Assignment 1 – IaaS and FaaS

Updated demo recording: SYLLABUS_EC2_LAMBDA.mp4

You will submit screenshots of the core steps in one PDF for all assignments. Please see the sample submission.
ACCEPTANCE CRITERIA – Include the followings in the **PDF**:

- Web page that shows your name. The web app in EC2.
- Lambda that returns your friends names.
- S3 bucket URL.

Please include the **entire screenshot of the desktop.** Not just portion of it.

The goal of assignment 1 – One of the main topics in lesson 1 is Cloud Services Models. We are covering the IaaS and FaaS models in the course and you are practicing these 2 in this lab.

1. The first one is, configuring the simplest web app on a virtual machine (IaaS), essentially playing with a server.
2. The second task is, practicing FaaS by creating a mock RESTful API in Lambda.
3. The third task is, to show you how powerful the cloud is. The third task demonstrates how effortlessly you can host websites built in React, Angular, etc., within minutes. This practical and cost-effective method has been used to host many real-life apps.

Task 1 – IaaS (EC2) – Launch a simple web app on EC2

- Spin up an EC2 instance.
    a. Allow HTTP:80 port from the world (0.0.0.0/0) in the Network Setting panel.
    b. SSH:22 from 0.0.0.0/0 is selected by default. Double check that. It will be used to connect to the instance.
- Connect to the instance. Ther are 4 ways to connect to your server, SSH, EC2 connect, IAM. Refer: Connect to your Linux instance
- Configure a web server on EC2.

```
sudo -s => Logging as a root user so you can execute any command
yum install httpd -y => Installing an Apache web server package
service httpd start => Starting the server
cd /var/www/html  => Changing the directory to customize the default page.
nano index.html => Create the index.html and write your name here as HTML.
```

If the web app is not responding:

- Make sure you are making http://<your_ip>, not **https** in your browser.
- Check Security Group if it allows port 80.
- Start the AWS Academy lab if applicable. AWS Academy automatically stops instances when the lab ends or after 4 hours.

Task 2 – FaaS (Lambda) – Simple API with Lambda function URL

Create a lambda function that returns an array of strings. Make it an API by enabling the public URL.
Refer: Creating and managing Lambda function URLs

a. Enable URL and enable **CORS** and set **Auth Type** to none.

b. [If it is AWS Academy account] Go to Change IAM role and select preconfigured **LabRole.** If it is a regular AWS account, skip this step. The IAM role will be created automatically.

Task 3 – Deploying a static website in S3

Call the API in Lambda from the React app and deploy the app in S3. Refer: Hosting a static website using Amazon S3

c. Install NodeJS on your laptop
d. npx create-react-app appname – It will create the React app template
e. npm install axios
f. npm start – to start your front-end app
g. npm run build – after testing, build the app
h. Create a bucket and deselect “Block public access”
i. Drop all files inside the build folder into the bucket.
j. Write a policy that makes all objects in the bucket public. Refer to the next section.
k. Enable “static website hosting” and define the index.html as the index and error page.

If you google, you will find examples of these 3 tasks all over the internet.

Task4: Delete the EC2 instance

Delete the EC2 instance once you are done. EC2 costs a lot whereas Lambda and S3 don’t cost much.

Snippets

The bucket policy that makes all objects inside it public:

```
{
  "Id": "Policy1650912821527",
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "Stmt1650912820312",
      "Action": [
        "s3:GetObject"
      ],
      "Effect": "Allow",
      "Resource": "arn:aws:s3:::<yourbucket>/*",
      "Principal": "*"
    }
  ]
}
```

The React web app:

```
import axios from "axios";
import { useEffect, useState } from "react";

export default function App() {
  const [students, setstudents] = useState([]);

  useEffect(() => {
    async function fetchStudents() {
      const studentsFromLambda = (
        await axios.get(
          "<your lambda URL>"
        )
      ).data;
      setstudents(studentsFromLambda);
      console.log(studentsFromLambda);
    }

    fetchStudents();
  }, []);
  return (
    <div>
      Cloud Computing course
      <ol>
        {students.map((student) => (
          <li>{student}</li>
        ))}
      </ol>
    </div>
  );
}
```