

Building a Media Sharing Website

OBJECTIVES:

- This technical report demonstrates how to launch a new Amazon EC2 instance to run your web server, creating an Amazon DynamoDB database to store data and creating a new Amazon S3 bucket to hold your media files.
- These procedures are essential for developing a web application that needs a database to hold your data, media storage, and a secure environment.

PREREQUISITES:

An AWS account, IAM permission to work with S3 and DynamoDB.

PROCEDURE:

1. Creation of a Amazon S3 bucket, we created a bucket named "assignmentarchitecture".
 - Various ways to access it, be it either CLI or the UI. UI seems to be an easier option for this assignment as there are frequent changes and it's visibly easier to delete and update files.
 - Create a bucket, name is unique which hasn't been ever used.
 - Choose the area where it should be created and press the "Create" button.
 - After creating your bucket, you can add your media files to it.
 - By default, it's not accessible by the public as we have blocked it. We can make it public in the permissions section under "Block public access (bucket settings)".
 - We then give a policy to our bucket which lets outside users make changes like PUT, GET etc. Policy can be created via the policy generator. In this assignment, we have selected "PutObject" and "GetObject".

2. Creating Amazon DynamoDB database to store data from the input. We created a table named "assignment". Some of the steps are as shown below:
 - Click the "Create Table" button on the DynamoDB console in your AWS account.
 - Type a name, a primary key, and any additional properties for your table if you wish to.
 - The primary key we set was "ID" which in our example have set up such a way that the image uploaded will be under that.
 - Set up your table's throughput and encryption preferences or leave it as by default.
 - Examine your table's specifics and put it together.

3. Launching a new Amazon EC2 instance to run your web server. We created "assignmentfinal" as the name of our instance. Some steps we followed were:
 - In the EC2 console in your AWS account and click on the "Launch Instance" button.
 - Choose an Amazon Machine Image (AMI) for your instance. (We kept that by default)
 - Choose an instance type based on your application requirements, t2.micro should be fine for this basic work.
 - Configure your instance details, including network settings, storage, and security groups.
 - Just make sure you have attached "LabRole" under Additional settings which will let all our IAM policies assigned under "LabRole" Role and not ask for access key or any credentials.
 - Review your instance details and launch your instance.

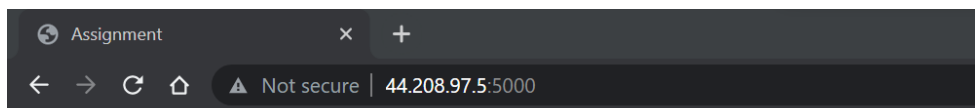
4. Creating Web application using Python inside EC2:
 - Initially, open the CLI of the instance or connect with SSH. Install python, pip, pillow, boto3 and flask.
 - Python is used to write the main code (app.py in our case) and index.html to display the webpage. Pip is a package manager for python which we use to install other libraries and modules.
 - Boto3 is an AWS SDK for python. It allows us to make software that can use S3, EC2 etc. in our assignment.
 - Pillow is again a library in python and used for web development and ML, also resizes, and converts image formats.
 - Flask is a web framework for python which allows us to build web applications. It receives the images and creates a thumbnail and

displays them on the table of our webpage. It also sends the metadata to DynamoDB.

5. How everything works at the end:

- Index.html is the most basic yet important part in this assignment. It laid out the forms where users can give their information. It also displays the information in the form of table with an option to delete next to each input received.
- App.py file is the heart of this app. It states what happens from the time we hit “Upload” button to deletion of the content from the sources. Here, app.py defines our flask application. It defines our routes which maps our URLs to functions which handles our requests. It also returns the HTML content to the browser and connect that with database and S3. The display of content on browser too works because of it.
- Both S3 and dynamodb are connected via the EC2 which can send the data and delete the respective resources simultaneously.

UI SCREENSHOTS:




Upload Image

Name:

Description:

Location:

Choose an image to upload: No file chosen

Thumbnail	Name	Description	Location	Action
	sahil	test to check if it works	toronto	<input type="button" value="Delete"/>

assignmentarchitecture [Info](#)

Publicly accessible

Objects

Properties

Permissions

Metrics

Management

Access Points

Objects (1)

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](#)

Copy S3 URI

Copy URL

Download

Open

Delete

Actions

Create folder

Upload

Find objects by prefix

Show versions

< 1 >

<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	Screenshot 2022-11-21 125516.png	png	February 24, 2023, 21:44:26 (UTC-05:00)	89.9 KB	Standard

DynamoDB > Items > assignment

>

assignment

Autopreview

View table details

▶ Scan or query items

Expand to query or scan items.

✔ Completed. Read capacity units consumed: 0.5

Items returned (1)

Actions

Create item

< 1 >

⚙️

<input type="checkbox"/>	id	description	location	name
<input type="checkbox"/>	Screenshot 2022-11-21 125516.png	test to check if it works	toronto	sahil

CONCLUSION:

We just made a simple webpage which sends, displays, fetches, and deletes data from the S3 bucket and the NOSQL database. There are a few key parts where it can be difficult, and some common errors can occur. We initially had to understand the code as we knew the basics but packages like boto3 were new for us. Other areas to check were the security inbound rules for ec2. The understanding was how any media form can be uploaded to any storage class and a NoSQL database can be created from a hosted EC2 instance.