Project - 3: AWS CAPSTONE

(DEPLOYING THE APPLICATION THROUGH MUTLI-TIER ARCHITECTURE)

Assignment Submitted By:-Hitesh Chauhan

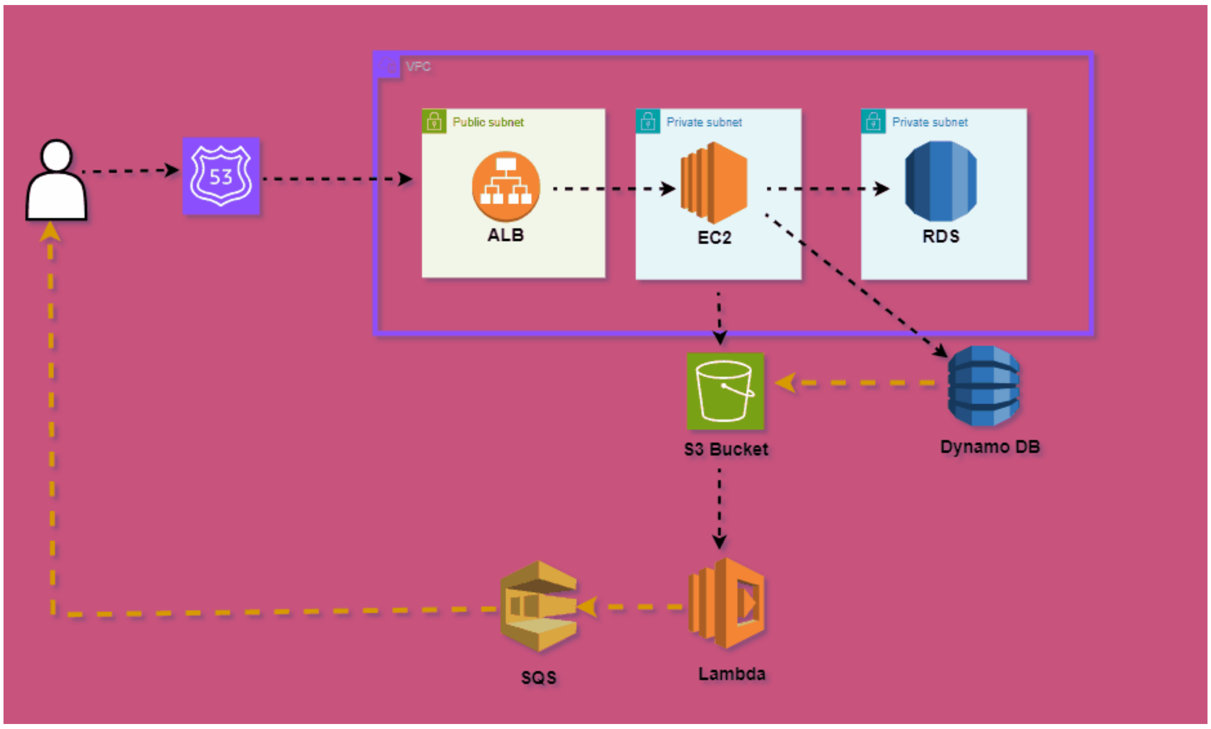
Course Offered: -Advanced Cloud Computing and Devops

Assignment By: -Intellipaat

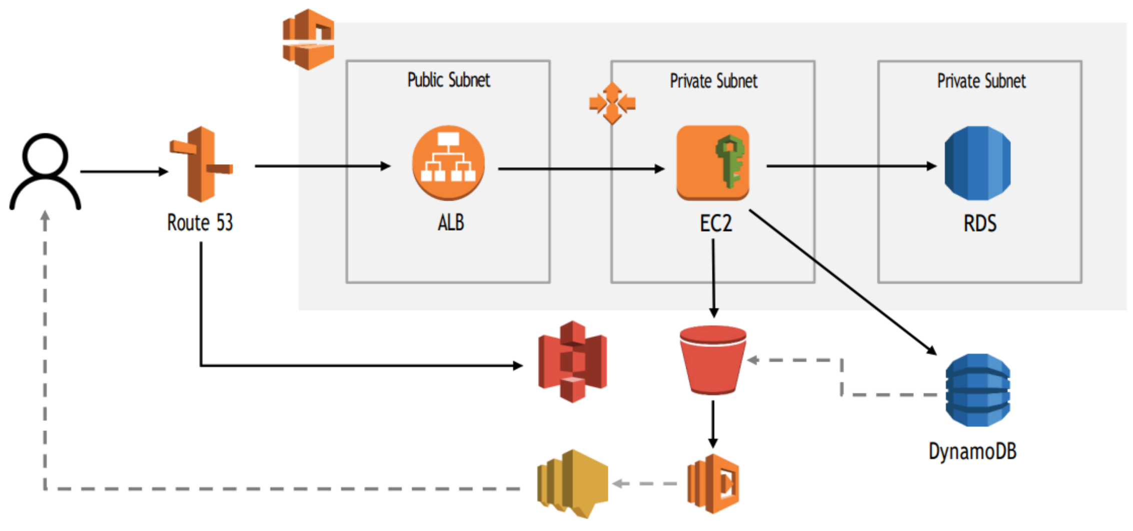
Date Of Submission: -14/03/2025

Employee profile of XYZ company – New employees input their information and upload photos. Existing employees can get their information.

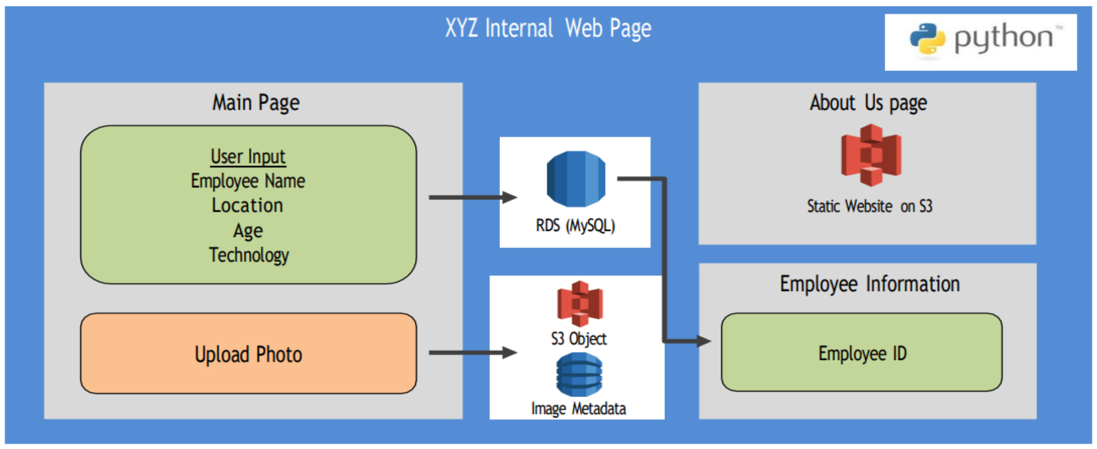
Here, we will work on an employee profile web application to send employee information, upload photos to specific RDS databases, S3 Bucket, and get notifications about the employee data to respect the user or admin team. We will publish the information to an Amazon SQS which keeps it secure and private to respect individual email or phone numbers.



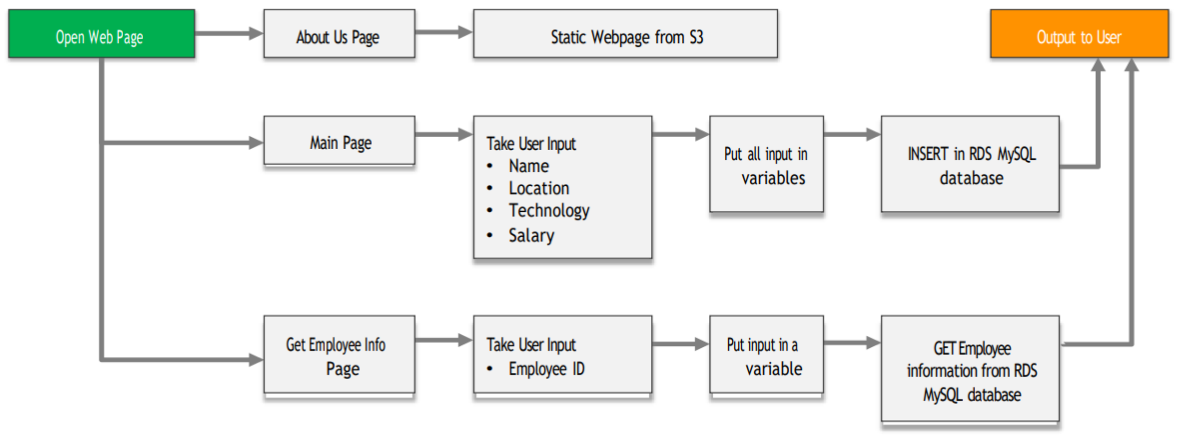
Technical Architecture



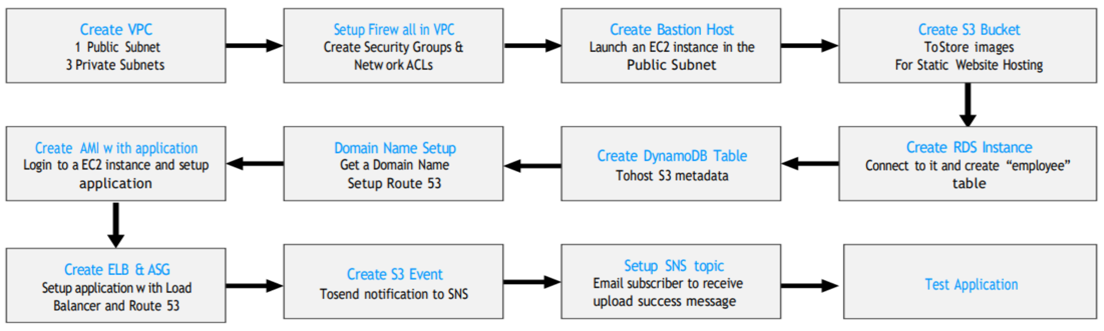
Application Architecture



Application Logic



Steps



Prerequisite:

Basic understanding of Python Flask Framework.

An AWS account and access credentials to configure the S3 connection.

Lambda Function and SQS Topic

A Flask web application project repository. (Don't worry if you don't have a repository feel free to use this repository **https://github.com/hiteshchauhan89/aws-code-main.git**)

This is Optional but Cloud Formation makes the creation of AWS resources easy. (Don’t worry if you don’t know feel free to use this repository **https://github.com/hiteshchauhan89/Cloud-formation-Template-Code/blob/main/vpc-new.yaml)**

Getting Started

Set up VPC for Load Balancer, Ec2 instance, and RDS Database — Two public and two private subnets. (But in the title Image you see one public subnet because Load Balancer required 2 subnets 1 for the current run and another for backup if the failure happens then switch to that one)

Created VPC with 1 public and 3 private subnets.

Security Group & Network ACLs for secure communication.

EC2 Instance in the public subnet.

S3 Bucket for static website hosting.

DynamoDB Table for additional data storage.

#Route 53 Domain Setup (placeholder for custom domain).

#AMI Creation for application deployment.

#Elastic Load Balancer (ELB) & Auto Scaling Group (ASG) for high availability.

S3 Event Notifications for automation.

SNS Topic for alerts and testing.

Step — 1:- Navigate to Cloud Formation and click Create New Stack, upload or copy the file as mentioned above. Now check whether all AWS resources are created or not.

Set up VPC for Load Balancer, Ec2 instance, and RDS Database — Two public and two private subnets. (But in the title Image you see one public subnet because Load Balancer required 2 subnets 1 for the current run and another for backup if the failure happens then switch to that one)

This is the aws yaml code for this projects

```
AWSTemplateFormatVersion: "2010-09-09"

Description: "CloudFormation template for VPC, EC2, S3, DynamoDB, S3 Event Notifications, and SNS."


Parameters:

  VpcCIDR:

    Type: String

    Default: "10.0.0.0/16"
```

```
 PublicSubnetCIDR:
   Type: String
   Default: "10.0.1.0/24"

 PrivateSubnet1CIDR:
   Type: String
   Default: "10.0.2.0/24"

 PrivateSubnet2CIDR:
   Type: String
   Default: "10.0.3.0/24"

 PrivateSubnet3CIDR:
   Type: String
   Default: "10.0.4.0/24"

 ImageId:
   Type: AWS::EC2::Image::Id
   Description: "Provide a valid AMI ID for your AWS region"

 InstanceType:
   Type: String
   Default: "t2.micro"

 KeyName:
   Type: AWS::EC2::KeyPair::KeyName
   Description: "Enter the name of an existing EC2 KeyPair"

Resources:
```

```
# VPC
MyVPC:
  Type: AWS::EC2::VPC
  Properties:
    CidrBlock: !Ref VpcCIDR
    EnableDnsSupport: true
    EnableDnsHostnames: true
    Tags:
      - Key: Name
        Value: EmployeeVPC

# Subnets
PublicSubnet:
  Type: AWS::EC2::Subnet
  Properties:
    VpcId: !Ref MyVPC
    CidrBlock: !Ref PublicSubnetCIDR
    MapPublicIpOnLaunch: true
    AvailabilityZone: !Select [0, !GetAZs ""]

PrivateSubnet1:
  Type: AWS::EC2::Subnet
  Properties:
    VpcId: !Ref MyVPC
    CidrBlock: !Ref PrivateSubnet1CIDR
    AvailabilityZone: !Select [1, !GetAZs ""]

PrivateSubnet2:
  Type: AWS::EC2::Subnet
  Properties:
    VpcId: !Ref MyVPC
    CidrBlock: !Ref PrivateSubnet2CIDR
    AvailabilityZone: !Select [2, !GetAZs ""]

  PrivateSubnet3:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref MyVPC
      CidrBlock: !Ref PrivateSubnet3CIDR
      AvailabilityZone: !Select [0, !GetAZs ""]

  # Internet Gateway & Route Table
  InternetGateway:
    Type: AWS::EC2::InternetGateway

  AttachGateway:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      VpcId: !Ref MyVPC
      InternetGatewayId: !Ref InternetGateway

  PublicRouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref MyVPC

  PublicRoute:
    Type: AWS::EC2::Route
    DependsOn: AttachGateway
    Properties:
```

```
    RouteTableId: !Ref PublicRouteTable
    DestinationCidrBlock: "0.0.0.0/0"
    GatewayId: !Ref InternetGateway

# Security Group
InstanceSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupDescription: "Allow SSH and HTTP access"
    VpcId: !Ref MyVPC
    SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: 22
        ToPort: 22
        CidrIp: 0.0.0.0/0
      - IpProtocol: tcp
        FromPort: 80
        ToPort: 80
        CidrIp: 0.0.0.0/0

# Network ACL
PublicSubnetACL:
  Type: AWS::EC2::NetworkAcl
  Properties:
    VpcId: !Ref MyVPC

AllowInboundHTTP:
  Type: AWS::EC2::NetworkAclEntry
  Properties:
    NetworkAclId: !Ref PublicSubnetACL
```

