KGiSL Institute of Technology

(Approved by AICTE, New Delhi; Affiliated to Anna University, Chennai)

Recognized by UGC, Accredited by NBA (IT)

365, KGiSL Campus, Thudiyalur Road, Saravanampatti, Coimbatore – 641035.

CCS336 – CLOUD SERVICE MANAGEMENT

NAME	: …………………………………
REG. NO.	: …………………………………
COURSE	: …………………………………
SEMESTER	: ………………………………….
BATCH	: ………………………………….

KGiSL Institute of Technology

(Approved by AICTE, New Delhi; Affiliated to Anna University, Chennai)
Recognized by UGC, Accredited by NBA (IT)
365, KGiSL Campus, Thudiyalur Road, Saravanampatti, Coimbatore – 641035.

NAME	**:**
CLASS	**:**
UNIVERSITY REG NO	**:**

Certified that, this is a bonafide record of work done by ……………………………………….

Of ………………………………………………………………. branch **CLOUD SERVICE MANAGEMENT LABORATORY**, during fourth semester of academic year 2022-2023.

Faculty In-charge **Head of the Department**

Submitted during Anna University Practical Examination held on …………………… at KGiSL Institute of Technology, Coimbatore – 641 035.

Internal Examiner **External Examiner**

S. NO	DATE	LIST OF THE EXPERIMENTS	PAGE NO	MARKS	SIGNATURE
1		CREATING ACCOUNT IN AWS			
2		CREATING A WEB APPLICATION USING AWS			
3		BILLING ALERT			
4		SETTING UP BUDGET ALERT			
5		DEPLOYING A WEB APPLICATION USING S3			

EXP. NO: 01 **DATE:**	**CREATING ACCOUNT IN AWS**

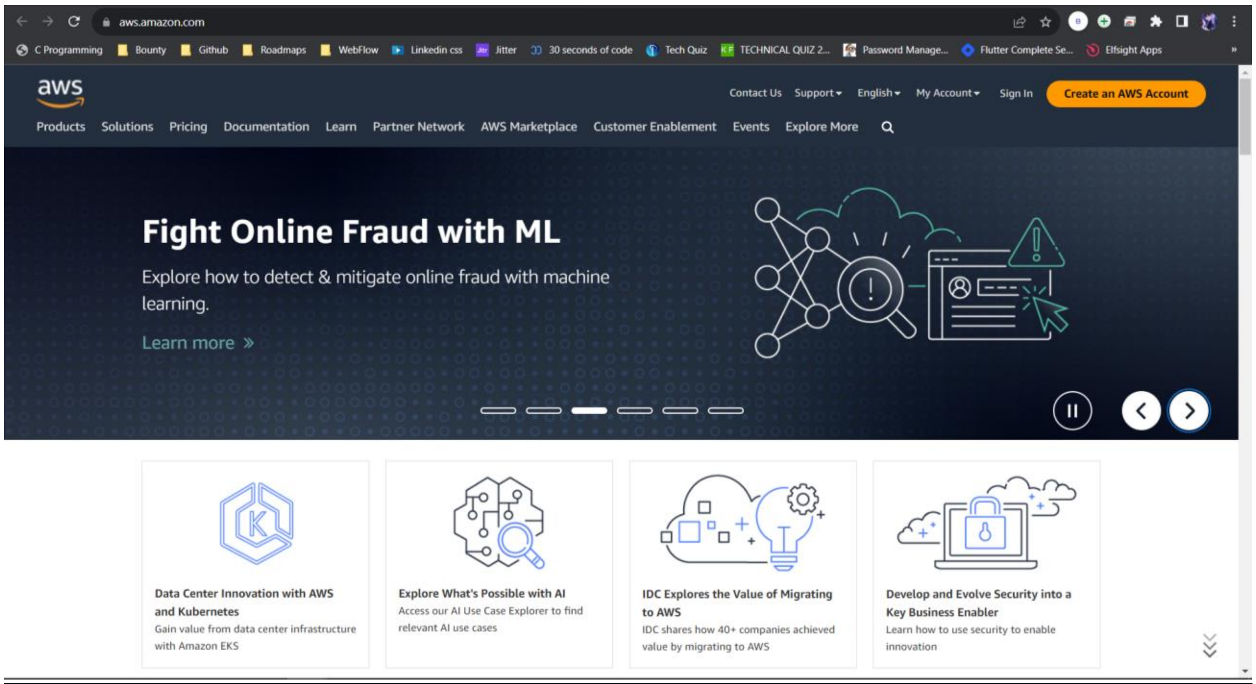
AIM:

Create a Cloud Organization in AWS/Google Cloud/or any equivalent Open-Source cloud softwares like Openstack, Eucalyptus, OpenNebula with Role-based access control.

PROCEDURE:

Creating an AWS account

1. **Open the Amazon Web Services home page .**



Fight Online Fraud with ML

Explore how to detect & mitigate online fraud with machine learning.

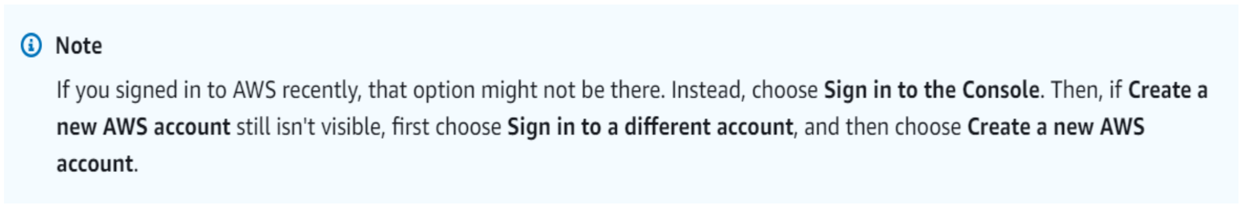
Data Center Innovation with AWS and Kubernetes
Gain value from data center infrastructure with Amazon EKS

Explore What's Possible with AI
Access our AI Use Case Explorer to find relevant AI use cases

IDC Explores the Value of Migrating to AWS
IDC shares how 40+ companies achieved value by migrating to AWS

Develop and Evolve Security into a Key Business Enabler
Learn how to use security to enable innovation

2. **Choose Create an AWS account.**



Note

If you signed in to AWS recently, that option might not be there. Instead, choose **Sign in to the Console**. Then, if **Create a new AWS account** still isn't visible, first choose **Sign in to a different account**, and then choose **Create a new AWS account**.

