**1. Hosting Static Websites on AWS S3 and EC2**

**Create S3 Bucket: Go to S3 → Create a globally unique bucket name → Uncheck "Block all**

**public access".**

**Upload Files: Upload your index.html, styles.css, script.js, etc., to the bucket.**

**Enable Static Website Hosting: In Properties, enable static website hosting → Set index.html**

**as index document.**

**Make Bucket Public: In Permissions → Add a bucket policy to allow public read access**

**Enable Versioning: In Properties → Enable bucket versioning to keep file history.**

**Access Website: Use the S3 website endpoint URL to view your live site.**

S3 Static Website:  
Uncheck “Block all public access” -> Enable Versioning -> upload html, css files -> Properties -> Static website hosting enable

Go to Permissions -> Bucket Policy and paste:

{

"Version": "2012-10-17",

"Statement": [{

"Sid": "PublicReadForWebsite",

"Effect": "Allow",

"Principal": "\*",

"Action": "s3:GetObject",

"Resource": "arn:aws:s3:::web-pds/\*"

}]

}

**2. EC2 Setup and MySQL Database Management, including Database with Triggers and Stored Procedures**

**Launch a Python Application on EC2 in One Region**

**Step 1: Log in to AWS**

**• Go to https://aws.amazon.com**

**• Sign in to the AWS Management Console.**

**Step 2: Launch EC2 Instance in a Region (e.g., N. Virginia)**

**1. Go to EC2 > Instances > Launch Instances.**

**2. Name the instance: PythonAppServer.**

**3. Choose Amazon Machine Image (AMI): Ubuntu Server 22.04 LTS.**

**4. Choose Instance Type: t2.micro (free tier).**

**5. Key Pair: Create new or use existing (download .pem file).**

**6. Configure:**

**o Allow SSH (22) and HTTP (80) in security group.**

**7. Launch the instance.**

**Step 3: Connect via SSH**

**Migrate EC2 Instance to Another Region**

**Direct migration is not allowed. Use AMI + Snapshots.**

**Step 1: Create an AMI**

**• Go to EC2 > Instances > Select instance > Actions > Image > Create Image**

**• Name it and create.**

**Step 2: Copy AMI to Another Region (e.g., Mumbai)**

**• Go to EC2 > AMIs > Select AMI > Actions > Copy AMI**

**• Choose destination region and copy.**

**Step 3: Launch Instance in New Region**

**• Switch to new region (top-right corner).**

**• Go to AMIs, find the copied AMI, launch a new instance.**

**Setup Another EC2 for MySQL and Basic DB Operations**

**Step 1: Launch Another EC2 Instance (e.g., Name: MySQLServer)**

**• Repeat EC2 launch steps as above.**

**• Allow SSH & MySQL (port 3306) in security group.**

**Step 2: Connect and Install MySQL**

**Step 3: Login and Create Database**

EC2 Python Console Code:

sudo apt update

sudo apt install python3 -y

echo “print(‘Hello World’)” > [app.py](http://app.py)

python3 [app.py](http://app.py)

EC2+MYSQL

sudo apt update

sudo apt install mysql-server

sudo mysql

ALTER USER 'root'@'localhost' IDENTIFIED WITH mysql\_native\_password by 'password';

FLUSH PRIVILEGES;

CREATE DATABASE READLIST;

USE READLIST;

CREATE TABLE readlt(id INT, name VARCHAR(30));

INSERT INTO readlt VALUES(1, ‘ABC’), (2, ‘DEF’);

SELECT \* FROM readlt;  
  
-- Switch to your database

USE READLIST;

-- Create procedure

DELIMITER //

CREATE PROCEDURE GetAllReads()

BEGIN

SELECT \* FROM readlt;

END //

DELIMITER ;

-- Call procedure

CALL GetAllReads();

CREATE TABLE readlt\_log (

log\_id INT AUTO\_INCREMENT PRIMARY KEY,

action\_time TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

message VARCHAR(100)

);

DELIMITER //

CREATE TRIGGER after\_insert\_readlt

AFTER INSERT ON readlt

FOR EACH ROW

BEGIN

INSERT INTO readlt\_log(message)

VALUES (CONCAT('New record inserted: ID=', NEW.id, ', Name=', NEW.name));

END //

DELIMITER ;

INSERT INTO readlt VALUES(3, 'GHI');

SELECT \* FROM readlt;

SELECT \* FROM readlt\_log;