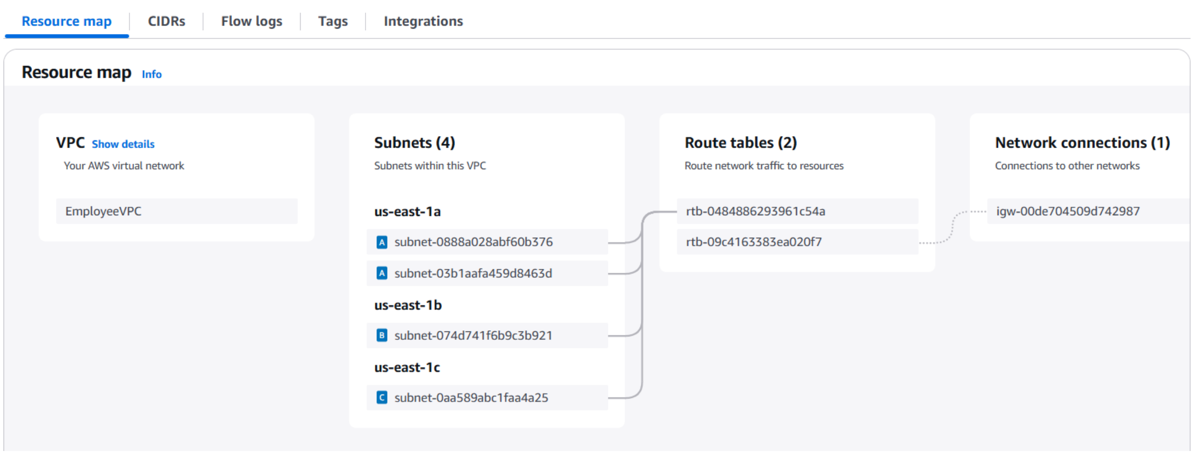
```
      RuleNumber: 100
      Protocol: 6
      RuleAction: allow
      Egress: false
      CidrBlock: 0.0.0.0/0
      PortRange:
        From: 80
        To: 80

  # EC2 Instance
  EC2Instance:
    Type: AWS::EC2::Instance
    Properties:
      InstanceType: !Ref InstanceType
      ImageId: !Ref ImageId
      KeyName: !Ref KeyName
      SubnetId: !Ref PublicSubnet
      SecurityGroupIds:
        - !Ref InstanceSecurityGroup
      Tags:
        - Key: Name
          Value: EmployeeAppInstance
      UserData:
        Fn::Base64: !Sub |
          #!/bin/bash -xe
          sudo yum update -y
          sudo yum install -y python3 python3-pip git mysql
          sudo pip3 install flask boto3 pymysql
          cd /home/ec2-user
          git clone https://github.com/hiteshchauhan89/aws-code-main.git
    cd aws-code-main
    nohup python3 EmpApp.py > app.log 2>&1 &

# S3 Bucket for Static Website Hosting
EmployeeS3Bucket:
 Type: AWS::S3::Bucket
 Properties:
  BucketName: !Sub "employee-static-website-${AWS::AccountId}"
  WebsiteConfiguration:
   IndexDocument: "index.html"
   ErrorDocument: "error.html"
  PublicAccessBlockConfiguration:
   BlockPublicAcls: false
   BlockPublicPolicy: false
   IgnorePublicAcls: false
   RestrictPublicBuckets: false

# DynamoDB Table
EmployeeDynamoDBTable:
 Type: AWS::DynamoDB::Table
 Properties:
  TableName: "EmployeeData"
  AttributeDefinitions:
   - AttributeName: "EmployeeID"
    AttributeType: "S"
  KeySchema:
   - AttributeName: "EmployeeID"
    KeyType: "HASH"
  BillingMode: PAY_PER_REQUEST
```

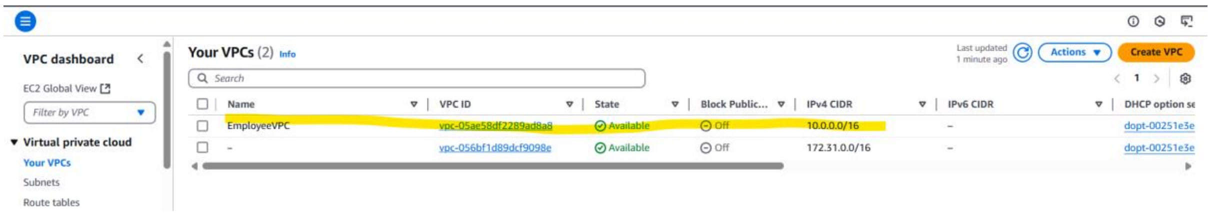
```
# S3 Event Notifications (Triggers SNS)
S3EventNotification:
 Type: AWS::Lambda::Permission
 Properties:
  Action: "lambda:InvokeFunction"
  FunctionName: !Ref S3LambdaFunction
  Principal: "s3.amazonaws.com"
  SourceArn: !GetAtt EmployeeS3Bucket.Arn

# Lambda Function for S3 Event
S3LambdaFunction:
 Type: AWS::Lambda::Function
 Properties:
  FunctionName: "S3EventProcessor"
  Runtime: "python3.8"
  Handler: "index.lambda_handler"
  Role: !GetAtt LambdaExecutionRole.Arn
  Code:
   ZipFile: |
    import json
    def lambda_handler(event, context):
      print("S3 Event Received:", json.dumps(event))
      return {"statusCode": 200, "body": "Event Processed"}

# Lambda Execution Role
LambdaExecutionRole:
 Type: AWS::IAM::Role
 Properties:
  AssumeRolePolicyDocument:
   Version: "2012-10-17"
```

```
      Statement:
        - Effect: Allow
          Principal:
            Service: "lambda.amazonaws.com"
          Action: "sts:AssumeRole"
    Policies:
      - PolicyName: "LambdaS3AccessPolicy"
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
            - Effect: Allow
              Action:
                - "s3:GetObject"
                - "s3:PutObject"
              Resource: !Sub "${EmployeeS3Bucket.Arn}/*"

  # SNS Topic
  EmployeeSNSTopic:
    Type: AWS::SNS::Topic
    Properties:
      TopicName: "EmployeeNotifications"

  EmployeeSNSTopicSubscription:
    Type: AWS::SNS::Subscription
    Properties:
      TopicArn: !Ref EmployeeSNSTopic
      Protocol: email
      Endpoint: "your-email@example.com"

Outputs:
```

```
VpcId:
  Value: !Ref MyVPC

PublicSubnetId:
  Value: !Ref PublicSubnet

EC2InstanceId:
  Value: !Ref EC2Instance

EmployeeS3BucketName:
  Value: !Ref EmployeeS3Bucket

DynamoDBTableName:
  Value: !Ref EmployeeDynamoDBTable

SNSTopicARN:
  Value: !Ref EmployeeSNSTopic
```

Now need to check the all things

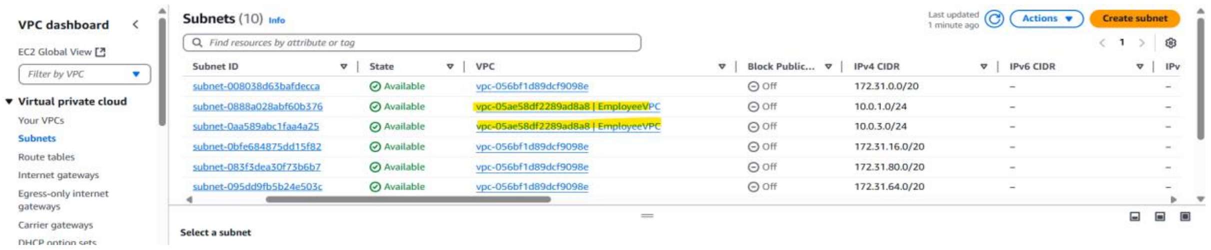




Your VPCs (2) Info

	Name	VPC ID	State	Block Public...	IPv4 CIDR	IPv6 CIDR	DHCP option se
☐	EmployeeVPC	vpc-05ae58df2289ad8a8	Available	Off	10.0.0.0/16	–	dopt-00251e3e
☐	-	vpc-056bf1d89dcf9098e	Available	Off	172.31.0.0/16	–	dopt-00251e3e

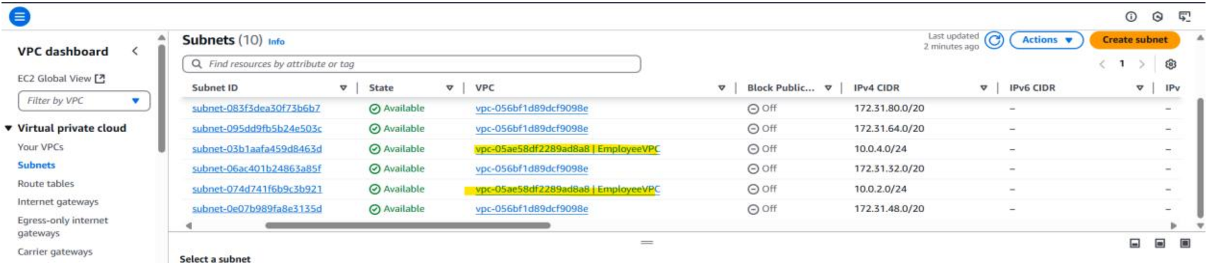
Subnets



Subnets (10) Info

Subnet ID	State	VPC	Block Public...	IPv4 CIDR	IPv6 CIDR	IPv
subnet-008038d63bafdecca	Available	vpc-056bf1d89dcf9098e	Off	172.31.0.0/20	–	–
subnet-0888a028abf60b376	Available	vpc-05ae58df2289ad8a8 | EmployeeVPC	Off	10.0.1.0/24	–	–
subnet-0aa589abc1faa4a25	Available	vpc-05ae58df2289ad8a8 | EmployeeVPC	Off	10.0.3.0/24	–	–
subnet-0bfe684875dd15f82	Available	vpc-056bf1d89dcf9098e	Off	172.31.16.0/20	–	–
subnet-083f3dea30f73b6b7	Available	vpc-056bf1d89dcf9098e	Off	172.31.80.0/20	–	–
subnet-095dd9fb5b24e503c	Available	vpc-056bf1d89dcf9098e	Off	172.31.64.0/20	–	–

Select a subnet

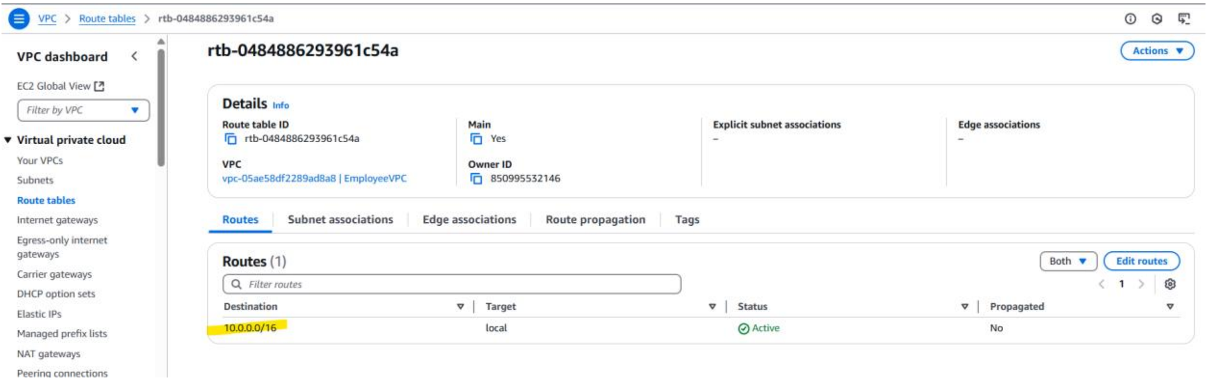


Subnets (10) Info

Subnet ID	State	VPC	Block Public...	IPv4 CIDR	IPv6 CIDR	IPv
subnet-083f3dea30f73b6b7	Available	vpc-056bf1d89dcf9098e	Off	172.31.80.0/20	–	–
subnet-095dd9fb5b24e503c	Available	vpc-056bf1d89dcf9098e	Off	172.31.64.0/20	–	–
subnet-03b1aafa459d8463d	Available	vpc-05ae58df2289ad8a8 | EmployeeVPC	Off	10.0.4.0/24	–	–
subnet-06ac401b24863a85f	Available	vpc-056bf1d89dcf9098e	Off	172.31.32.0/20	–	–
subnet-074d741f6b9c3b921	Available	vpc-05ae58df2289ad8a8 | EmployeeVPC	Off	10.0.2.0/24	–	–
subnet-0e07b989fa8e3135d	Available	vpc-056bf1d89dcf9098e	Off	172.31.48.0/20	–	–