3. Enter your account information, and then choose Verify email address. This will send a verification code to your specified email address.
4. Enter your verification code, and then choose Verify.
5. Enter a strong password for your root user, confirm it, and then choose Continue. AWS requires that your password meet the following conditions:

    a. It must have a minimum of 8 characters and a maximum of 128 characters.

    b. It must include a minimum of three of the following mix of character types: uppercase, lowercase, numbers, and ! @ # $ % ^ & * () <> [] {} | _+-= symbols.

    c. It must not be identical to your AWS account name or email address.

6. Choose Business or Personal. Personal accounts and business accounts have the same features and functions.
7. Enter your personal information.
8. Read and accept the AWS Customer Agreement. Be sure that you read and understand the terms of the AWS Customer Agreement.
9. Choose Continue. At this point, you'll receive an email message to confirm that your AWS account is ready to use. You can sign in to your new account by using the email address and password you provided during sign up. However, you can't use any AWS services until you finish activating your account.

10. Enter the information about your payment method, and then choose Verify and Continue. If you want to use a different billing address for your AWS billing information, choose Use a new address. You can't proceed with the sign-up process until you add a valid payment method.
11. Enter your country or region code from the list, and then enter a phone number where you can be reached in the next few minutes.
12. Enter the code displayed in the CAPTCHA, and then submit.
13. When the automated system contacts you, enter the PIN you receive and then submit.

14. Select one of the available AWS Support plans and choose Complete sign up. A confirmation page appears that indicates that your account is being activated.
15. Check your email and spam folder for an email message that confirms your account was activated. Activation usually takes a few minutes but can sometimes take up to 24 hours. After you receive the activation message, you have full access to all AWS services.

RESULT:

Create a Cloud Organization in AWS/Google Cloud/or any equivalent Open-Source cloud softwares like Openstack, Eucalyptus, OpenNebula with Role-based access control is executed successfully and the output is verified.

EXP. NO: 02 DATE:	CREATING A WEB APPLICATION USING AWS

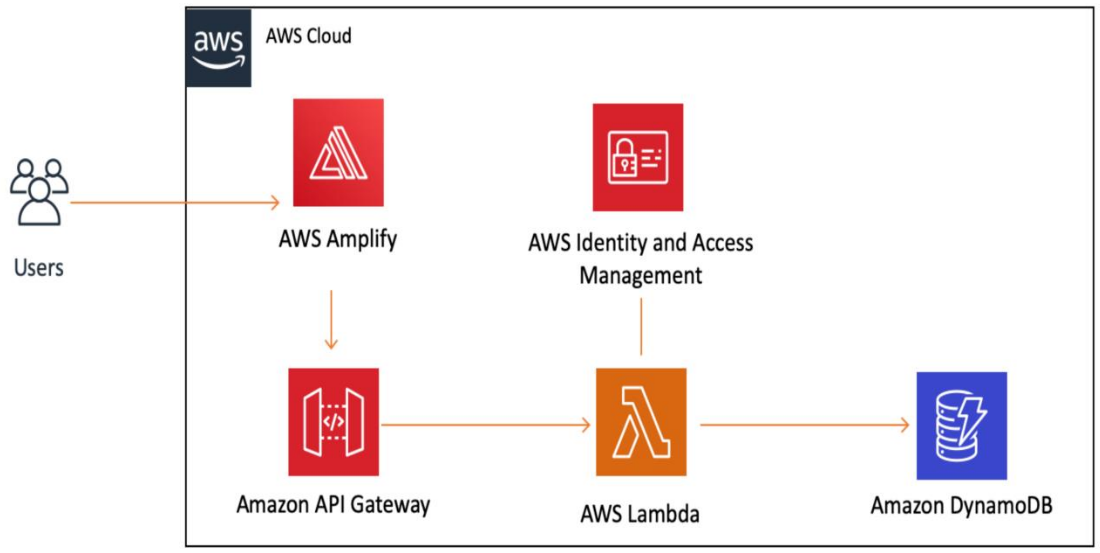
AIM:

To Create a Cost-model for a web application using various services and do Cost-benefit analysis.

PROCEDURE:

Application architecture

The diagram below provides a visual representation of the services used in this exercise and how they are connected. This application uses AWS Amplify, Amazon API Gateway, AWS Lambda, Amazon DynamoDB, and AWS Identity and Access Management (IAM).



1. Create an AWS S3 bucket to store static web application resources.
2. Build Serverless Function : Build a serverless function using AWS Lambda.
3. Link Serverless Function to Web App: Deploy your serverless function with API Gateway.
4. Create Data Table: Persist data in an Amazon DynamoDB table.

5. Add Interactivity to Web App: Modify your web app to invoke your API.
6. Clean Up Resources: Clean up resources used in this exercise

Step 1: Create a Web App

a) Create a Web App

1. Open your favorite text editor on your computer. Create a new file and paste the following HTML in it:

```
<!DOCTYPE html>
<html>
<head>
	<meta charset="UTF-8">
	<title>Hello World</title>
</head>
<body>
	Hello World
</body>
</html>
```

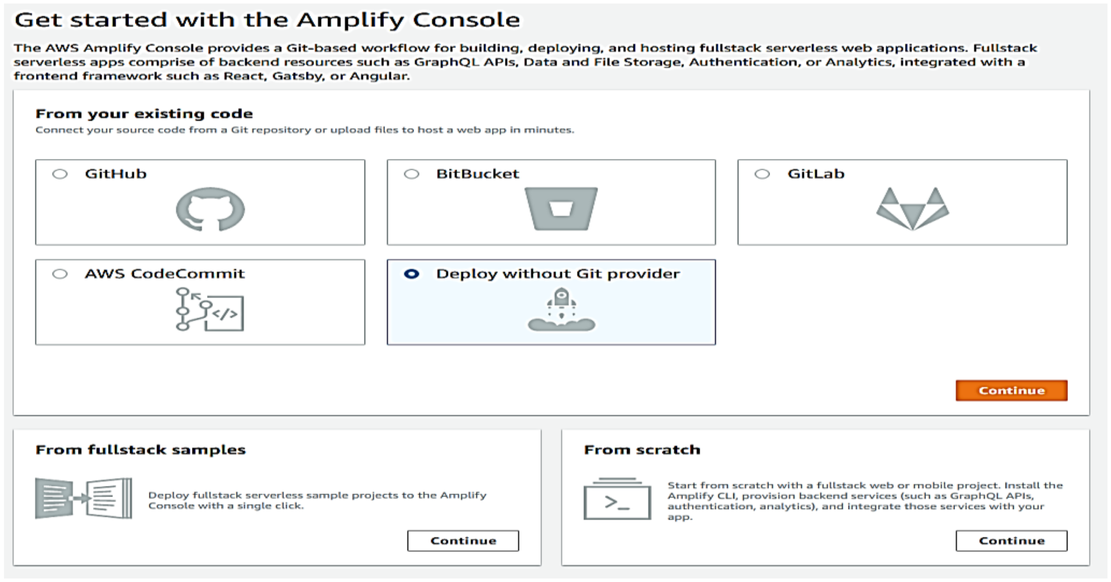
2. Save the file as index.html.

