**Practical Training: Using EC2 and S3 in AWS Sandbox**

**Objective**

In this lab-based session, students will:

1. Create an Amazon EC2 instance
2. Create an Amazon S3 bucket
3. Upload a CSV file (apartment prices dataset) to the bucket
4. Configure the EC2 instance to access the S3 bucket
5. Write and run a Python script on the EC2 instance to read and process the dataset from S3

**Note**: This exercise is conducted in the **AWS Academy sandbox**, which is region-restricted to **us-east-1** and uses pre-created roles (**LabRole**, **LabInstanceProfile**) for permissions.

**Step 1: Create an Amazon S3 Bucket**

1. Go to the **AWS Management Console**
2. Navigate to **S3**
3. Click **Create bucket**
4. Provide a globally unique name, e.g., yourname-studentid-bucket
5. Select the region: **US East (N. Virginia) – us-east-1**
6. Leave other settings as default and click **Create bucket**

**Step 2: Upload a CSV File to S3**

1. Open the newly created bucket
2. Click **Upload** > **Add files**
3. Upload the **apartment prices dataset** (e.g., apartment\_data.csv)
4. Click **Upload**

**Step 3: Launch an Amazon EC2 Instance**

1. Go to **EC2** in the AWS console
2. Click **Launch Instance**
3. Instance Name: s3-access-demo
4. Amazon Machine Image: **Amazon Linux 2 AMI**
5. Instance Type: **t3.micro** (within sandbox limits)
6. Key pair: Choose existing key pair → vockey
7. Network settings: Allow **SSH traffic**
8. Storage: Leave as default (8 GB)
9. Under **Advanced Details** → IAM Role: LabRole
10. Click **Launch Instance**

**Step 4: Connect to the EC2 Instance**

**Option A: Using EC2 Instance Connect**

1. Go to **Instances**, select your instance
2. Click **Connect** > **EC2 Instance Connect**
3. Click **Connect** to open a browser-based terminal

**Option B: Using SSH (if enabled)**

1. Download the labsuser.pem key
2. Open your terminal and run:

chmod 400 labsuser.pem

ssh -i labsuser.pem ec2-user@<instance-public-ip>

**Step 5: Install Required Python Packages**

In the EC2 terminal:

sudo yum update -y

sudo yum install python3 -y

pip3 install boto3 pandas

**Step 6: Write Python Script to Read and Analyze Dataset from S3**

1. Open a new file:

nano read\_apartment\_data.py

1. Paste the following code (replace your-bucket-name and apartment\_data.csv with your actual values):

import boto3

import pandas as pd

from io import StringIO

# S3 client

s3 = boto3.client('s3', region\_name='us-east-1')

# S3 bucket and file details

bucket = 'your-bucket-name'

key = 'apartment\_data.csv'

# Get the object

response = s3.get\_object(Bucket=bucket, Key=key)

content = response['Body'].read().decode('utf-8')

# Load CSV into pandas DataFrame

df = pd.read\_csv(StringIO(content))

# Display data and basic statistics

print("First 5 records:")

print(df.head())

print("\nAverage apartment price by city:")

print(df.groupby('City')['Price'].mean())

1. Save and exit: Press Ctrl+O, Enter, then Ctrl+X
2. Run the script:

bash

CopyEdit

python3 read\_apartment\_data.py

**Step 7: Verify IAM Role Permissions**

The EC2 instance uses the LabRole, which is pre-configured with S3 access.

To verify:

1. Go to **IAM** > **Roles**
2. Select **LabRole**
3. Confirm that it includes **AmazonS3ReadOnlyAccess** or similar policies that allow s3:GetObject

**Wrap-up Discussion**

* Highlight how EC2 and S3 can be integrated for basic data workflows
* Discuss the cost-effectiveness and scalability of cloud computing
* Explain the use of IAM roles for secure and managed permissions
* Point out how this workflow mimics real-world data pipeline stages

**Optional Exercise: Processing Larger Datasets on High-End EC2 Instances**

To emphasize the scalability of cloud computing:

1. Choose a large dataset from the AWS Open Data Registry (e.g., NYC Taxi Trip Data)
2. Launch a high-performance EC2 instance (e.g., m5.2xlarge)
3. Upload a data sample to S3
4. Repeat the same process to read and analyze data using pandas
5. Discuss how on-demand resources can significantly reduce processing time for large-scale analytics tasks