Select a subnet

Route Tables



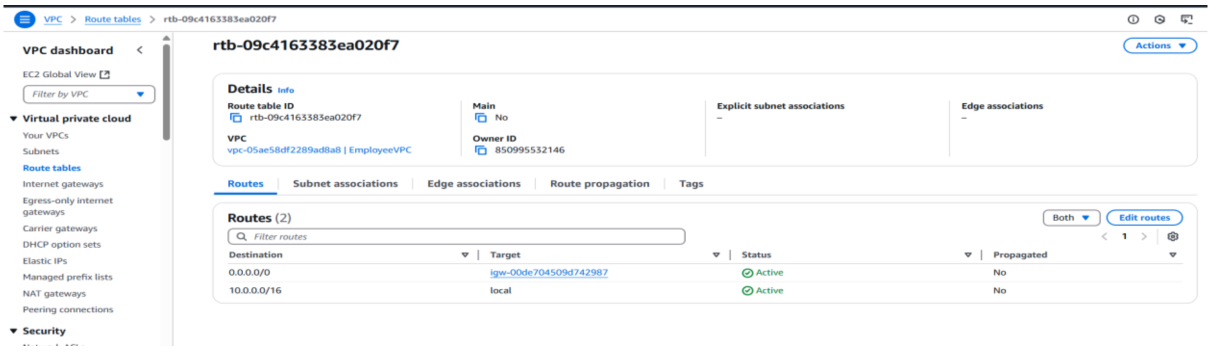
rtb-0484886293961c54a

Details Info

Route table ID: rtb-0484886293961c54a
Main: Yes
Explicit subnet associations: –
Edge associations: –
VPC: vpc-05ae58df2289ad8a8 | EmployeeVPC
Owner ID: 850995532146

Routes (1)

Destination	Target	Status	Propagated
10.0.0.0/16	local	Active	No



rtb-09c4163383ea020f7

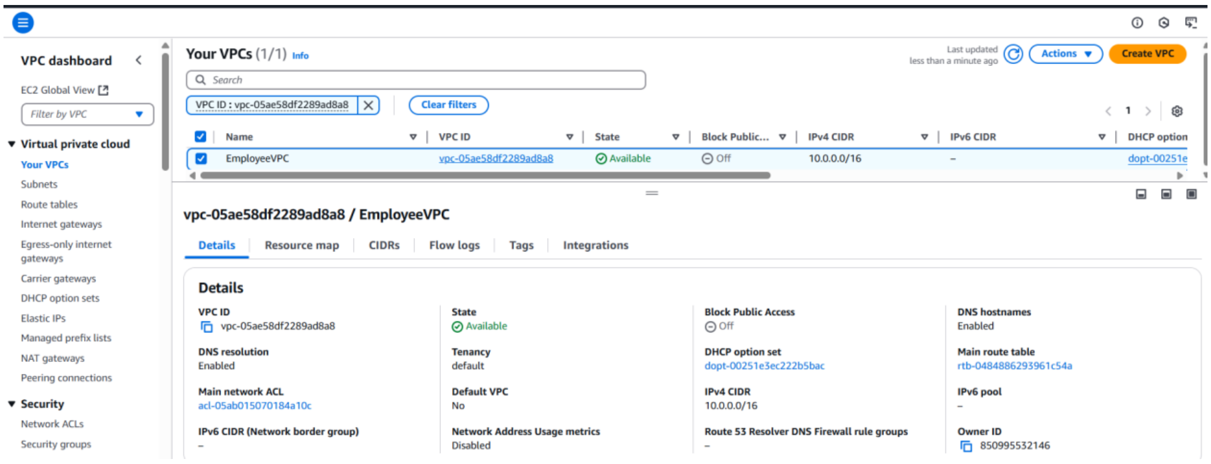
Details Info

Route table ID: rtb-09c4163383ea020f7
Main: No
Explicit subnet associations: –
Edge associations: –
VPC: vpc-05ae58df2289ad8a8 | EmployeeVPC
Owner ID: 850995532146

Routes (2)

Destination	Target	Status	Propagated
0.0.0.0/0	igw-00de704509d742987	Active	No
10.0.0.0/16	local	Active	No

Main VPC



Your VPCs (1/1) Info

VPC ID : vpc-05ae58df2289ad8a8

Name	VPC ID	State	Block Public...	IPv4 CIDR	IPv6 CIDR	DHCP option
EmployeeVPC	vpc-05ae58df2289ad8a8	Available	Off	10.0.0.0/16	–	dopt-00251e

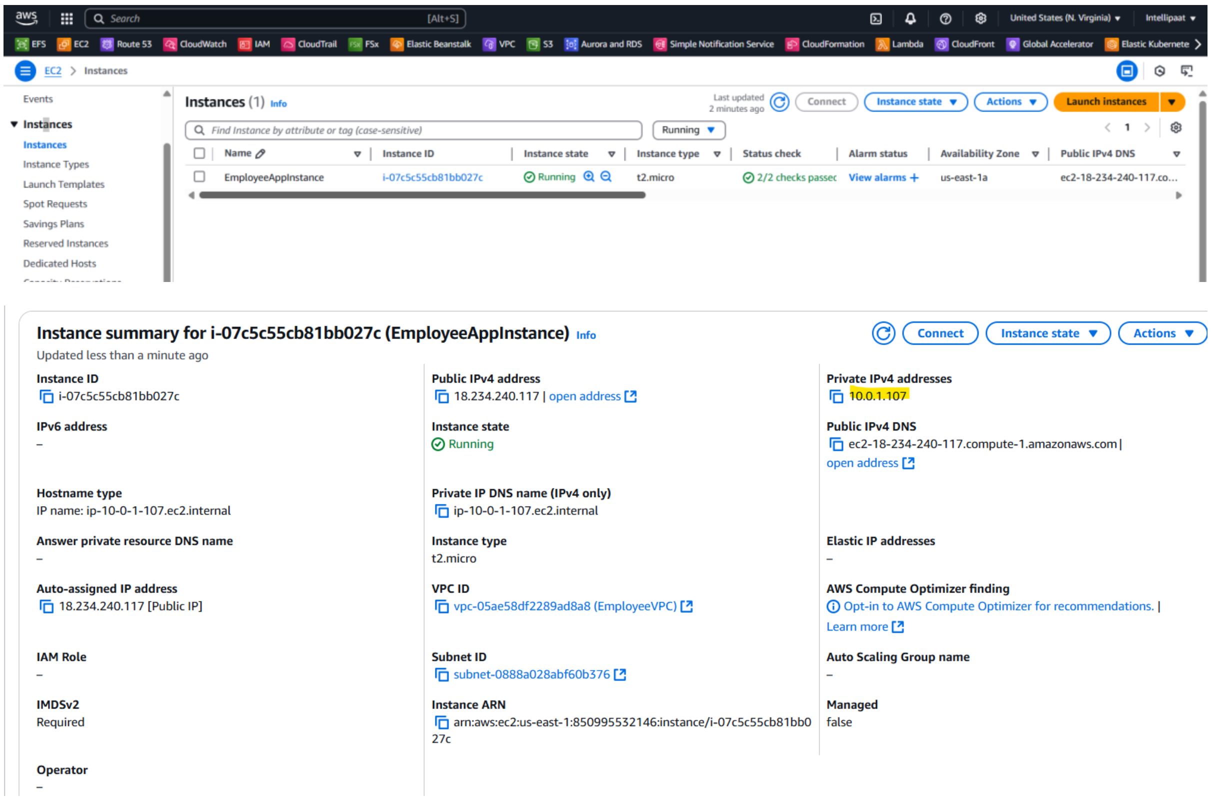
vpc-05ae58df2289ad8a8 / EmployeeVPC

Details | Resource map | CIDRs | Flow logs | Tags | Integrations

Details

VPC ID vpc-05ae58df2289ad8a8	**State** Available	**Block Public Access** Off	**DNS hostnames** Enabled
DNS resolution Enabled	**Tenancy** default	**DHCP option set** dopt-00251e3ec222b5bac	**Main route table** rtb-0484886293961c54a
Main network ACL acl-05ab015070184a10c	**Default VPC** No	**IPv4 CIDR** 10.0.0.0/16	**IPv6 pool** –
IPv6 CIDR (Network border group) –	**Network Address Usage metrics** Disabled	**Route 53 Resolver DNS Firewall rule groups** –	**Owner ID** 850995532146

In This VPC Created 1 ec2 instance in public EmployeeVPC subnet



Instances (1) Info

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
EmployeeAppInstance	i-07c5c55cb81bb027c	Running	t2.micro	2/2 checks passec	View alarms +	us-east-1a	ec2-18-234-240-117.co...

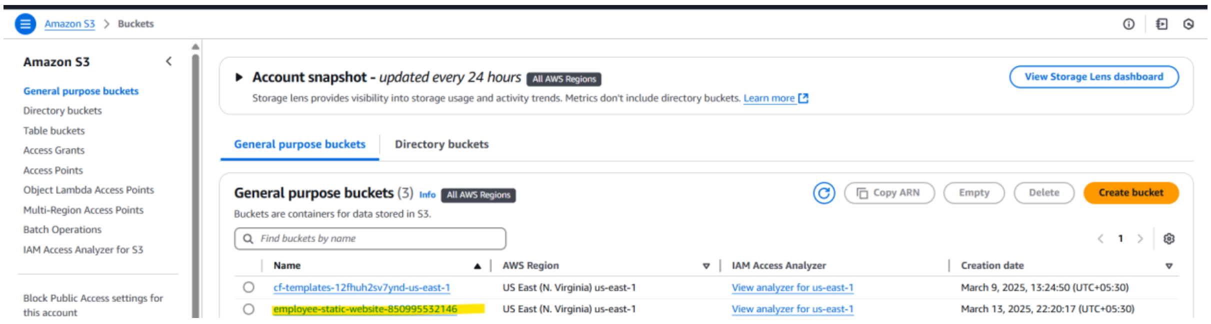
Instance summary for i-07c5c55cb81bb027c (EmployeeAppInstance) Info

Updated less than a minute ago

Connect | Instance state | Actions

Instance ID i-07c5c55cb81bb027c	**Public IPv4 address** 18.234.240.117 | open address	**Private IPv4 addresses** 10.0.1.107
IPv6 address –	**Instance state** Running	**Public IPv4 DNS** ec2-18-234-240-117.compute-1.amazonaws.com | open address
Hostname type IP name: ip-10-0-1-107.ec2.internal	**Private IP DNS name (IPv4 only)** ip-10-0-1-107.ec2.internal	
Answer private resource DNS name –	**Instance type** t2.micro	**Elastic IP addresses** –
Auto-assigned IP address 18.234.240.117 [Public IP]	**VPC ID** vpc-05ae58df2289ad8a8 (EmployeeVPC)	**AWS Compute Optimizer finding** Opt-in to AWS Compute Optimizer for recommendations. | Learn more
IAM Role –	**Subnet ID** subnet-0888a028abf60b376	**Auto Scaling Group name** –
IMDSv2 Required	**Instance ARN** arn:aws:ec2:us-east-1:850995532146:instance/i-07c5c55cb81bb027c	**Managed** false
Operator –		

CREATED S3 BUCKETS



Account snapshot - *updated every 24 hours* All AWS Regions

Storage lens provides visibility into storage usage and activity trends. Metrics don't include directory buckets. Learn more

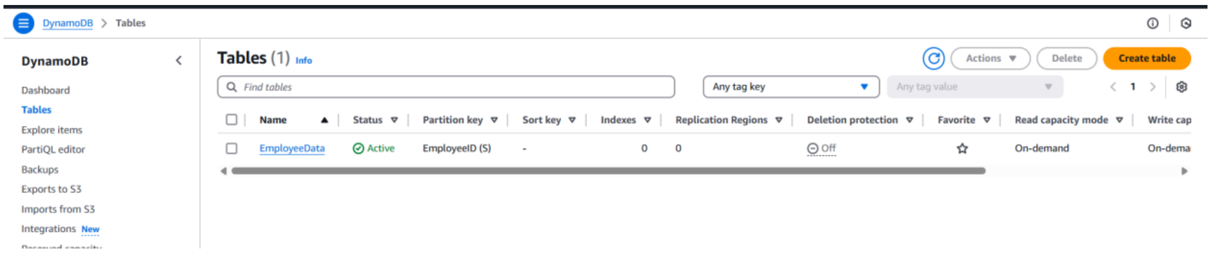
General purpose buckets | Directory buckets

General purpose buckets (3) Info All AWS Regions

Buckets are containers for data stored in S3.