3. ZIP (compress) only the HTML file.

4. In a new browser window, log into the Amplify console. Note: We will be using the Oregon (us-west-2) Region for this tutorial.

5. In the Get Started section, under Host your web app, choose the orange Get started button.

6. Select Deploy without Git provider. This is what you should see on the screen:


Get started with the Amplify Console

The AWS Amplify Console provides a Git-based workflow for building, deploying, and hosting fullstack serverless web applications. Fullstack serverless apps comprise of backend resources such as GraphQL APIs, Data and File Storage, Authentication, or Analytics, integrated with a frontend framework such as React, Gatsby, or Angular.

From your existing code

Connect your source code from a Git repository or upload files to host a web app in minutes.

GitHub / BitBucket / GitLab / AWS CodeCommit / [✓ Deploy without Git provider]

From fullstack samples

Deploy fullstack serverless sample projects to the Amplify Console with a single click.

From scratch

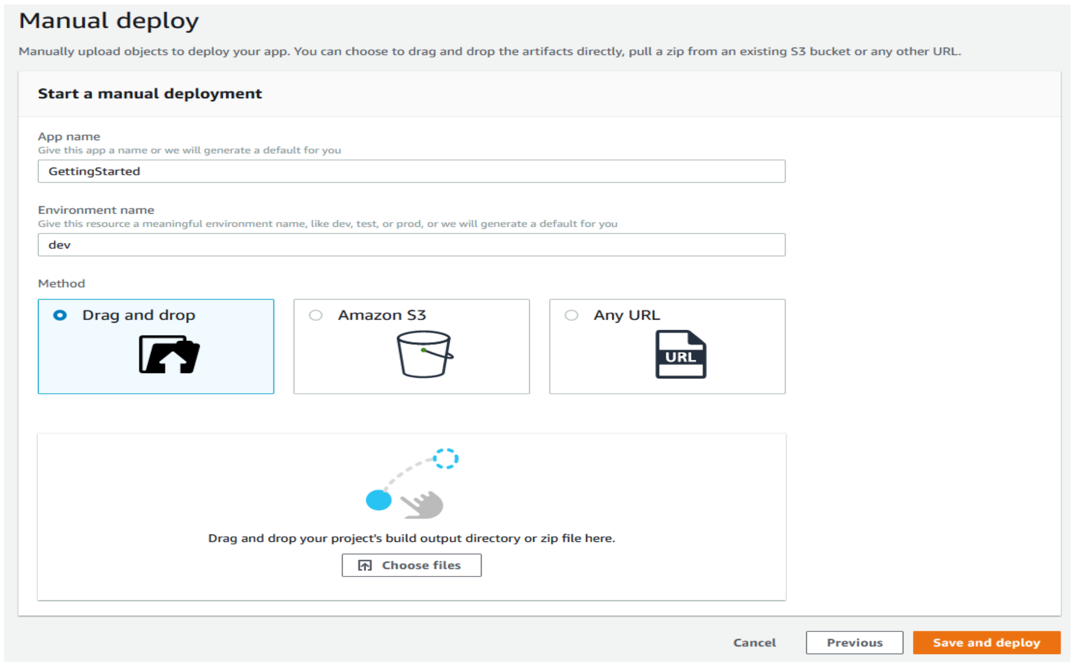
Start from scratch with a fullstack web or mobile project. Install the Amplify CLI, provision backend services (such as GraphQL APIs, authentication, analytics), and integrate those services with your app.

7. Choose the Continue button.

8. In the App name field, enter GettingStarted.

9. For Environment name, enter dev.

10. Select the Drag and drop method. This is what you should see on your screen:


Manual deploy

Manually upload objects to deploy your app. You can choose to drag and drop the artifacts directly, pull a zip from an existing S3 bucket or any other URL.

Start a manual deployment

App name
Give this app a name or we will generate a default for you
GettingStarted

Environment name
Give this resource a meaningful environment name, like dev, test, or prod, or we will generate a default for you
dev

Method
[✓ Drag and drop] / Amazon S3 / Any URL

Drag and drop your project's build output directory or zip file here.

11. Choose the Choose files button.

12. Select the ZIP file you created in Step 3.

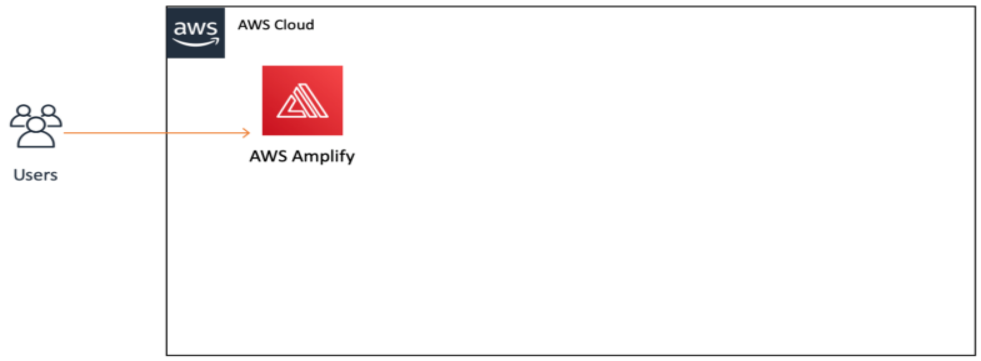
13. Choose the Save and deploy button.