**3. Web Application Deployment using AWS Elastic Beanstalk**

**STEPS TO DEPLOY THE WEBSITE-**

**• Prepare your Flask app**

**• Ensure you have application.py or app.py with Flask app object named application.**

**• Add requirements.txt with all dependencies.**

**• Add a gitignore file to exclude unnecessary files.**

**• Set up Elastic Beanstalk project**

**• Initialize EB project in your app folder.**

**• Select platform (Python 3.x).**

**• Create a new application environment.**

**• Configure environment**

**• Choose region.**

**• Set environment name and application name.**

**• Select load-balanced or single-instance.**

**• Deploy the app**

**• Create an environment (first deploy).**

**• EB will package your code, upload, and launch EC2 instances.**

**• Once deployed, EB gives you a URL to access the app.**

**• Verify and test**

**• Open the given EB URL in the browser.**

**• Confirm the Flask app is running.**

**• Manage updates**

**• For any code changes, deploy new versions to the environment.**

**• You can scale, configure environment variables, or enable logging from EB console.**

**4. Serverless Computing – S3 and Lambda Integration**

import json

import urllib.parse

import boto3

print('Loading function')

s3 = boto3.client('s3')

def lambda\_handler(event, context):

# print("Received event: " + json.dumps(event, indent=2))

# Get the object from the event and show its content type

bucket = event['Records'][0]['s3']['bucket']['name']

key = urllib.parse.unquote\_plus(event['Records'][0]['s3']['object']['key'], encoding='utf-8')

try:

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

print("CONTENT TYPE: " + response['ContentType'])

return response['ContentType']

except Exception as e:

print(e)

print(

f"Error getting object {key} from bucket {bucket}. "

"Make sure they exist and your bucket is in the same region as this function."

)

raise e

**5. EC2 Auto Scaling using Launch Templates and Scaling Policies**

Lab overview

In this lab, you will use Amazon Elastic Compute Cloud (Amazon EC2) to create a launch template and an Auto Scaling group.

Duration

This lab requires approximately 45 minutes to complete.

Access the AWS Management Console

1. To start the lab session, choose Start Lab in the upper-right corner of the page.

o The lab session starts.

o A timer displays at the top of this page and shows the time remaining in the session.

Tip: To refresh the session length at any time, choose Start Lab again before the timer reaches 0:00.

2. Before continuing, wait until the lab environment is ready. The environment is ready when the lab details appear on

the right side of the page and the circle icon next to the AWS link in the upper-left corner turns green.

3. To return to these instructions, choose the Readme link in the upper-right corner.

4. To connect to the AWS Management Console, choose the AWS link in the upper-left corner, above the terminal

window.

A new browser tab opens and connects you to the AWS Management Console.

Tip: If a new browser tab does not open, a banner or icon is usually at the top of your browser with the message that your

browser is preventing the site from opening pop-up windows. Choose the banner or icon, and then choose Allow pop-ups.

Note: You are using the console through the lab environment, so you are not incurring any actual costs. However, in the real

world, when using a personal or business account to access the console, users incur charges for use of specific AWS services.

Task 1. Create a launch template

4. Choose the Services menu, locate the Compute category, and choose EC2.

5. Note which Region you are in by looking at the upper-right corner of the console (for example, N. Virginia, which is

the us-east-1 Region).

The Amazon EC2 Auto Scaling resources that you create are tied to the Region that you specify.

6. In the left navigation pane, choose Launch Templates.

7. Choose Create launch template, and configure the following:

o Launch template name: Enter my\_template

o Template version description: Enter Test launch template for an Auto Scaling group

This description can help you remember what this launch template is for later.

o Auto Scaling guidance: Select Provide guidance....

o AMI: Choose a version of Amazon Linux 2023 AMI x86\_64 (HVM) from the Quick Start list.

The Amazon Machine Image (AMI) serves as a basic configuration template for your EC2 instances.

o Instance type: Choose t2.micro.

o Key pair name: Choose vockey.

Note that this is optional for a launch template.

o Subnet: Choose Don't include in launch template.

o Skip Security groups.

You will configure a security group later. When a network interface is specified, the security group must be part of it.

o Advanced network configuration: Choose Add network interface and configure with the following settings:

▪ Auto-assign public IP: Choose Enable.

This setting means that public IP addresses will be assigned to instances in a nondefault VPC. This provides the ability for your

instances to communicate with the internet and other AWS services.

▪ Security groups: Choose the security group name that contains Ec2SecurityGroup.

This specifies the default security group of the VPC for the network interface.

▪ Delete on termination: Choose Yes.

This setting deletes the network interface when the Auto Scaling group scales in and terminates the instance to which the

network interface is attached.

o Choose Create launch template.

8. On the confirmation page, hoose View launch templates.

9. Choose the link for the Launch template ID.

10. From the the Actions menu, choose Create Auto Scaling group.

Task 2. Create an Auto Scaling group

11. On the Step 1 page, configure the following:

o Auto Scaling group name: Enter my-first-asg

o Launch template: Choose the launch template that you created in the previous task.

o Version: Choose Latest.

You can also choose a specific version of the launch template when scaling out.

o Choose Next.

12. On the Step 2 page, configure the following:

o VPC: Choose the VPC name that contains Lab VPC.

o Subnets: Choose the subnet name that contains Public Subnet 1.

o Choose Skip to review.