Name	AWS Region	IAM Access Analyzer	Creation date
cf-templates-12fhuh2sv7ynd-us-east-1	US East (N. Virginia) us-east-1	View analyzer for us-east-1	March 9, 2025, 13:24:50 (UTC+05:30)
employee-static-website-850995532146	US East (N. Virginia) us-east-1	View analyzer for us-east-1	March 13, 2025, 22:20:17 (UTC+05:30)

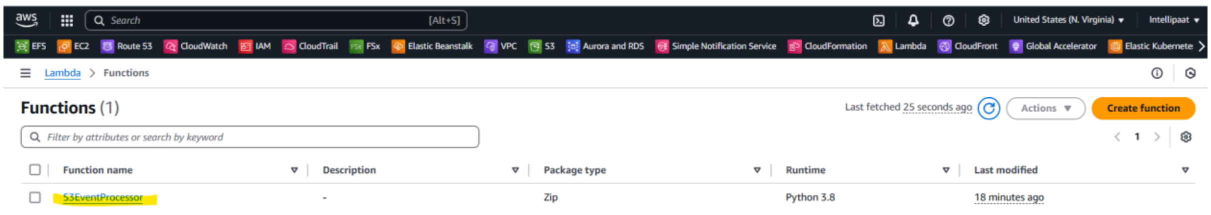
CREATED DYNAMODB



Tables (1) Info

Name	Status	Partition key	Sort key	Indexes	Replication Regions	Deletion protection	Favorite	Read capacity mode	Write cap
EmployeeData	Active	EmployeeID (S)	-	0	0	Off	☆	On-demand	On-dema

CREATED LAMBDA FUNCTION

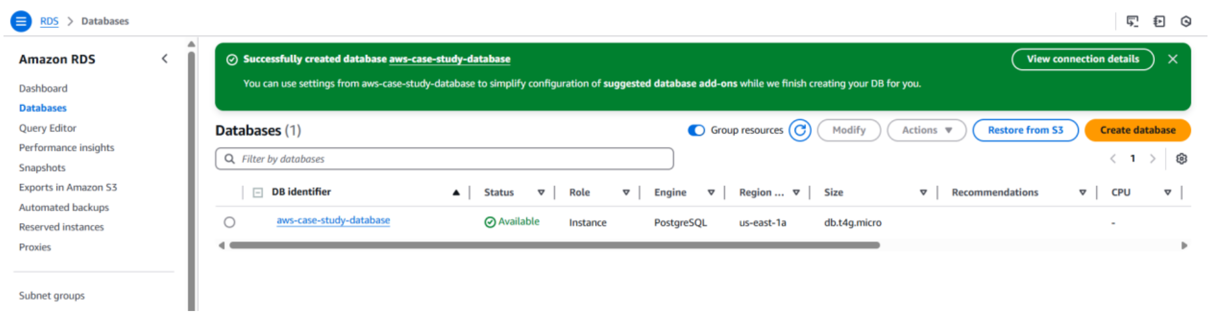


Functions (1)

Last fetched 25 seconds ago

Function name	Description	Package type	Runtime	Last modified
S3EventProcessor	-	Zip	Python 3.8	18 minutes ago

CREATED RDS DATABASE



Successfully created database aws-case-study-database

You can use settings from aws-case-study-database to simplify configuration of **suggested database add-ons** while we finish creating your DB for you.

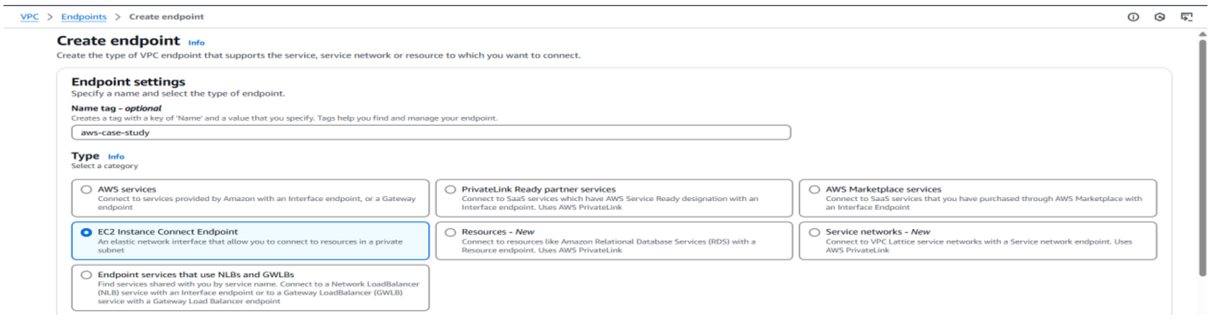
Databases (1)

DB identifier	Status	Role	Engine	Region ...	Size	Recommendations	CPU
aws-case-study-database	Available	Instance	PostgreSQL	us-east-1a	db.t4g.micro		-

Step — 2:- Connect to AWS EC2 instance in private subnet and get internet via Nat-gateway.

Additionally, create a VPC endpoint for connecting with private ec2 instance

Created VPC Endpoints in EmployeeVPC



VPC > Endpoints > Create endpoint

Create endpoint Info

Create the type of VPC endpoint that supports the service, service network or resource to which you want to connect.

Endpoint settings

Specify a name and select the type of endpoint.

Name tag - *optional*

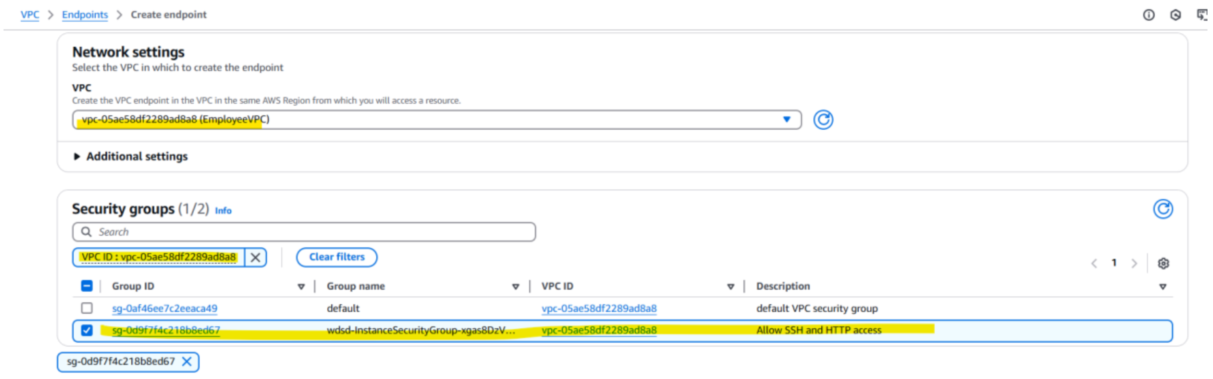
Creates a tag with a key of 'Name' and a value that you specify. Tags help you find and manage your endpoint.

aws-case-study

Type Info

Select a category

- AWS services — Connect to services provided by Amazon with an Interface endpoint, or a Gateway endpoint
- PrivateLink Ready partner services — Connect to SaaS services which have AWS Service Ready designation with an Interface endpoint. Uses AWS PrivateLink
- AWS Marketplace services — Connect to SaaS services that you have purchased through AWS Marketplace with an Interface Endpoint
- [✓ EC2 Instance Connect Endpoint] — An elastic network interface that allow you to connect to resources in a private subnet
- Resources - *New* — Connect to resources like Amazon Relational Database Services (RDS) with a Resource endpoint. Uses AWS PrivateLink
- Service networks - *New* — Connect to VPC Lattice service networks with a Service network endpoint. Uses AWS PrivateLink
- Endpoint services that use NLBs and GWLBs — Find services shared with you by service name. Connect to a Network LoadBalancer (NLB) service with an Interface endpoint or to a Gateway LoadBalancer (GWLB) service with a Gateway Load Balancer endpoint



VPC > Endpoints > Create endpoint

Network settings

Select the VPC in which to create the endpoint

VPC

Create the VPC endpoint in the VPC in the same AWS Region from which you will access a resource.

vpc-05ae58df2289ad8a8 (EmployeeVPC)

▸ Additional settings

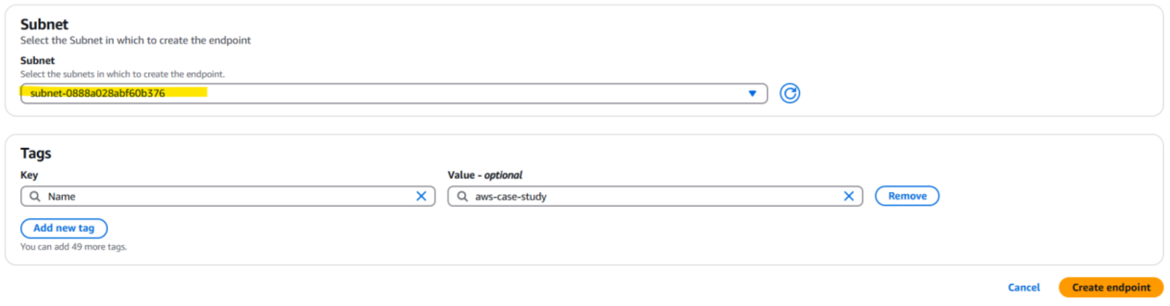
Security groups (1/2) Info

Search

VPC ID : vpc-05ae58df2289ad8a8 | Clear filters

	Group ID	Group name	VPC ID	Description
☐	sg-0af46ee7c2eeaca49	default	vpc-05ae58df2289ad8a8	default VPC security group
☑	sg-0d9f7f4c218b8ed67	wdsd-InstanceSecurityGroup-xgas8DzV...	vpc-05ae58df2289ad8a8	Allow SSH and HTTP access

sg-0d9f7f4c218b8ed67



Subnet

Select the Subnet in which to create the endpoint

Subnet

Select the subnets in which to create the endpoint.

subnet-0888a028abf60b376

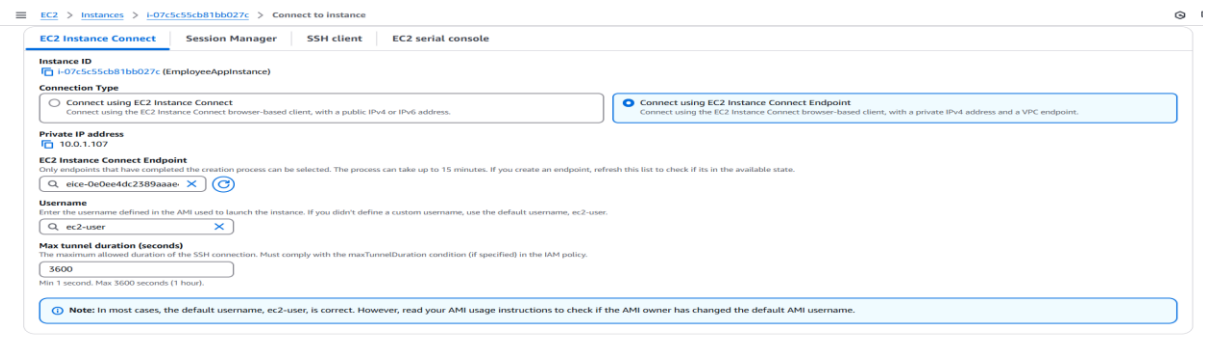
Tags

Key	Value - *optional*
Name	aws-case-study

Add new tag

You can add 49 more tags.

Then Click Create endpoint.



EC2 > Instances > i-07c5c55cb81bb027c > Connect to instance

Instance ID

i-07c5c55cb81bb027c (EmployeeAppInstance)

Connection Type

- Connect using EC2 Instance Connect — Connect using the EC2 Instance Connect browser-based client, with a public IPv4 or IPv6 address.
- [✓ Connect using EC2 Instance Connect Endpoint] — Connect using the EC2 Instance Connect browser-based client, with a private IPv4 address and a VPC endpoint.

Private IP address

10.0.1.107

EC2 Instance Connect Endpoint

Only endpoints that have completed the creation process can be selected. The process can take up to 15 minutes. If you create an endpoint, refresh this list to check if its in the available state.

eice-0e0ee4dc2389aaae

Username

Enter the username defined in the AMI used to launch the instance. If you didn't define a custom username, use the default username, ec2-user.

ec2-user

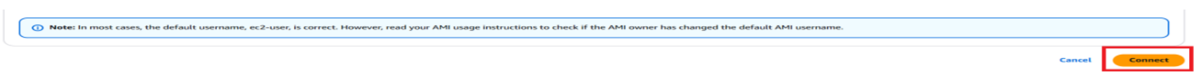
Max tunnel duration (seconds)

The maximum allowed duration of the SSH connection. Must comply with the maxTunnelDuration condition (if specified) in the IAM policy.

3600

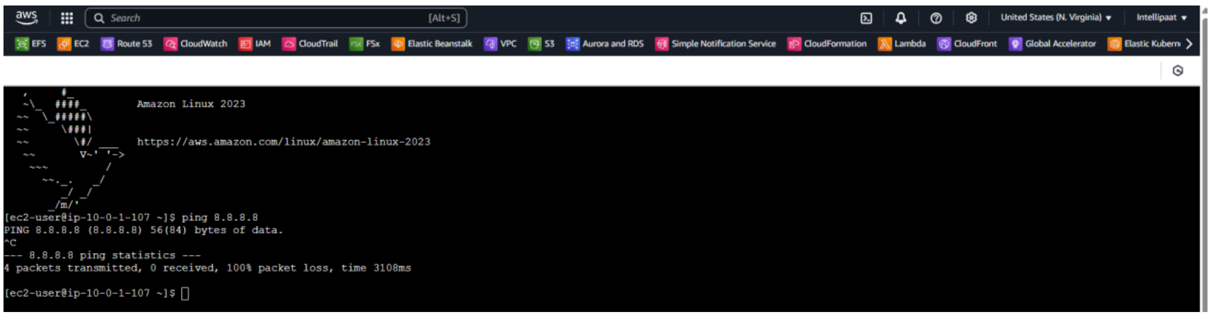
Min 1 second. Max 3600 seconds (1 hour).

Note: In most cases, the default username, ec2-user, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.