14. After a few seconds, you should see the message Deployment successfully completed.

b) Test your web app

1. Select Domain Management in the left navigation menu.

2. Copy and paste the URL displayed in the form into your browser.



Step 2: Build a Serverless Function

a) Create and configure Lambda function:

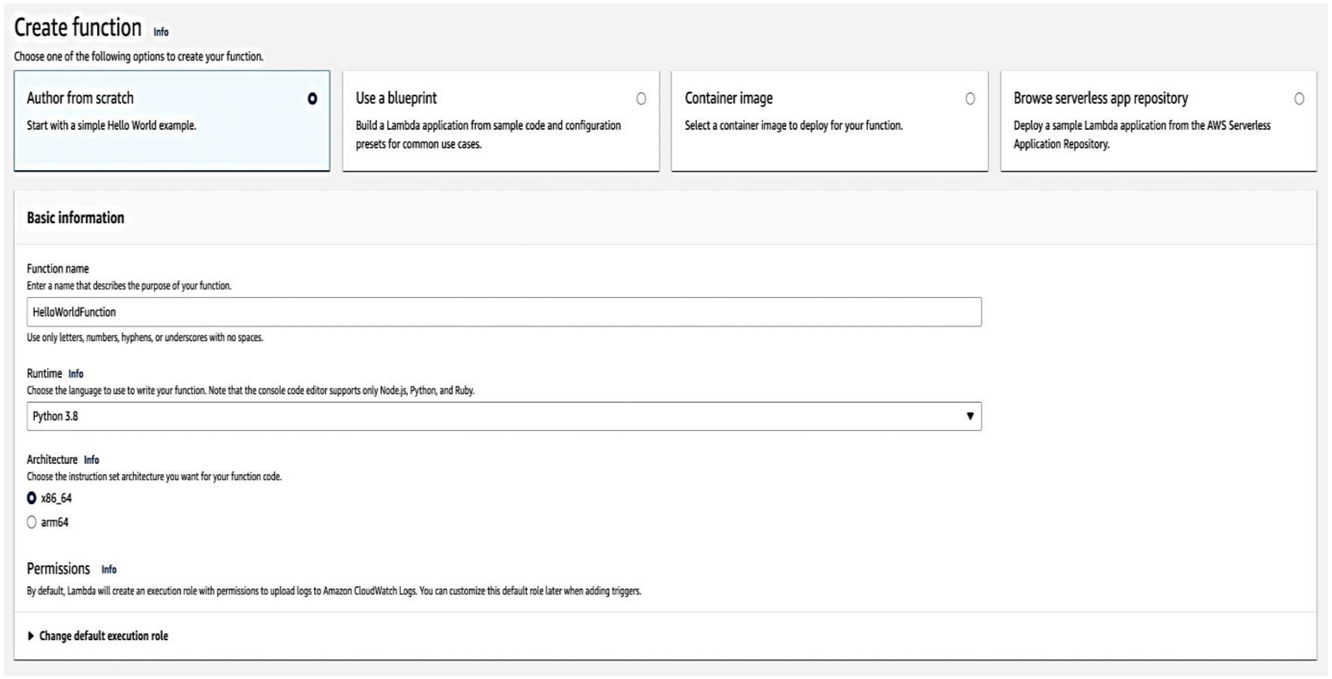
In a new browser tab, log in to the AWS Lambda console.

Make sure you create your function in the same Region in which you created the web app in the previous module. You can see this at the very top of the page, next to your account name.

Choose the orange Create function button.

Under Function name, enter HelloWorldFunction.

Select Python 3.8 from the runtime dropdown and leave the rest of the defaults unchanged.



Create function Info

Choose one of the following options to create your function.

Author from scratch — Start with a simple Hello World example. [✓]

Use a blueprint — Build a Lambda application from sample code and configuration presets for common use cases.

Container image — Select a container image to deploy for your function.

Browse serverless app repository — Deploy a sample Lambda application from the AWS Serverless Application Repository.

Basic information

Function name
Enter a name that describes the purpose of your function.
HelloWorldFunction
Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime Info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Python 3.8

Architecture Info
Choose the instruction set architecture you want for your function code.
[✓ x86_64] arm64

Permissions Info
By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

6. Choose the orange Create function button.

7. You should see a green message box at the top of your screen with the following message "Successfully created the function HelloWorldFunction."

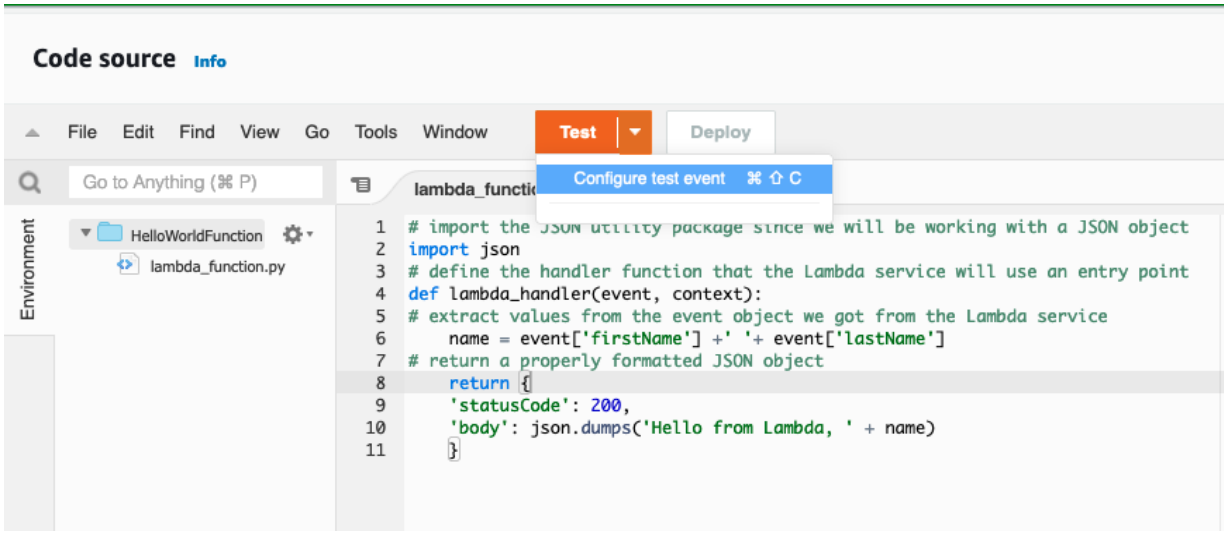
8. Under Code source, replace the code in lambda_function.py with the following:

```
# import the JSON utility package since we will be working with a JSON object
import json
# define the handler function that the Lambda service will use as an entry point
def lambda_handler(event, context):
# extract values from the event object we got from the Lambda service
    name = event['firstName'] +' '+ event['lastName']
# return a properly formatted JSON object
    return {
    'statusCode': 200,
    'body': json.dumps('Hello from Lambda, ' + name)
    }
```

