ArchiTech Assignment

1. EC2 instance Setup

* Launched Ubuntu EC2 instance with t2.micro size, 20 GB storage (for better storage)
* Created a new keypair .pem file – SimpleWebApp.pem
* Created an Elastic IP and associated it with the instance (so the IP shall not be changed again and again)
* Commands :
  + Updating the packages : sudo apt update
  + Install python and pip : sudo apt install python3 python3-pip -y
  + Install Git : sudo apt install git -y
  + Install nginx : sudo apt install nginx -y
* A screenshot of a computer

  Description automatically generatedWhen checked on Instance ip, we get the nginx home page.
* Created a virtual environment : sudo apt install python3.12-venv OR sudo apt-get install python3-venv
* sudo python3 -m venv /home/ubuntu/pythonApp/venv
* Activate the venv : source venv/bin/activate
* Install flask : pip3 install flask
* Install GUNICORN : pip3 install gunicorn

1. Deploying an app

All the pre-requisistes for deploying an application is done. Hence part 2 is done

The app is running using Gunicorn in EC2 instance.

A screenshot of a computer

Description automatically generated

1. Create a Static file hosting in S3

* Created a bucket with name as given – devops-assignment-shashank
* Created 2 files – Introduction.txt and error.txt
* Uploaded the 2 files in s3 bucket.
* The public access is enabled.
* Static website hosting is enabled and specified the Introduction.txt and error.txt files as the files to be visible when URL is hit.
* For the file to be accessible the bucket policy has been set to Get Object and is entered in S3 bucket policy.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

4. Create CICD Pipeline.

The CICD pipeline is created using AWS Developer Tools service.

The Source is set as GitHub. Any commit done in GitHub will trigger the pipeline. The Deploy stage has deployment group created which has the target EC2 instance. The deploy stage is created using Code Deploy service. For Code Deploy to work, an appspec.yml file is created and saved in the GitHub at root location. It specifies the phases in which the pre-requisites are supposed to be done. All the dependencies for app are specified using install\_dependencies.sh file and also the existing app running and run the latest app.   
The pipeline works as it is supposed to be.

A screenshot of a computer

Description automatically generated

The image shows that an update made in github in the app.py file, triggered the pipeline and is successfully executed. This message is just below the red box marked.

The IP is given here, if you wish to make changes in app.py and check the app deployment : http://52.1.52.67

5. Manage Access and Permissions.

The IAM Role is create with required permissions and attached to EC2 instance using option ‘Actions’ 🡪 Modify Role 🡪 Select the IAM role.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated  
The role contains the permissions required for the Code Pipeline to work and S3 bucket to interact with EC2.

A screenshot of a computer

Description automatically generated  
Port 80 is open for the instance.

6. To create Cron job

Sudo apt install python3-pandas

A cron job is created in ec2 instance.

\*/5 \* \* \* \* /home/ubuntu/cron-job/cron-job.sh . The cron-job is to check the health status.

To start and stop the instance at specified time, a lambda function is created to manage the ction of instance – start/stop.

lambda code :   
  
import boto3

ec2 = boto3.client('ec2')

def lambda\_handler(event, context):

instance\_id = 'i-040d914e62abcc832' # Replace with your instance ID

action = event.get('action')

if action == 'start':

response = ec2.start\_instances(InstanceIds=[instance\_id])

print(f"Starting instance {instance\_id}")

elif action == 'stop':

response = ec2.stop\_instances(InstanceIds=[instance\_id])

print(f"Stopping instance {instance\_id}")

else:

raise ValueError("Invalid action: must be 'start' or 'stop'")

return response  
  
  
2 Event bridge rules are created to trigger the lambda with payload { “action” : “start”} and similarly to stop the instance.

The rule is to start the instance at 8 am daily for 5 days and stop istance at 17 :00:00 daily for 5 days.

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated

A screenshot of a computer

Description automatically generated