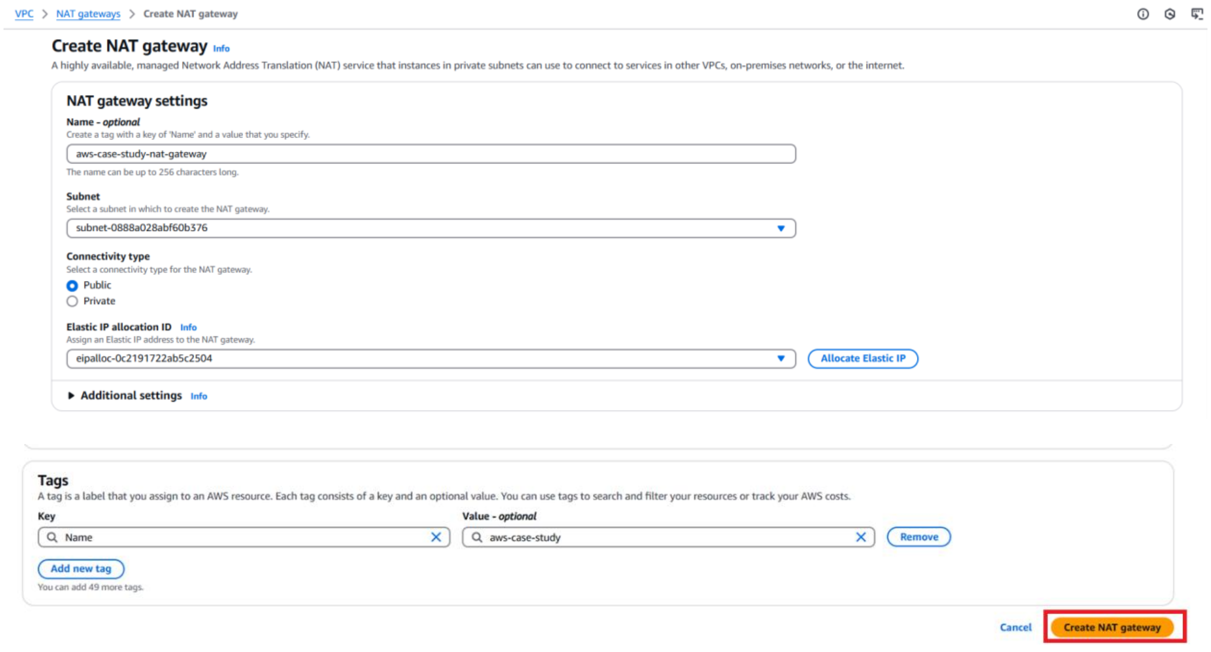
Note: In most cases, the default username, ec2-user, is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

After Endpoint creation need to connect the endpoint through ec2 instance


Search [Alt+S]
United States (N. Virginia)
Intellipaat
EFS EC2 Route 53 CloudWatch IAM CloudTrail FSx Elastic Beanstalk VPC S3 Aurora and RDS Simple Notification Service CloudFormation Lambda CloudFront Global Accelerator Elastic Kubern

```
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-10-0-1-107 ~]$ ping 8.8.8.8
PING 8.8.8.8 (8.8.8.8) 56(84) bytes of data.
^C
--- 8.8.8.8 ping statistics ---
4 packets transmitted, 0 received, 100% packet loss, time 3108ms

[ec2-user@ip-10-0-1-107 ~]$ 
```

Created NAT Gateway



Create NAT gateway Info

A highly available, managed Network Address Translation (NAT) service that instances in private subnets can use to connect to services in other VPCs, on-premises networks, or the internet.

NAT gateway settings

Name - *optional*
Create a tag with a key of 'Name' and a value that you specify.
aws-case-study-nat-gateway
The name can be up to 256 characters long.

Subnet
Select a subnet in which to create the NAT gateway.
subnet-0888a028abf60b376

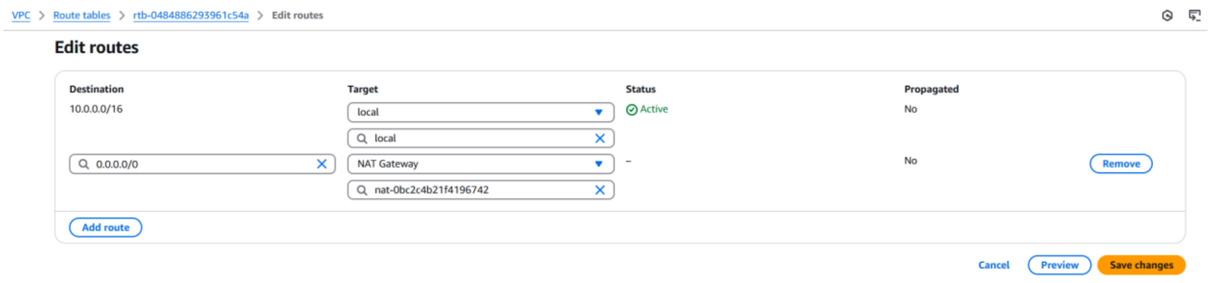
Connectivity type
Select a connectivity type for the NAT gateway.
[✓ Public]
Private

Elastic IP allocation ID Info
Assign an Elastic IP address to the NAT gateway.
eipalloc-0c2191722ab5c2504

Tags
A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

Key	Value - *optional*
Name	aws-case-study

You can add 49 more tags.

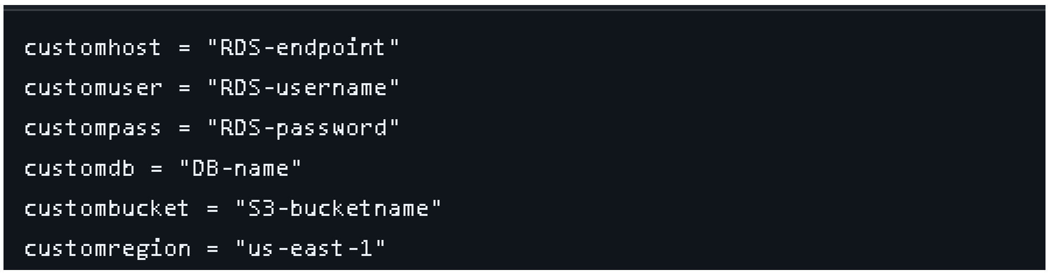


Edit routes

Destination	Target	Status	Propagated
10.0.0.0/16	local	Active	No
0.0.0.0/0	NAT Gateway nat-0bc2c4b21f4196742	-	No

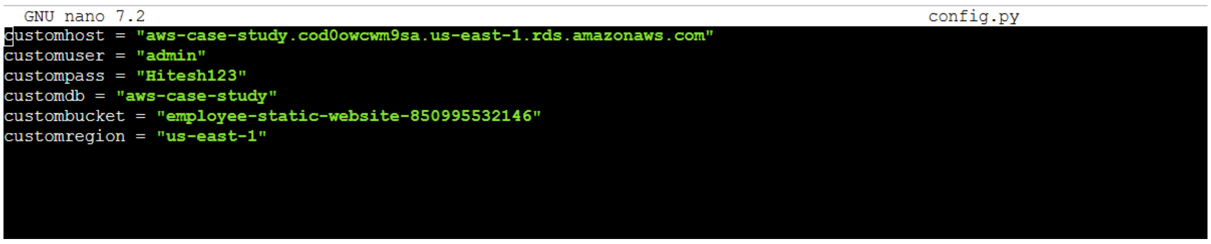
Step — 3: Next, go to the aws-code-main directory and list out all files by using the simple command ls

Change all requirements of config.py and EmpApp.py files by using the sudo nano config.py and sudo nano EmpApp.py commands from all aws resources created in step -1 and Create S3 bucket, specify your region like us-east-1.



```
customhost = "RDS-endpoint"
customuser = "RDS-username"
custompass = "RDS-password"
customdb = "DB-name"
custombucket = "S3-bucketname"
customregion = "us-east-1"
```

```
customhost = "aws-case-study.cod0owcwm9sa.us-east-1.rds.amazonaws.com"

customuser = "admin"

custompass = "Hitesh123"

customdb = "aws-case-study"

custombucket = "employee-static-website-850995532146"

customregion = "us-east-1
```



```
  GNU nano 7.2                                                                     config.py
customhost = "aws-case-study.cod0owcwm9sa.us-east-1.rds.amazonaws.com"
customuser = "admin"
custompass = "Hitesh123"
customdb = "aws-case-study"
custombucket = "employee-static-website-850995532146"
customregion = "us-east-1"
```

EmpApp.py file

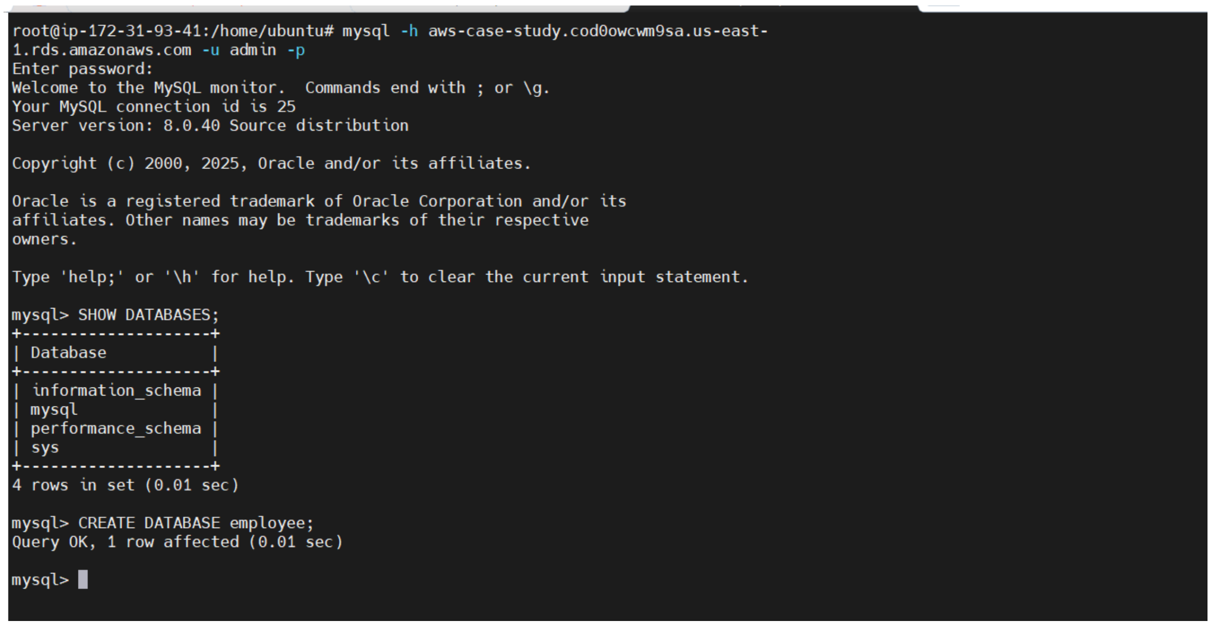
```
from flask import Flask, render_template, request

from pymysql import connections

import os

import boto3

from config import *


app = Flask(__name__)


# DBHOST = os.environ.get("DBHOST")

# DBPORT = os.environ.get("DBPORT")
# DBPORT = int(DBPORT)

# DBUSER = os.environ.get("DBUSER")

# DBPWD = os.environ.get("DBPWD")

# DATABASE = os.environ.get("DATABASE")


bucket= "employee-static-website-850995532146"

region= "us-east-1"


db_conn = connections.Connection(

   host= "aws-case-study-database.cod0owcwm9sa.us-east-1.rds.amazonaws.com",

   port=3306,

   user= "admin",

   password= "Hitesh123",

   db= "aws-case-study"


)

output = {}

table = 'employee';


@app.route("/", methods=['GET', 'POST'])

def home():

   return render_template('AddEmp.html')


@app.route("/about", methods=['POST'])

def about():

   return render_template('www.intellipaat.com');

@app.route("/addemp", methods=['POST'])

def AddEmp():

   emp_id = request.form['emp_id']

   first_name = request.form['first_name']

   last_name = request.form['last_name']
```

```
pri_skill = request.form['pri_skill']

location = request.form['location']

emp_image_file = request.files['emp_image_file']


insert_sql = "INSERT INTO employee VALUES (%s, %s, %s, %s, %s)"

cursor = db_conn.cursor()


if emp_image_file.filename == "":

  return "Please select a file"


try:


  cursor.execute(insert_sql,(emp_id, first_name, last_name, pri_skill, location))

  db_conn.commit()

  emp_name = "" + first_name + " " + last_name

  # Uplaod image file in S3 #

  emp_image_file_name_in_s3 = "emp-id-"+str(emp_id) + "_image_file"

  s3 = boto3.resource('s3')


   try:

    print("Data inserted in MySQL RDS... uploading image to S3...")

    s3.Bucket(custombucket).put_object(Key=emp_image_file_name_in_s3,
Body=emp_image_file)

    bucket_location = boto3.client('s3').get_bucket_location(Bucket=custombucket)

    s3_location = (bucket_location['LocationConstraint'])


    if s3_location is None:

      s3_location = ''

    else:

      s3_location = '-' + s3_location
```

```
        object_url = "https://s3{0}.amazonaws.com/{1}/{2}".format(
            s3_location,
            custombucket,
            emp_image_file_name_in_s3)

        # Save image file metadata in DynamoDB #
        print("Uploading to S3 success... saving metadata in dynamodb...")



        try:
            dynamodb_client = boto3.client('dynamodb', region_name='us-east-1')
            dynamodb_client.put_item(
             TableName='EmployeeData',
                Item={
                 'empid': {
                     'N': emp_id
                  },
                  'image_url': {
                        'S': object_url
                    }
                }
            )

        except Exception as e:
            program_msg = "Flask could not update DynamoDB table with S3 object URL"
            return str(e)

    except Exception as e:
        return str(e)

finally:
        cursor.close()

    print("all modification done...")
    return render_template('AddEmpOutput.html', name=emp_name)

@app.route("/getemp", methods=['GET', 'POST'])
def GetEmp():
    return render_template("GetEmp.html")



@app.route("/fetchdata", methods=['GET','POST'])
def FetchData():
    emp_id = request.form['emp_id']

    output = {}
    select_sql = "SELECT emp_id, first_name, last_name, pri_skill, location from employee where
emp_id=%s"
    cursor = db_conn.cursor()

    try:
        cursor.execute(select_sql,(emp_id))
        result = cursor.fetchone()

        output["emp_id"] = result[0]
        print('EVERYTHING IS FINE TILL HERE')
        output["first_name"] = result[1]
        output["last_name"] = result[2]
        output["primary_skills"] = result[3]
        output["location"] = result[4]
        print(output["emp_id"])
        dynamodb_client = boto3.client('dynamodb', region_name=customregion)
        try:
            response = dynamodb_client.get_item(
                TableName='EmployeeData',
                Key={
                    'empid': {
                        'N': str(emp_id)
                    }
                }
            )
            image_url = response['Item']['image_url']['S']

        except Exception as e:
            program_msg = "Flask could not update DynamoDB table with S3 object URL"
            return render_template('addemperror.html', errmsg1=program_msg, errmsg2=e)

    except Exception as e:
        print(e)

    finally:
        cursor.close()

    return render_template("GetEmpOutput.html", id=output["emp_id"],
fname=output["first_name"],
                           lname=output["last_name"], interest=output["primary_skills"],
location=output["location"],
                           image_url=image_url)

if __name__ == '__main__':
    app.run(host='0.0.0.0',port=80,debug=True)
```

Save all changes in the Ec2 instance and proceed with the Next Steps.