9. Save by going to the file menu and selecting Save to save the changes.

10. Choose Deploy to deploy the changes.

11. Let's test our new function. Choose the orange Test button to create a test event by selecting Configure test event.



```python
# import the JSON utility package since we will be working with a JSON object
import json
# define the handler function that the Lambda service will use an entry point
def lambda_handler(event, context):
# extract values from the event object we got from the Lambda service
    name = event['firstName'] +' '+ event['lastName']
# return a properly formatted JSON object
    return {
    'statusCode': 200,
    'body': json.dumps('Hello from Lambda, ' + name)
    }
```

12. Under Event name, enter HelloWorldTestEvent.

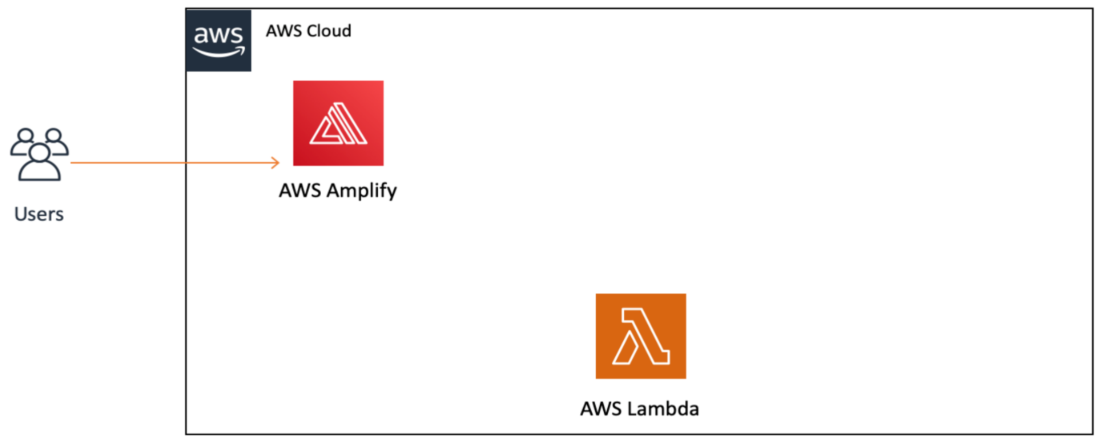
13. Copy and paste the following JSON object to replace the default one:

```
{
"firstName": "Ada",
"lastName": "Lovelace"
}
```

14. Choose the Save button at the bottom of the page.

b)Test your Lambda function

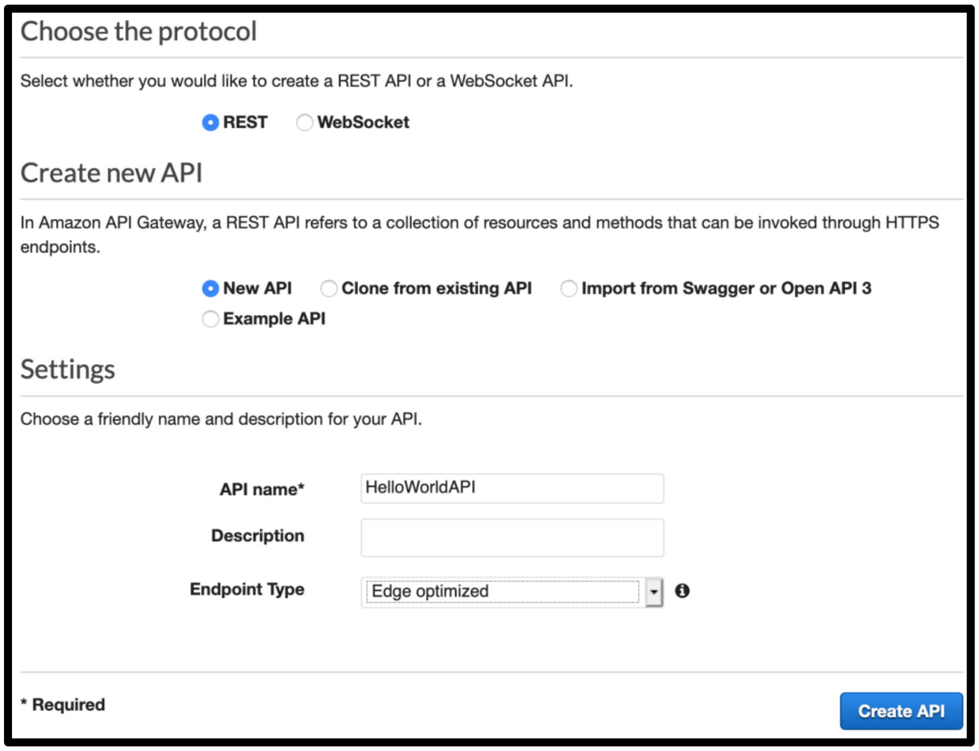
1. Under the HelloWorldFunction section at the top of the page, select Test tab.
2. You should see a light green box at the top of the page with the following text: Execution result: succeeded. You can choose Details to see the event the function returned.



Step 3: Link a Serverless Function to a Web App

a)Create REST API

1. Log in to the API Gateway console.
2. In the Choose an API type section, find the REST API card and choose the Build button on the card.
3. Under Choose the protocol, select REST.
4. Under Create new API, select New API.
5. In the API name field, enter HelloWorldAPI.
6. Select Edge optimized from the Endpoint Type dropdown. (Note: Edge-optimized endpoints are best for geographically distributed clients. This makes them a good choice for public services being accessed from the internet. Regional endpoints are typically used for APIs that are accessed primarily from within the same AWS Region.)
7. Choose the blue Create API button. Your settings should look like the accompanying screenshot.



Choose the protocol

Select whether you would like to create a REST API or a WebSocket API.

