Background on VM**

Use nohup to keep the Flask application running after logout:

nohup python3 app.py > output.log 2>&1 &

Check if it's running:

ps aux | grep python

## **4. Configuring Nginx as a Reverse Proxy**

### **Install Nginx**

sudo apt install nginx -y

### **Configure Nginx**

Edit the default Nginx configuration:

sudo nano /etc/nginx/sites-available/default

Replace the content with:

server {  
 listen 80;  
 server\_name YOUR\_VM\_IP; # Replace with your VM’s public IP  
  
 location / {  
 proxy\_pass <http://127.0.0.1:5000>;  
 proxy\_set\_header Host $host;  
 proxy\_set\_header X-Real-IP $remote\_addr;  
 proxy\_set\_header X-Forwarded-For $proxy\_add\_x\_forwarded\_for;  
 }  
}

Save and exit (CTRL + X, then Y, then Enter).

### **Restart Nginx**

sudo nginx -t # Test configuration  
sudo systemctl restart nginx

## **5. Allowing External Access**

### **Open Firewall Ports**

sudo ufw allow 80  
sudo ufw allow 5000  
sudo ufw reload

### **Access the API from Public IP**

curl -X POST <http://YOUR_VM_IP/execute> -H "Content-Type: application/json" -d '{"query": "MATCH (n) RETURN n LIMIT 1"}'

## **6. Checking If Everything Works**

### **Verify Flask is Running**

sudo netstat -tulnp | grep 5000

### **Check Nginx Status**

sudo systemctl status nginx

### **Check Flask Logs**

tail -f output.log

### **Test API from Browser/Postman**

Open a browser and visit:

<http://YOUR_VM_IP/execute>

Send a POST request with JSON body:

{  
 "query": "MATCH (n) RETURN n LIMIT 1"  
}

If everything is set up correctly, you should get a JSON response from Neo4j!

## **7. Keeping Flask Running After Reboot**

### **Create a Systemd Service**

sudo nano /etc/systemd/system/flask\_app.service

Add:

[Unit]  
Description=Flask API Service  
After=network.target  
  
[Service]  
User=ubuntu  
WorkingDirectory=/home/ubuntu/flask\_app  
ExecStart=/home/ubuntu/flask\_app/venv/bin/python3 /home/ubuntu/flask\_app/app.py  
Restart=always  
  
[Install]  
WantedBy=multi-user.target

### **Enable and Start the Service**

sudo systemctl daemon-reload  
sudo systemctl enable flask\_app  
sudo systemctl start flask\_app

Check status:

sudo systemctl status flask\_app

Now, the Flask API will start automatically on reboot.