Before We need to create the database



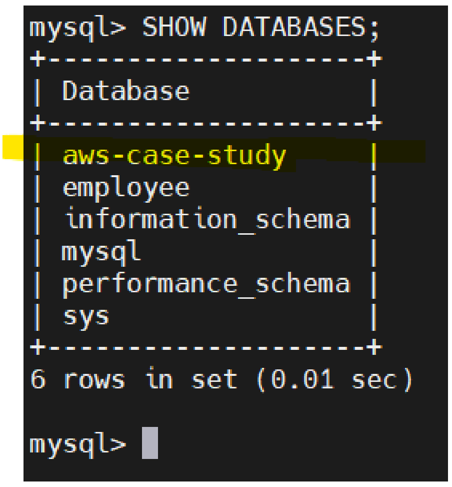
```
root@ip-172-31-93-41:/home/ubuntu# mysql -h aws-case-study.cod0owcwm9sa.us-east-
1.rds.amazonaws.com -u admin -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 25
Server version: 8.0.40 Source distribution

Copyright (c) 2000, 2025, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
4 rows in set (0.01 sec)

mysql> CREATE DATABASE employee;
Query OK, 1 row affected (0.01 sec)

mysql> 
```

CREATE DATABASES aws-case-study;

Then Type **SHOW DATABASES;**

Then Create Tables name is employee;

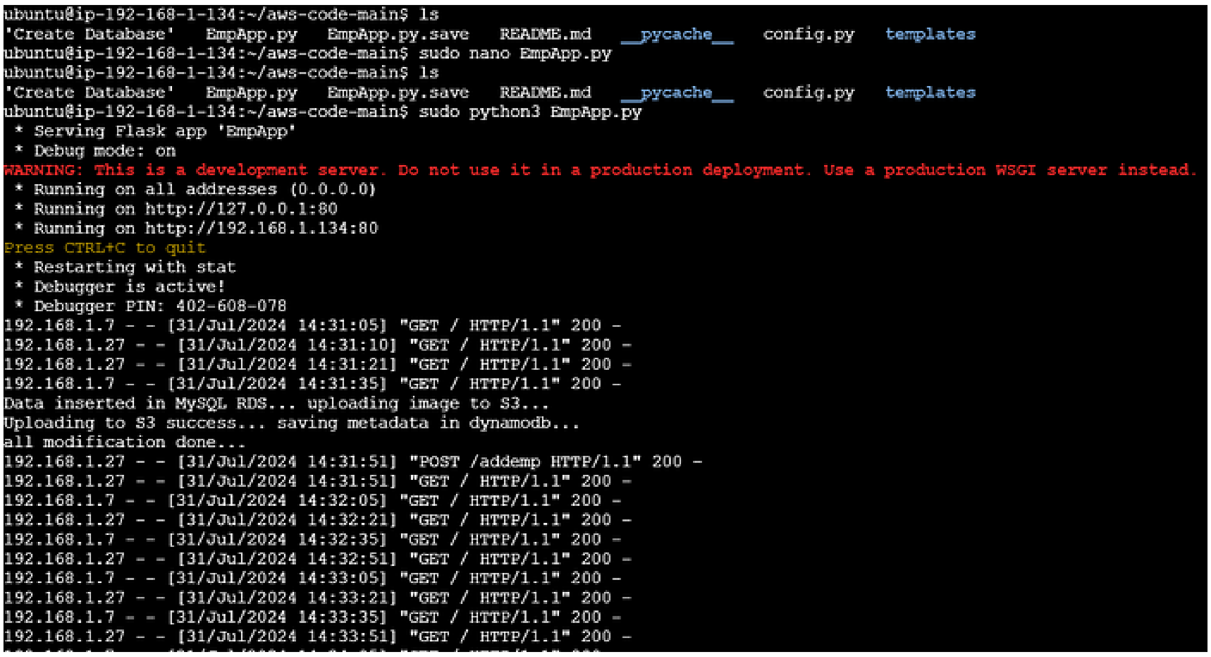
CREATE TABLE employee (empid VARCHAR (20), first_name VARCHAR (20), last_name VARCHAR (20), primary_skills VARCHAR (20), location VARCHAR (20));



```
mysql> SHOW DATABASES;
+--------------------+
| Database           |
+--------------------+
| aws-case-study     |
| employee           |
| information_schema |
| mysql              |
| performance_schema |
| sys                |
+--------------------+
6 rows in set (0.01 sec)

mysql> 
```

Step — 5: Run python files with sudo python3 EmpApp.py command from the flask web application and check whether it works or not.

Instance profile that has to be attached to the EC2 instance being launched. The instance profile should have permission to access RDS, DynamoDB, and S3 Bucket. (This will be there in the cloud formation script)



```
ubuntu@ip-192-168-1-134:~/aws-code-main$ ls
'Create Database'   EmpApp.py   EmpApp.py.save   README.md   __pycache__   config.py   templates
ubuntu@ip-192-168-1-134:~/aws-code-main$ sudo nano EmpApp.py
ubuntu@ip-192-168-1-134:~/aws-code-main$ ls
'Create Database'   EmpApp.py   EmpApp.py.save   README.md   __pycache__   config.py   templates
ubuntu@ip-192-168-1-134:~/aws-code-main$ sudo python3 EmpApp.py
 * Serving Flask app 'EmpApp'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:80
 * Running on http://192.168.1.134:80
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 402-608-078
192.168.1.7 - - [31/Jul/2024 14:31:05] "GET / HTTP/1.1" 200 -
192.168.1.27 - - [31/Jul/2024 14:31:10] "GET / HTTP/1.1" 200 -
192.168.1.27 - - [31/Jul/2024 14:31:21] "GET / HTTP/1.1" 200 -
192.168.1.7 - - [31/Jul/2024 14:31:35] "GET / HTTP/1.1" 200 -
Data inserted in MySQL RDS... uploading image to S3...
Uploading to S3 success... saving metadata in dynamodb...
all modification done...
192.168.1.27 - - [31/Jul/2024 14:31:51] "POST /addemp HTTP/1.1" 200 -
192.168.1.27 - - [31/Jul/2024 14:31:51] "GET / HTTP/1.1" 200 -
192.168.1.7 - - [31/Jul/2024 14:32:05] "GET / HTTP/1.1" 200 -
192.168.1.27 - - [31/Jul/2024 14:32:21] "GET / HTTP/1.1" 200 -
192.168.1.7 - - [31/Jul/2024 14:32:35] "GET / HTTP/1.1" 200 -
192.168.1.27 - - [31/Jul/2024 14:32:51] "GET / HTTP/1.1" 200 -
192.168.1.7 - - [31/Jul/2024 14:33:05] "GET / HTTP/1.1" 200 -
192.168.1.27 - - [31/Jul/2024 14:33:21] "GET / HTTP/1.1" 200 -
192.168.1.7 - - [31/Jul/2024 14:33:35] "GET / HTTP/1.1" 200 -
192.168.1.27 - - [31/Jul/2024 14:33:51] "GET / HTTP/1.1" 200 -
```

Now All this done you need to run the command

python3 EmpApp.py

If You any getting error you need to install the dependencies of python.