[✓ REST] WebSocket

Create new API

In Amazon API Gateway, a REST API refers to a collection of resources and methods that can be invoked through HTTPS endpoints.

[✓ New API] Clone from existing API Import from Swagger or Open API 3 Example API

Settings

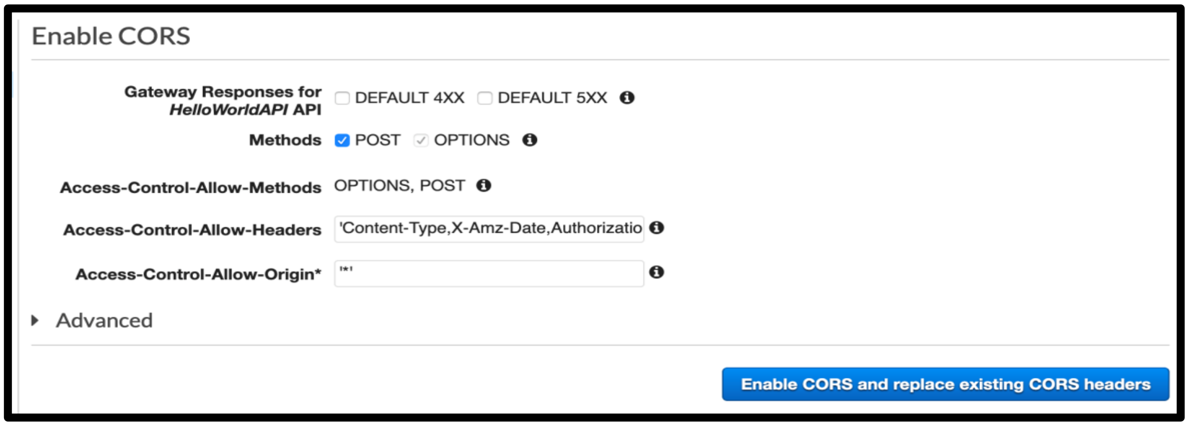
Choose a friendly name and description for your API.

API name*	HelloWorldAPI
Description	
Endpoint Type	Edge optimized

b)Create a new resource and method

1. In the left navigation pane, select Resources under API: HelloWorldAPI.
2. Ensure the "/" resource is selected.
3. From the Actions dropdown menu, select Create Method.
4. Select POST from the new dropdown that appears, then select the checkmark.
5. Select Lambda Function for the Integration type.
6. Select the Lambda Region you used when making the function (or else you will see a warning box reading "You do not have any Lambda Functions in...").
7. Enter HelloWorldFunction in the Lambda Function field.
8. Choose the blue Save button.
9. You should see a message letting you know you are giving the API you are creating permission to call your Lambda function. Choose the OK button.
10. With the newly created POST method selected, select Enable CORS from the Action dropdown menu.

11. Leave the POST checkbox selected and choose the blue Enable CORS and replace existing CORS headers button.



Enable CORS

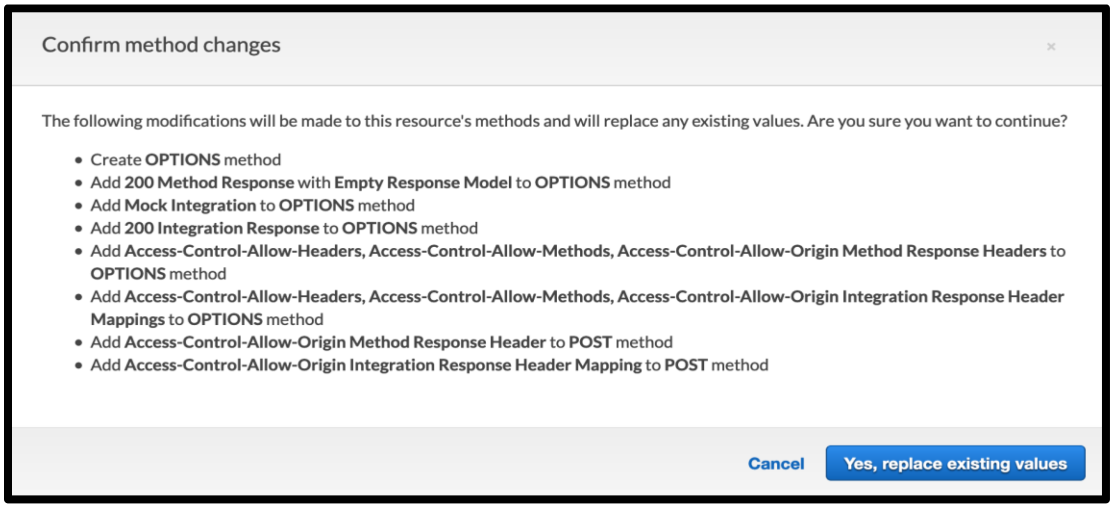
Gateway Responses for *HelloWorldAPI* API ☐ DEFAULT 4XX ☐ DEFAULT 5XX

Methods ☑ POST ☑ OPTIONS

Access-Control-Allow-Methods OPTIONS, POST

Access-Control-Allow-Headers 'Content-Type,X-Amz-Date,Authorizatio

Access-Control-Allow-Origin* '*'



Confirm method changes

The following modifications will be made to this resource's methods and will replace any existing values. Are you sure you want to continue?

- Create **OPTIONS** method
- Add **200 Method Response** with **Empty Response Model** to **OPTIONS** method
- Add **Mock Integration** to **OPTIONS** method
- Add **200 Integration Response** to **OPTIONS** method
- Add **Access-Control-Allow-Headers, Access-Control-Allow-Methods, Access-Control-Allow-Origin Method Response Headers** to **OPTIONS** method
- Add **Access-Control-Allow-Headers, Access-Control-Allow-Methods, Access-Control-Allow-Origin Integration Response Header Mappings** to **OPTIONS** method
- Add **Access-Control-Allow-Origin Method Response Header** to **POST** method
- Add **Access-Control-Allow-Origin Integration Response Header Mapping** to **POST** method

c)Deploy API

1. In the Actions dropdown list, select Deploy API.
2. Select [New Stage] in the Deployment stage dropdown list.
3. Enter dev for the Stage Name.
4. Choose Deploy.
5. Copy and save the URL next to Invoke URL

d)Validate API

1. In the left navigation pane, select Resources.
2. The methods for our API will now be listed on the right. Choose POST.