13. On the Review page, choose Create Auto Scaling group.

Task 3. Verify your Auto Scaling group

14. In the left navigation page, choose EC2 Dashboard.

15. Choose Instances (running).

Notice that a new EC2 instance has been started. If the instance does not appear yet, wait a moment and then choose the

refresh icon. The Instance state shows that it is Running.

Lab complete

Congratulations! You have completed the lab.

**6. S3 Bucket File Management and Public Access Configuration**

While creating bucket: Object ownership(ACL enabled, Object writer) then Uncheck ‘Block all public access

Then one of the 2 ways:

I. For System Operators who prefer GUI:

1. Login to AWS Management Console — Navigate to S3 from the list of services.
2. Select Your Bucket — Find the S3 bucket that contains the object you want to make public.
3. Locate the Object — Navigate through the bucket’s folders until you find the desired object.
4. Make it Public— Right-click on the object, choose ‘Make public’.

II. Navigate to Your Bucket’s Permissions Bucket Policy

{

"Version":"2012–10–17",

"Statement":[{

"Sid":"PublicRead",

"Effect":"Allow",

"Principal": "\*",

"Action":["s3:GetObject"],

"Resource":["arn:aws:s3:::BUCKET-NAME/\*"],

"Condition": {

"Bool": {

"aws:SecureTransport":"true"

}

}

}]

}

—------------------------------------------------------------------------------------------------------------------------

S3 Static Website:

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>About Me</title>

<style>

body {

font-family: 'Poppins', sans-serif;

margin: 0;

padding: 0;

background: linear-gradient(to right, #f5f5dc, #d2b48c);

color: #5d4037;

}

header {

background: #8b4513;

color: white;

text-align: center;

padding: 20px;

box-shadow: 0px 4px 6px rgba(0, 0, 0, 0.1);

}

marquee {

font-size: 18px;

color: #fff;

background-color: #d2691e;

padding: 10px 0;

margin: 0;

}

section {

padding: 20px;

margin: 20px auto;

background-color: rgba(255, 248, 220, 0.9);

border-radius: 10px;

box-shadow: 0 4px 8px rgba(0, 0, 0, 0.2);

max-width: 800px;

}

h2 {

color: #8b4513;

margin-bottom: 10px;

}

img {

max-width: 100%;

border-radius: 10px;

}

.content {

display: flex;

gap: 20px;

flex-wrap: wrap;

}

.content div {

flex: 1;

min-width: 250px;

}

.image-section {

text-align: center;

}

.image-section img {

width: 150px;

border-radius: 50%;

margin: 10px auto;

}

footer {

text-align: center;

padding: 15px;

background: #8b4513;

color: white;

position: static;

margin-top: 20px;

}

.social-links a {

text-decoration: none;

margin: 0 10px;

color: #ffdab9;

font-size: 18px;

}

.social-links a:hover {

text-decoration: underline;

}

</style>

</head>

<body>

<header>

<h1>Welcome to my first webpage!</h1>

</header>

<marquee behavior="scroll" direction="left">Hi, I’m Nishita Jain, and this is a glimpse into my life, passions, and projects!</marquee>

<section class="image-section">

<h2>Hello, I'm Nishita! (22BBS0161)</h2>

<p>A tech enthusiast, programmer, and lifelong learner.</p>

</section>

<section>

<h2>Introduction</h2>

<p>I am a B.Tech student at VIT Vellore, specializing in Computer Science with a focus on Business Systems. My journey is fueled by a passion for technology and innovation, constantly seeking to explore new horizons in the tech world.</p>

</section>

<section>

<h2>My Skills</h2>

<div class="content">

<div>

<h3>Programming Languages</h3>

<ul>

<li>Python</li>

<li>C, C++</li>

<li>Java</li>

</ul>

</div>

<div>

<h3>Other Skills</h3>

<ul>

<li>Web Development</li>

<li>Data Science</li>

<li>SuiteScripts</li>

</ul>

</div>

</div>

</section>

<section>

<h2>My Interests</h2>

<p>I enjoy exploring emerging technologies, contributing to tech communities, and being part of exciting projects. As Vice Chair of the French Literary Association, I actively contribute to cultural events that bring students together.</p>

<img src="tech\_image.png" alt="Interest-related image">

</section>

<section>

<h2>Fun Facts</h2>

<p>📚 I love reading books on AI and ML.<br>🌍 I'm learning French and exploring diverse cultures.<br>🎨 In my downtime, I enjoy designing interfaces and working on creative projects.</p>

</section>

<footer>

<p>Contact me: <a href="mailto:nishitajain@example.com">nishitajain@example.com</a></p>

<div class="social-links">

<a href="https://in.linkedin.com/in/nishita11jain">LinkedIn</a> | <a href="https://github.com/nishita24jain">GitHub</a> | <a href="#">Portfolio</a>

</div>

</footer>

</body>

</html>

—------------------------------------------------------------------------------------------------------------------------