sudo apt-get update -y

sudo apt-get install python3 -y

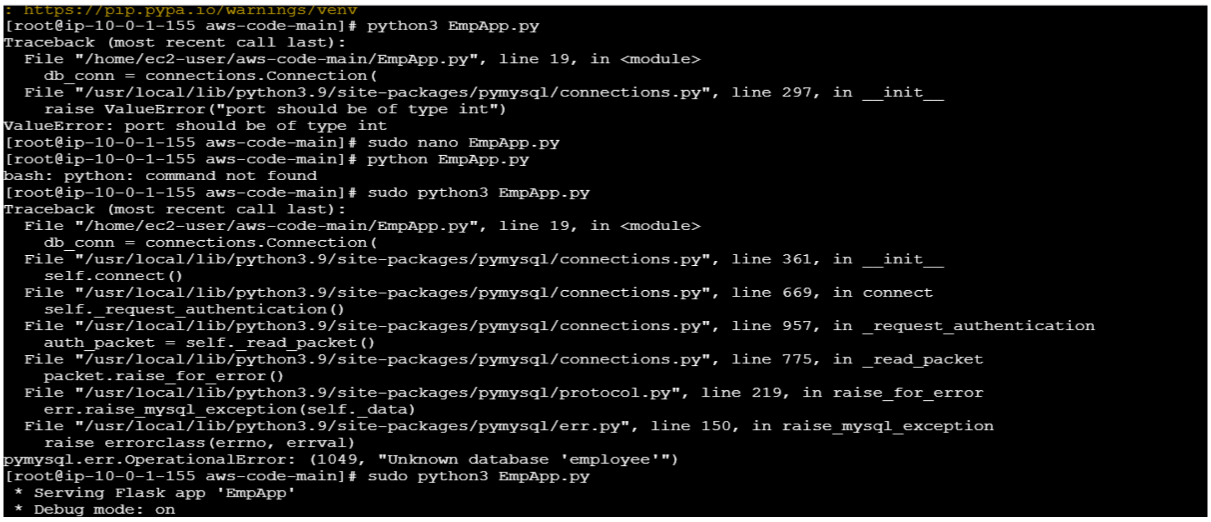
sudo apt-get install python3-boto3 -y

sudo apt-get install python3-flask -y

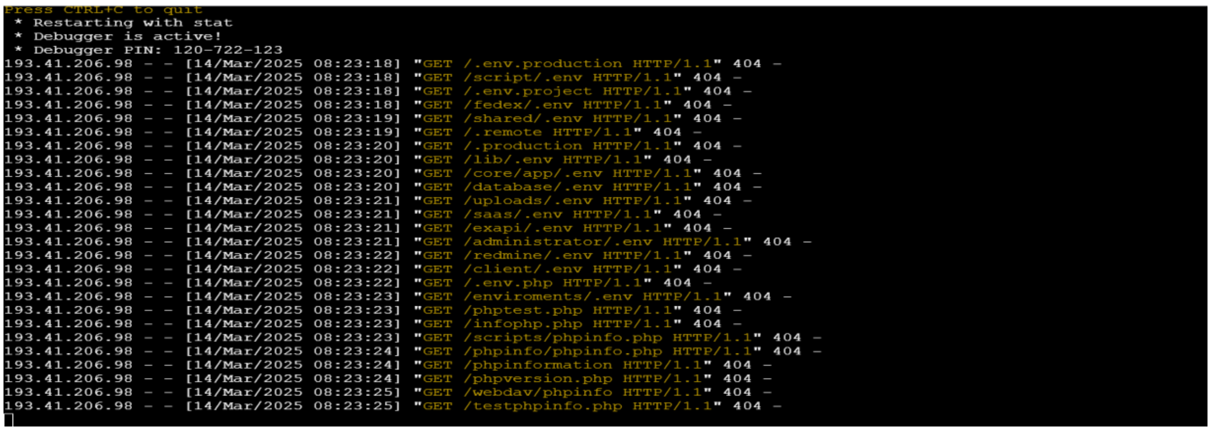
sudo apt-get install mysql-server -y

sudo apt-get install python3-pymysql -y

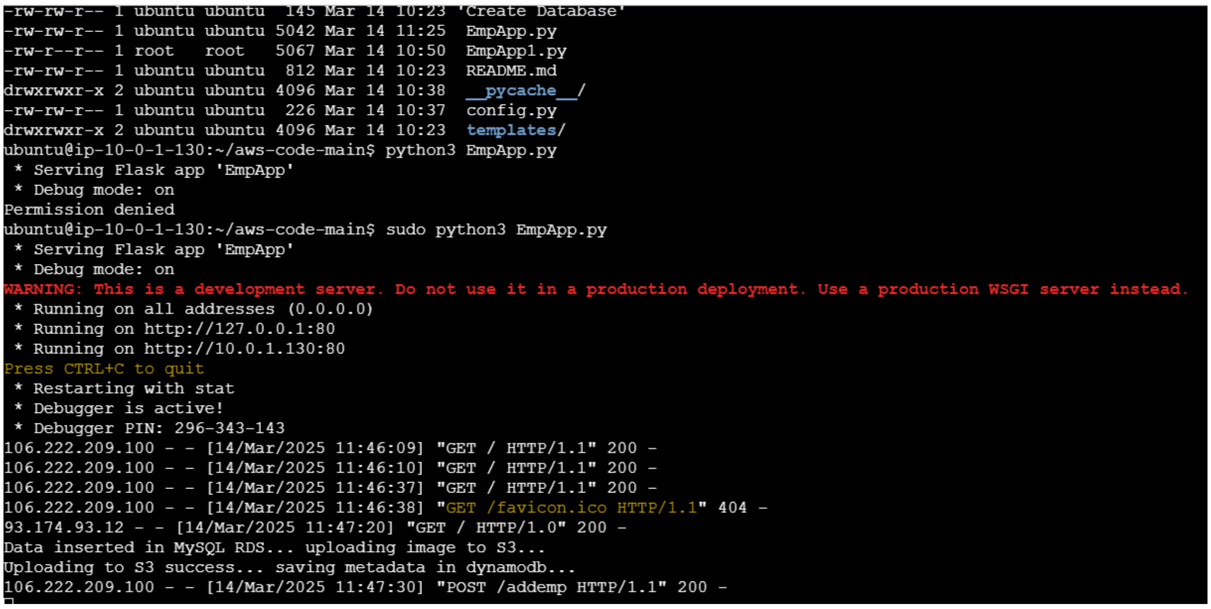
git clone https://github.com/hiteshchauhan89/aws-code-main.git



```
: https://pip.pypa.io/warnings/venv
[root@ip-10-0-1-155 aws-code-main]# python3 EmpApp.py
Traceback (most recent call last):
  File "/home/ec2-user/aws-code-main/EmpApp.py", line 19, in <module>
    db_conn = connections.Connection(
  File "/usr/local/lib/python3.9/site-packages/pymysql/connections.py", line 297, in __init__
    raise ValueError("port should be of type int")
ValueError: port should be of type int
[root@ip-10-0-1-155 aws-code-main]# sudo nano EmpApp.py
[root@ip-10-0-1-155 aws-code-main]# python EmpApp.py
bash: python: command not found
[root@ip-10-0-1-155 aws-code-main]# sudo python3 EmpApp.py
Traceback (most recent call last):
  File "/home/ec2-user/aws-code-main/EmpApp.py", line 19, in <module>
    db_conn = connections.Connection(
  File "/usr/local/lib/python3.9/site-packages/pymysql/connections.py", line 361, in __init__
    self.connect()
  File "/usr/local/lib/python3.9/site-packages/pymysql/connections.py", line 669, in connect
    self._request_authentication()
  File "/usr/local/lib/python3.9/site-packages/pymysql/connections.py", line 957, in _request_authentication
    auth_packet = self._read_packet()
  File "/usr/local/lib/python3.9/site-packages/pymysql/connections.py", line 775, in _read_packet
    packet.raise_for_error()
  File "/usr/local/lib/python3.9/site-packages/pymysql/protocol.py", line 219, in raise_for_error
    err.raise_mysql_exception(self._data)
  File "/usr/local/lib/python3.9/site-packages/pymysql/err.py", line 150, in raise_mysql_exception
    raise errorclass(errno, errval)
pymysql.err.OperationalError: (1049, "Unknown database 'employee'")
[root@ip-10-0-1-155 aws-code-main]# sudo python3 EmpApp.py
 * Serving Flask app 'EmpApp'
 * Debug mode: on
```

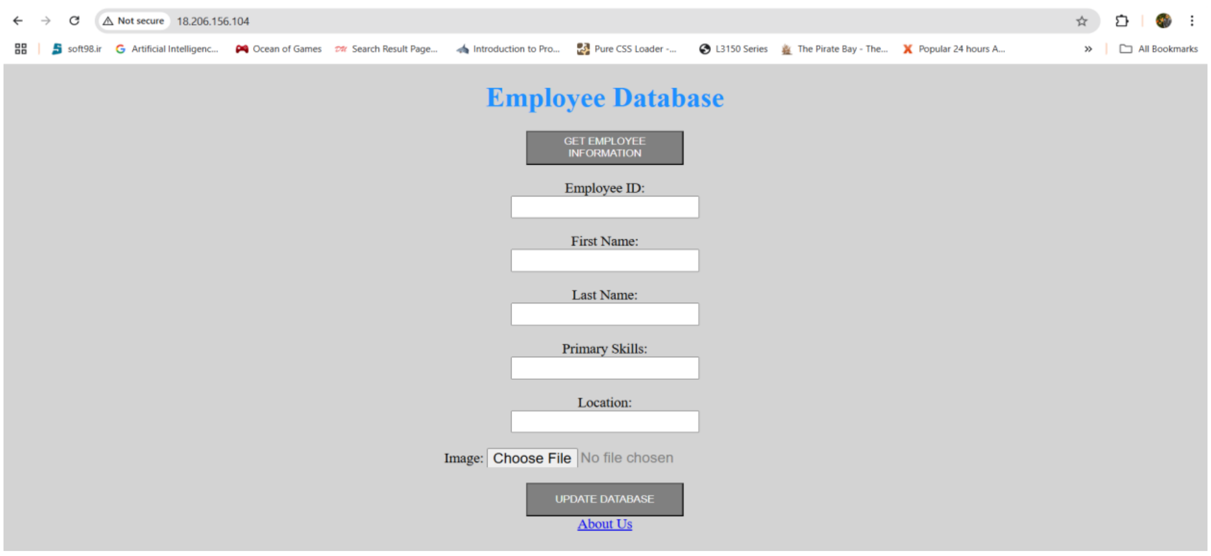


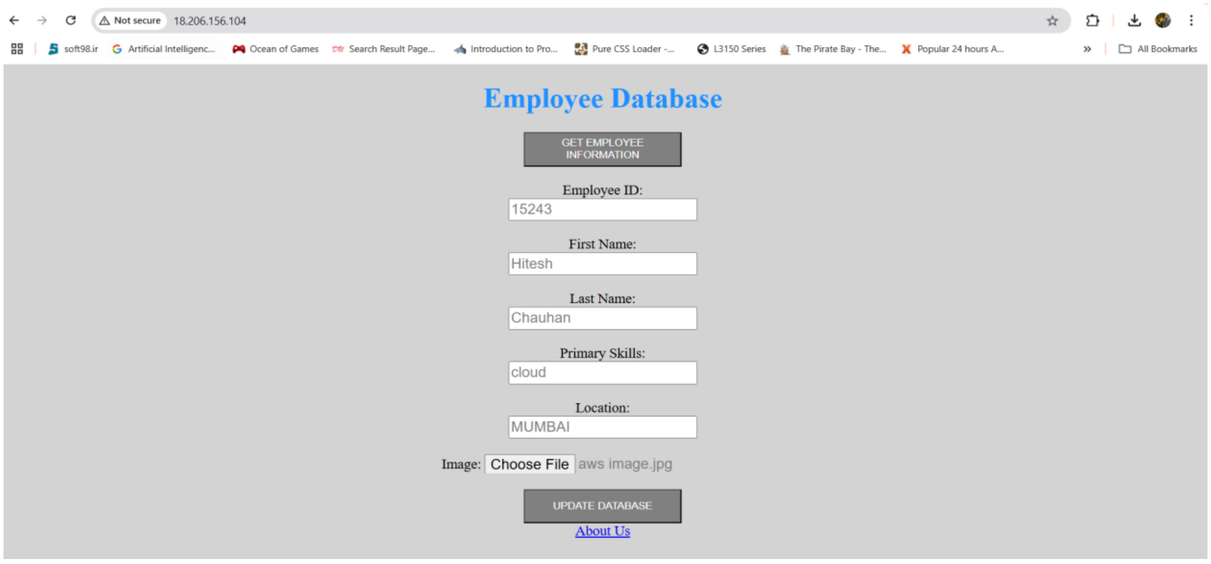
```
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 120-722-123
193.41.206.98 - - [14/Mar/2025 08:23:18] "GET /.env.production HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:18] "GET /script/.env HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:18] "GET /.env.project HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:18] "GET /fedex/.env HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:19] "GET /shared/.env HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:19] "GET /.remote HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:20] "GET /.production HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:20] "GET /lib/.env HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:20] "GET /core/app/.env HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:20] "GET /database/.env HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:21] "GET /uploads/.env HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:21] "GET /saas/.env HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:21] "GET /exapi/.env HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:21] "GET /administrator/.env HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:22] "GET /redmine/.env HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:22] "GET /client/.env HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:22] "GET /.env.php HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:23] "GET /enviroments/.env HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:23] "GET /phptest.php HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:23] "GET /infophp.php HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:23] "GET /scripts/phpinfo.php HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:24] "GET /phpinfo/phpinfo.php HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:24] "GET /phpinformation HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:24] "GET /phpversion.php HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:25] "GET /webdav/phpinfo HTTP/1.1" 404 -
193.41.206.98 - - [14/Mar/2025 08:23:25] "GET /testphpinfo.php HTTP/1.1" 404 -
```



```
-rw-rw-r-- 1 ubuntu ubuntu  145 Mar 14 10:23 'Create Database'
-rw-rw-r-- 1 ubuntu ubuntu 5042 Mar 14 11:25  EmpApp.py
-rw-r--r-- 1 root   root   5067 Mar 14 10:50  EmpApp1.py
-rw-rw-r-- 1 ubuntu ubuntu  812 Mar 14 10:23  README.md
drwxrwxr-x 2 ubuntu ubuntu 4096 Mar 14 10:38  __pycache__/
-rw-rw-r-- 1 ubuntu ubuntu  226 Mar 14 10:37  config.py
drwxrwxr-x 2 ubuntu ubuntu 4096 Mar 14 10:23  templates/
ubuntu@ip-10-0-1-130:~/aws-code-main$ python3 EmpApp.py
 * Serving Flask app 'EmpApp'
 * Debug mode: on
Permission denied
ubuntu@ip-10-0-1-130:~/aws-code-main$ sudo python3 EmpApp.py
 * Serving Flask app 'EmpApp'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on all addresses (0.0.0.0)
 * Running on http://127.0.0.1:80
 * Running on http://10.0.1.130:80
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 296-343-143
106.222.209.100 - - [14/Mar/2025 11:46:09] "GET / HTTP/1.1" 200 -
106.222.209.100 - - [14/Mar/2025 11:46:10] "GET / HTTP/1.1" 200 -
106.222.209.100 - - [14/Mar/2025 11:46:37] "GET / HTTP/1.1" 200 -
106.222.209.100 - - [14/Mar/2025 11:46:38] "GET /favicon.ico HTTP/1.1" 404 -
93.174.93.12 - - [14/Mar/2025 11:47:20] "GET / HTTP/1.0" 200 -
Data inserted in MySQL RDS... uploading image to S3...
Uploading to S3 success... saving metadata in dynamodb...
106.222.209.100 - - [14/Mar/2025 11:47:30] "POST /addemp HTTP/1.1" 200 -
```

After Things Completed copy the public ip and paste the web browser.you will see like this page.





18.206.156.104

Employee Database

Employee ID: 15243

First Name: Hitesh

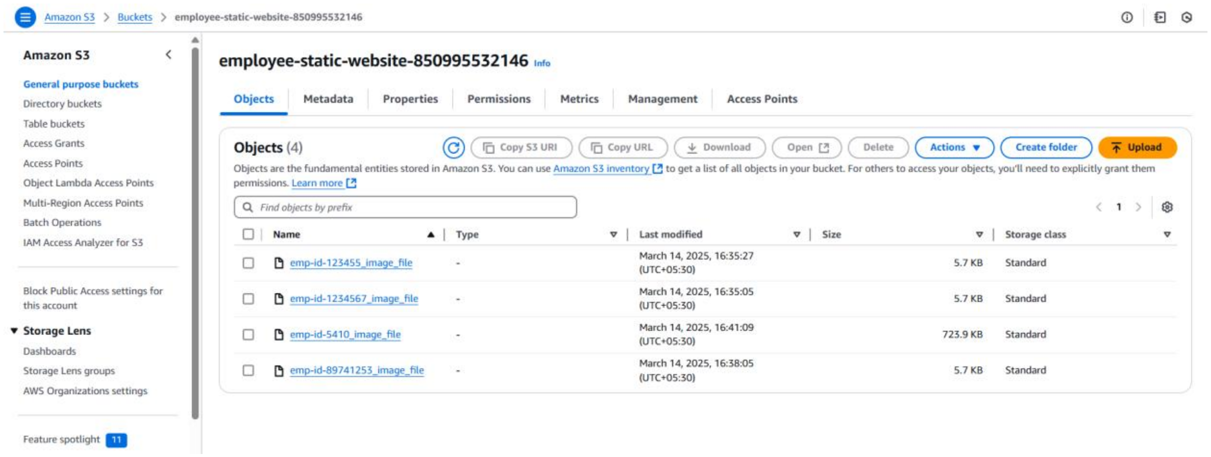
Last Name: Chauhan

Primary Skills: cloud

Location: MUMBAI

Image: Choose File aws image.jpg

This is s3 bucket.the image has been automatically uploaded from EmpApp.py scripts.