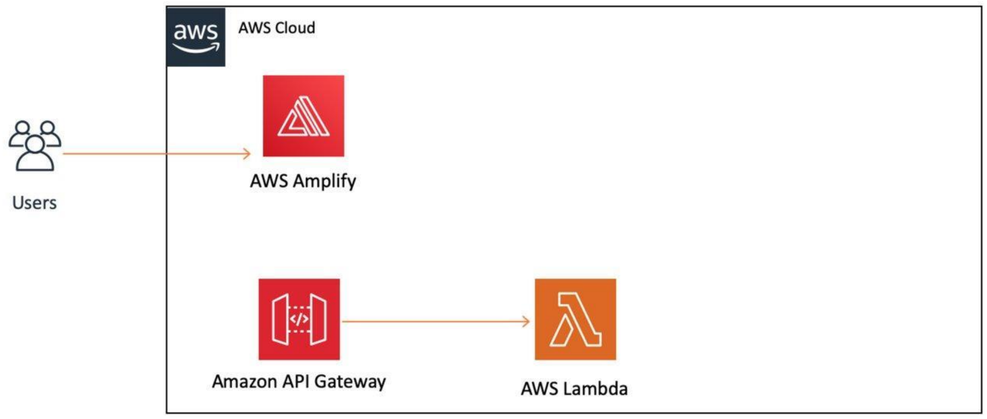
3. Choose the small blue lightning bolt.
4. Paste the following into the Request Body field:

```
{
   "firstName":"Grace",
   "lastName":"Hopper"
}
```

5. Choose the blue Test button.

6. On the right side, you should see a response with Code 200.

Great! We have built and tested an API that calls our Lambda function.

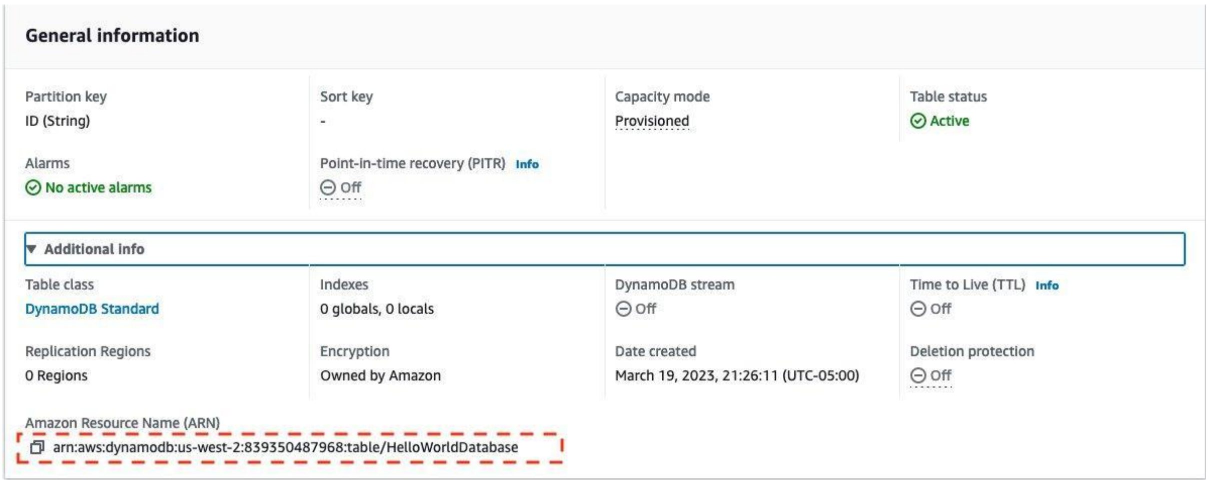


Step 4: Create a Data Table

a) Create a DynamoDB table

1. log in to the Amazon DynamoDB console.
2. Make sure you create your table in the same Region in which you created the web app in the previous module. You can see this at the very top of the page, next to your account name.
3. Choose the orange Create table button.
4. Under Table name, enter HelloWorldDatabase.
5. In the Partition key field, enter ID. The partition key is part of the table's primary key.
6. Leave the rest of the default values unchanged and choose the orange Create table button.

7. In the list of tables, select the table name, HelloWorldDatabase.
8. In the General information section, show Additional info by selecting the down arrow.



General information

Partition key	Sort key	Capacity mode	Table status
ID (String)	-	Provisioned	Active
Alarms	Point-in-time recovery (PITR) Info		
No active alarms	Off		

Additional info

Table class	Indexes	DynamoDB stream	Time to Live (TTL) Info
DynamoDB Standard	0 globals, 0 locals	Off	Off
Replication Regions	Encryption	Date created	Deletion protection
0 Regions	Owned by Amazon	March 19, 2023, 21:26:11 (UTC-05:00)	Off

Amazon Resource Name (ARN)
arn:aws:dynamodb:us-west-2:839350487968:table/HelloWorldDatabase

9. Copy the Amazon Resource Name (ARN). You will need it later.

b) Create and add IAM policy to Lambda function

1. Now that we have a table, let's edit our Lambda function to be able to write data to it. In a new browser window, open the AWS Lambda console.
2. Select the function we created in module two (if you have been using our examples, it will be called HelloWorldFunction). If you don't see it, check the Region dropdown in the upper right next to your name to ensure you're in the same Region you created the function in.
3. We'll be adding permissions to our function so it can use the DynamoDB service, and we will be using AWS Identity and Access Management (IAM) to do so.
4. Select the Configuration tab and select Permissions from the right side menu.
5. In the Execution role box, under Role name, choose the link. A new browser tab will open.
6. In the Permissions policies box, open the Add permissions dropdown and select Create inline policy.
7. Select the JSON tab.
8. Paste the following policy in the text area, taking care to replace your table's ARN in the Resource field in line 15:

```
{
"Version": "2012-10-17",
"Statement": [
   {
      "Sid": "VisualEditor0",
      "Effect": "Allow",
      "Action": [
         "dynamodb:PutItem",


         "dynamodb:DeleteItem",
         "dynamodb:GetItem",
         "dynamodb:Scan",
         "dynamodb:Query",
         "dynamodb:UpdateItem"
      ],
      "Resource": "YOUR-TABLE-ARN"
   }
   ]
}
```

9. This policy will allow our Lambda function to read, edit, or delete items, but restrict it to only be able to do so in the table we created.
10. Choose the blue Review Policy button.

