Bangalore Temperature Monitoring & Alert System - AWS DevOps Project

1■■ Prerequisites

AWS Account (Free tier is fine)
Python 3.x installed
GitHub account
Basic AWS knowledge (Lambda, S3, SNS, CloudWatch)
OpenWeatherMap API Key → Sign up here (Free plan works)

2■■ AWS Services Used

AWS S3 → Store temperature data (JSON format)
AWS Lambda → Python script to fetch & process data
AWS CloudWatch → Trigger Lambda every 10 mins
AWS SNS → Send alerts via email
AWS IAM → Roles & permissions for Lambda to access S3 & SNS

3■■ Project Architecture

[CloudWatch Schedule] ---> [Lambda Function] ---> [S3 Bucket]
■---> [SNS Alert]

4■■ Step-by-Step Implementation

Step 1: Create an S3 Bucket
- Go to AWS S3 → Create Bucket → Name: bangalore-temp-data
- Region: ap-south-1 (Mumbai)

Step 2: Create an SNS Topic
- SNS → Create Topic → Name: temperature-alerts
- Create subscription (Email) → Confirm email

Step 3: Create Lambda Function
- AWS Lambda → Create Function (Python 3.12)
- IAM Role with S3FullAccess & SNSFullAccess

Step 4: Lambda Python Code:

```
import json
import requests
import boto3
import datetime
import os

s3 = boto3.client('s3')
sns = boto3.client('sns')

BUCKET_NAME = os.environ['BUCKET_NAME']
TOPIC_ARN = os.environ['TOPIC_ARN']
API_KEY = os.environ['API_KEY']

def lambda_handler(event, context):
    city = "Bangalore"
    url = f"http://api.openweathermap.org/data/2.5/weather?q={city}&appid={API_KEY}&units=metric"

    response = requests.get(url)
    data = response.json()

    if response.status_code != 200:
        print("Error fetching data:", data)
        return

    temp = data['main']['temp']
    timestamp = datetime.datetime.now().strftime("%Y-%m-%d_%H-%M-%S")

    file_name = f"{timestamp}.json"
    s3.put_object(
        Bucket=BUCKET_NAME,
        Key=file_name,
        Body=json.dumps(data),
        ContentType='application/json'
    )

    print(f"Data saved to S3: {file_name}")

    if temp > 35:
        message = f"■■ Alert! Temperature in Bangalore is {temp}°C"
        sns.publish(TopicArn=TOPIC_ARN, Message=message, Subject="Bangalore Temperature Alert")
        print("Alert sent:", message)

    return {"statusCode": 200, "body": json.dumps("Execution completed")}
```

Step 5: Add Environment Variables in Lambda
- BUCKET_NAME → bangalore-temp-data
- TOPIC_ARN → your SNS topic ARN
- API_KEY → your OpenWeatherMap API key

Step 6: Set CloudWatch Trigger
- CloudWatch → Create Rule → Schedule: Every 10 minutes → Target: Lambda

Step 7: Test
- Run Lambda → Data saved in S3
- If temp > 35°C → Email alert sent

5■■ How to Push to GitHub

Push to GitHub:
1. mkdir bangalore-temp-monitor && cd bangalore-temp-monitor
2. Add lambda_function.py, README.md, requirements.txt (requests, boto3)
3. git init && git add . && git commit -m "Bangalore Temperature Monitoring Project"
4. git branch -M main
5. git remote add origin https://github.com//bangalore-temp-monitor.git
6. git push -u origin main

6■■ Interview Explanation

Interview Explanation: "I built an AWS-based Bangalore Temperature Monitoring system. It uses AWS Lambda to fetch data from OpenWeatherMap API every 10 minutes, stores it in S3, and sends alerts via SNS when temperature crosses a threshold. I automated the pipeline using CloudWatch and wrote Infrastructure as Code using Terraform.