Amazon S3 > Buckets > employee-static-website-850995532146

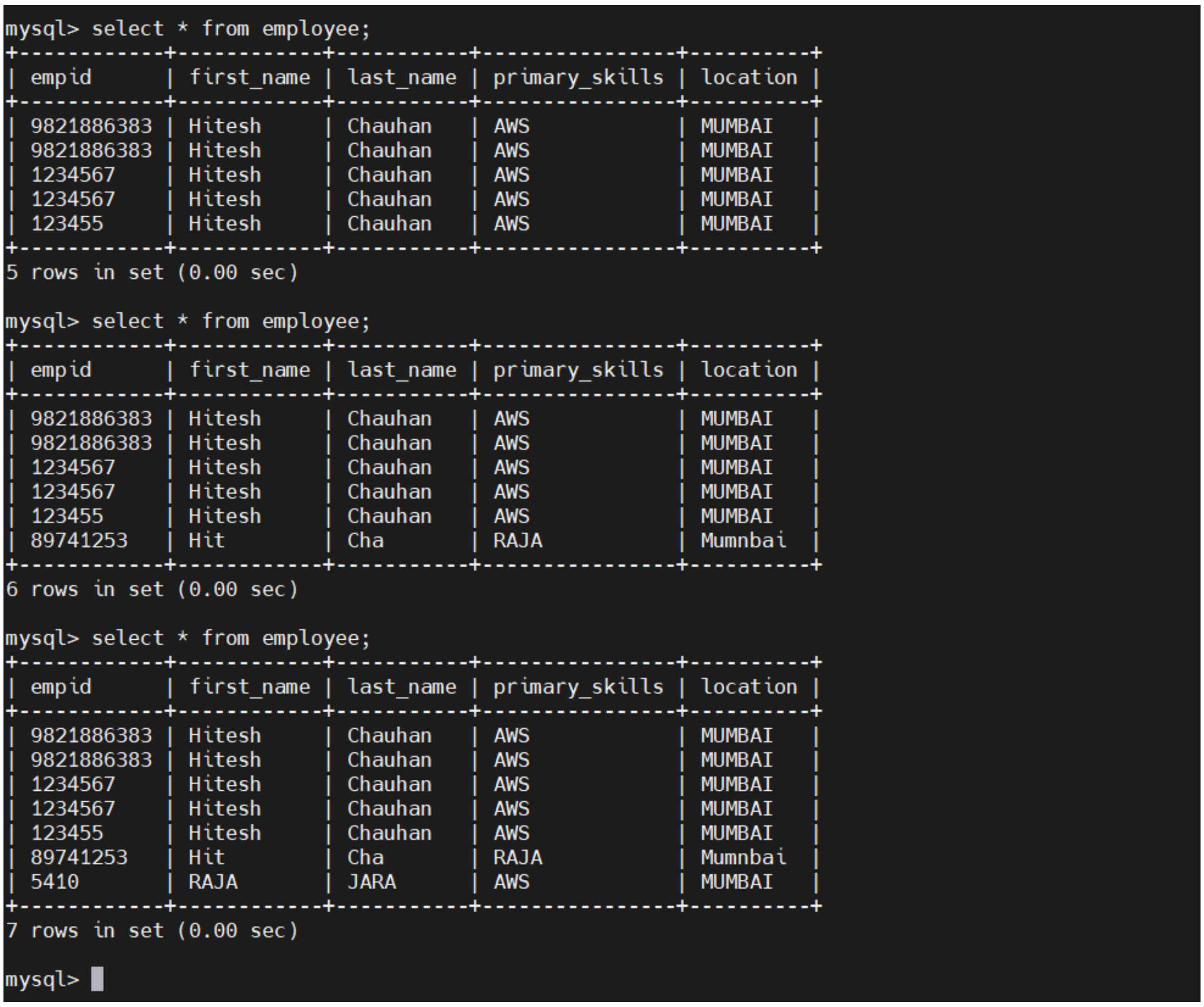
employee-static-website-850995532146

Objects (4)

Objects are the fundamental entities stored in Amazon S3. You can use Amazon S3 inventory to get a list of all objects in your bucket. For others to access your objects, you'll need to explicitly grant them permissions. Learn more

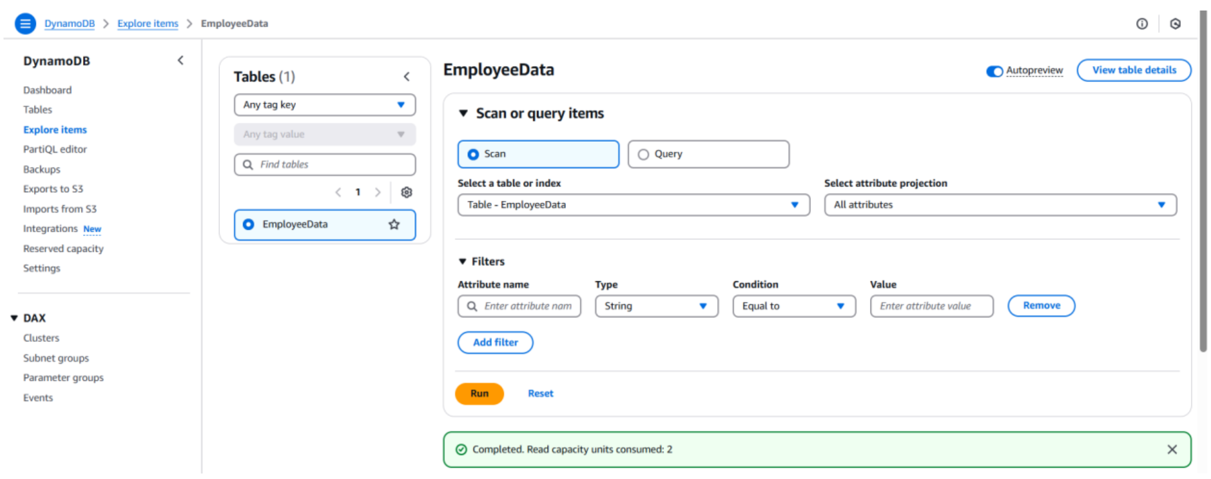
Name	Type	Last modified	Size	Storage class
emp-id-123455_image_file	-	March 14, 2025, 16:35:27 (UTC+05:30)	5.7 KB	Standard
emp-id-1234567_image_file	-	March 14, 2025, 16:35:05 (UTC+05:30)	5.7 KB	Standard
emp-id-5410_image_file	-	March 14, 2025, 16:41:09 (UTC+05:30)	723.9 KB	Standard
emp-id-89741253_image_file	-	March 14, 2025, 16:38:05 (UTC+05:30)	5.7 KB	Standard

Here fill in the data and click update Database button then navigate to our RDS instance and log in earlier exiting the MySQL database.



```
mysql> select * from employee;
+------------+------------+-----------+----------------+----------+
| empid      | first_name | last_name | primary_skills | location |
+------------+------------+-----------+----------------+----------+
| 9821886383 | Hitesh     | Chauhan   | AWS            | MUMBAI   |
| 9821886383 | Hitesh     | Chauhan   | AWS            | MUMBAI   |
| 1234567    | Hitesh     | Chauhan   | AWS            | MUMBAI   |
| 1234567    | Hitesh     | Chauhan   | AWS            | MUMBAI   |
| 123455     | Hitesh     | Chauhan   | AWS            | MUMBAI   |
+------------+------------+-----------+----------------+----------+
5 rows in set (0.00 sec)

mysql> select * from employee;
+------------+------------+-----------+----------------+----------+
| empid      | first_name | last_name | primary_skills | location |
+------------+------------+-----------+----------------+----------+
| 9821886383 | Hitesh     | Chauhan   | AWS            | MUMBAI   |
| 9821886383 | Hitesh     | Chauhan   | AWS            | MUMBAI   |
| 1234567    | Hitesh     | Chauhan   | AWS            | MUMBAI   |
| 1234567    | Hitesh     | Chauhan   | AWS            | MUMBAI   |
| 123455     | Hitesh     | Chauhan   | AWS            | MUMBAI   |
| 89741253   | Hit        | Cha       | RAJA           | Mumnbai  |
+------------+------------+-----------+----------------+----------+
6 rows in set (0.00 sec)

mysql> select * from employee;
+------------+------------+-----------+----------------+----------+
| empid      | first_name | last_name | primary_skills | location |
+------------+------------+-----------+----------------+----------+
| 9821886383 | Hitesh     | Chauhan   | AWS            | MUMBAI   |
| 9821886383 | Hitesh     | Chauhan   | AWS            | MUMBAI   |
| 1234567    | Hitesh     | Chauhan   | AWS            | MUMBAI   |
| 1234567    | Hitesh     | Chauhan   | AWS            | MUMBAI   |
| 123455     | Hitesh     | Chauhan   | AWS            | MUMBAI   |
| 89741253   | Hit        | Cha       | RAJA           | Mumnbai  |
| 5410       | RAJA       | JARA      | AWS            | MUMBAI   |
+------------+------------+-----------+----------------+----------+
7 rows in set (0.00 sec)

mysql> 
```

RDS Instance



EmployeeData

Scan or query items

Scan

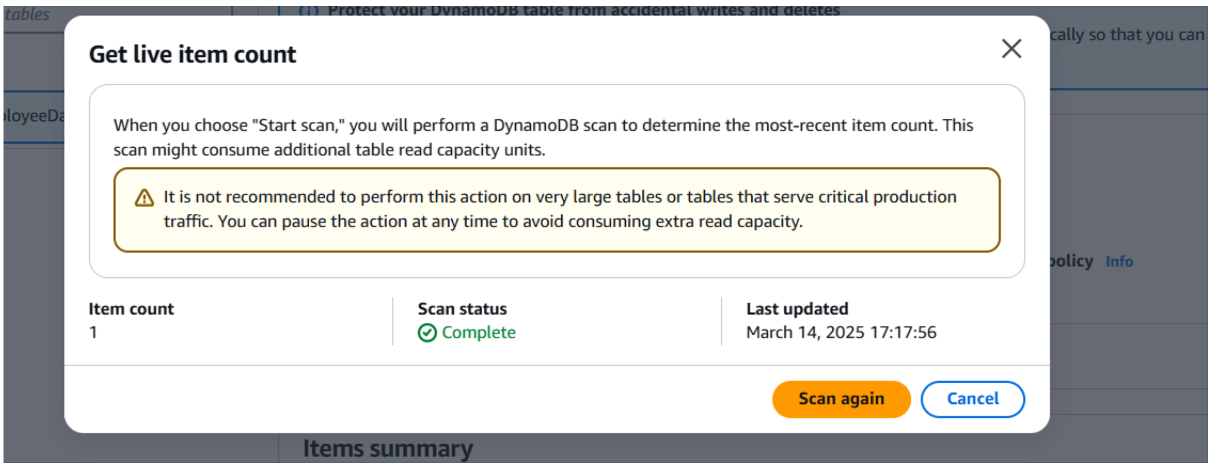
Select a table or index: Table - EmployeeData

Select attribute projection: All attributes

Filters

Attribute name: Enter attribute nam
Type: String
Condition: Equal to
Value: Enter attribute value

Completed. Read capacity units consumed: 2


tables | Protect your DynamoDB table from accidental writes and deletes

Get live item count

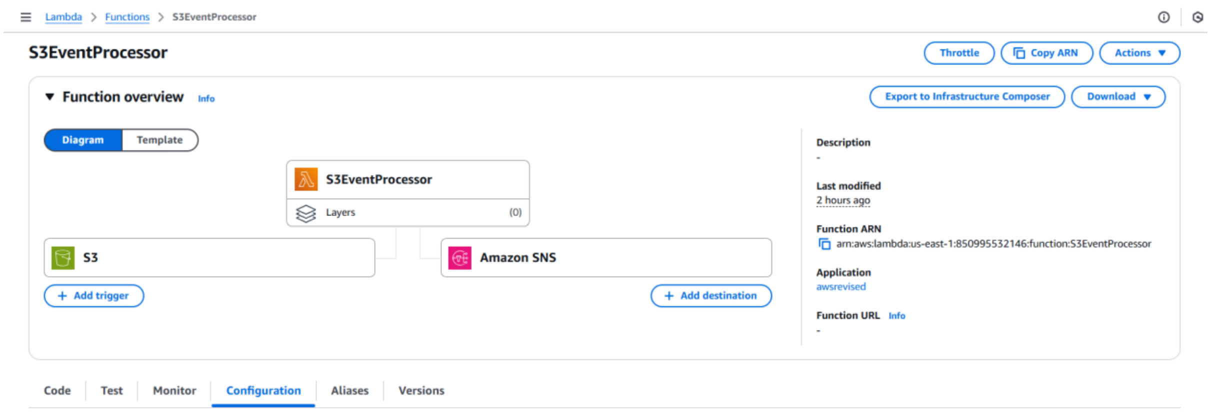
When you choose "Start scan," you will perform a DynamoDB scan to determine the most-recent item count. This scan might consume additional table read capacity units.

⚠ It is not recommended to perform this action on very large tables or tables that serve critical production traffic. You can pause the action at any time to avoid consuming extra read capacity.

Item count	Scan status	Last updated
1	✓ Complete	March 14, 2025 17:17:56

Step — 6: Create a lambda function that will trigger the s3 bucket as soon as files are uploaded in the S3 bucket and set the destination to SQS notification to get the user or admin of this web application to know about employee data.

Lambda function to get triggered when an object is uploaded to the bucket. SQS Queue for Lambda to respond with user email ID subscription.



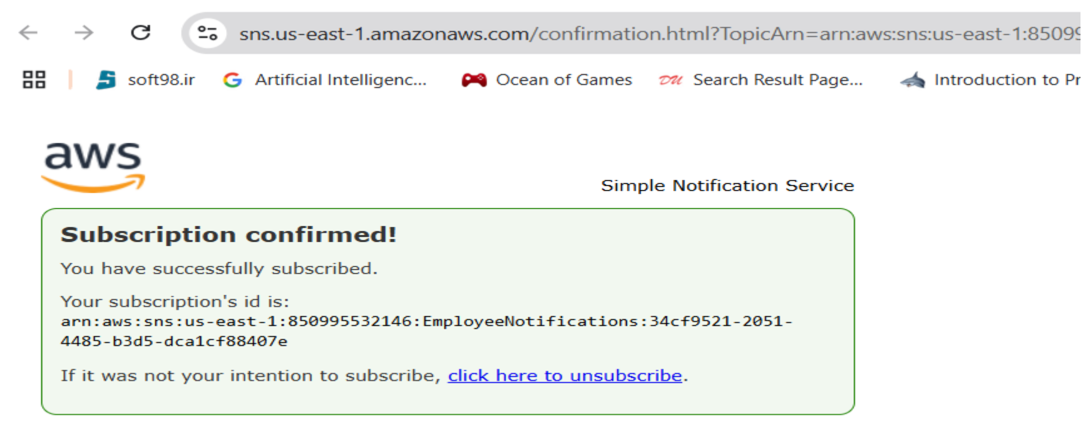
Description
-

Last modified
2 hours ago

Function ARN
arn:aws:lambda:us-east-1:850995532146:function:S3EventProcessor

Application
awsrevised

Function URL Info
-


sns.us-east-1.amazonaws.com/confirmation.html?TopicArn=arn:aws:sns:us-east-1:85099

Subscription confirmed!

You have successfully subscribed.

Your subscription's id is:
arn:aws:sns:us-east-1:850995532146:EmployeeNotifications:34cf9521-2051-4485-b3d5-dca1cf88407e

If it was not your intention to subscribe, click here to unsubscribe.