11. Next to Name, enter HelloWorldDynamoPolicy.

12. Choose the blue Create Policy button.

13. You can now close this browser tab and go back to the tab for your Lambda function.

c) Modify Lambda function to write to DynamoDB table

1. Select the Code tab and select your function from the navigation pane on the left side of the code editor.
2. Replace the code for your function with the following:

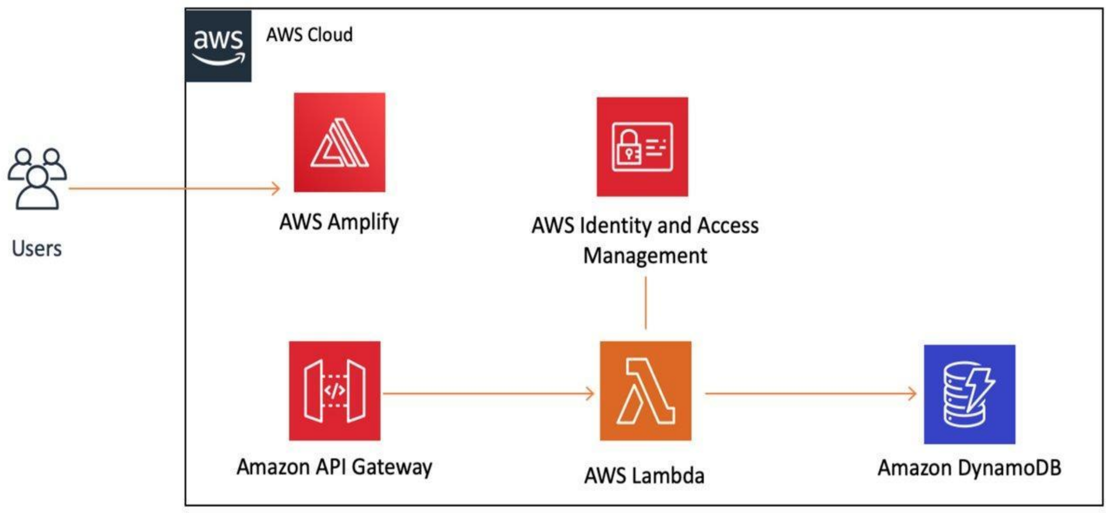
```
      # import the json utility package since we will be working with a JSON object

      import json
      # import the AWS SDK (for Python the package name is boto3)
      import boto3
      # import time
      import time
      # import two packages to help us with dates and date formatting
      # create a DynamoDB object using the AWS SDK
      dynamodb = boto3.resource('dynamodb')
      # use the DynamoDB object to select our table

      table = dynamodb.Table('HelloWorldDatabase')
      # define the handler function that the Lambda service will use as an entry point
      def lambda_handler(event, context):
       # Get the current GMT time
          gmt_time = time.gmtime()
 # store the current time in a human readable format in a variable
          # Format the GMT time string
          now = time.strftime('%a, %d %b %Y %H:%M:%S +0000', gmt_time)
# extract values from the event object we got from the Lambda service and store in a variable
          name = event['firstName'] +' '+ event['lastName']
      # write name and time to the DynamoDB table using the object we instantiated and save
      response in a variable
          response = table.put_item(
              Item={
                  'ID': name,
                  'LatestGreetingTime':now
                  })
      # return a properly formatted JSON object
          return {
```

```
        'statusCode': 200,
        'body': json.dumps('Hello from Lambda, ' + name)
    }
```

3. Choose the Deploy button at the top of the code editor

d)Test your changes

1. Choose the orange Test button.
2. You should see an Execution result: succeeded message with a green background.
3. In a new browser tab, open the DynamoDB console.
4. In the left-hand navigation pane, select Tables > Explore items.
5. Select HelloWorldDatabase, which we created earlier in this module.
6. Select the Items tab on the right.
7. Items matching your test event appear under Items returned. If you have been using our examples, the item ID will be Hello from Lambda, Ada Lovelace or Ada Lovelace.
8. Every time your Lambda function executes, your DynamoDB table will be updated. If the same name is used, only the time stamp will change.



Step 5: Add Interactivity to Your Web App

a) Update Web app with Amplify console:

1. Open the index.html file you created in module one.
2. Replace the existing code with the following:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="UTF-8">
  <title>Hello World</title>
  <!-- Add some CSS to change client UI -->
  <style>
  body {
    background-color: #232F3E;
    }
  label, button {
    color: #FF9900;
    font-family: Arial, Helvetica, sans-serif;
    font-size: 20px;
    margin-left: 40px;
    }

   input {
    color: #232F3E;
    font-family: Arial, Helvetica, sans-serif;
    font-size: 20px;
    margin-left: 20px;
    }
  </style>
  <script>
    // define the callAPI function that takes a first name and last name as parameters
    var callAPI = (firstName,lastName)=>{
      // instantiate a headers object
```

```
            var myHeaders = new Headers();
            // add content type header to object
            myHeaders.append("Content-Type", "application/json");
            // using built in JSON utility package turn object to string and store in a variable
            var raw = JSON.stringify({"firstName":firstName,"lastName":lastName});
            // create a JSON object with parameters for API call and store in a variable
            var requestOptions = {
                method: 'POST',
                headers: myHeaders,
                body: raw,
                redirect: 'follow'
            };
            // make API call with parameters and use promises to get response
            fetch("YOUR-API-INVOKE-URL", requestOptions)
            .then(response => response.text())
            .then(result => alert(JSON.parse(result).body))
            .catch(error => console.log('error', error));
        }
    </script>
</head>
<body>
    <form>
        <label>First Name :</label>

        <input type="text" id="fName">
        <label>Last Name :</label>
        <input type="text" id="lName">
        <!-- set button onClick method to call function we defined passing input values as
parameters -->
```

```
        <button type="button"
onclick="callAPI(document.getElementById('fName').value,document.getElementById('lNa
me').value)">Call API</button>
    </form>
</body>
</html>
```

3. Make sure you add your API Invoke URL on Line 41 (from module three). Note: If you do not have your API's URL, you can get it from the API Gateway console by selecting your API and choosing stages.

4. Save the file.

5. ZIP (compress) only the HTML file.

6. Open the Amplify console.

7. Choose the web app created in module one.

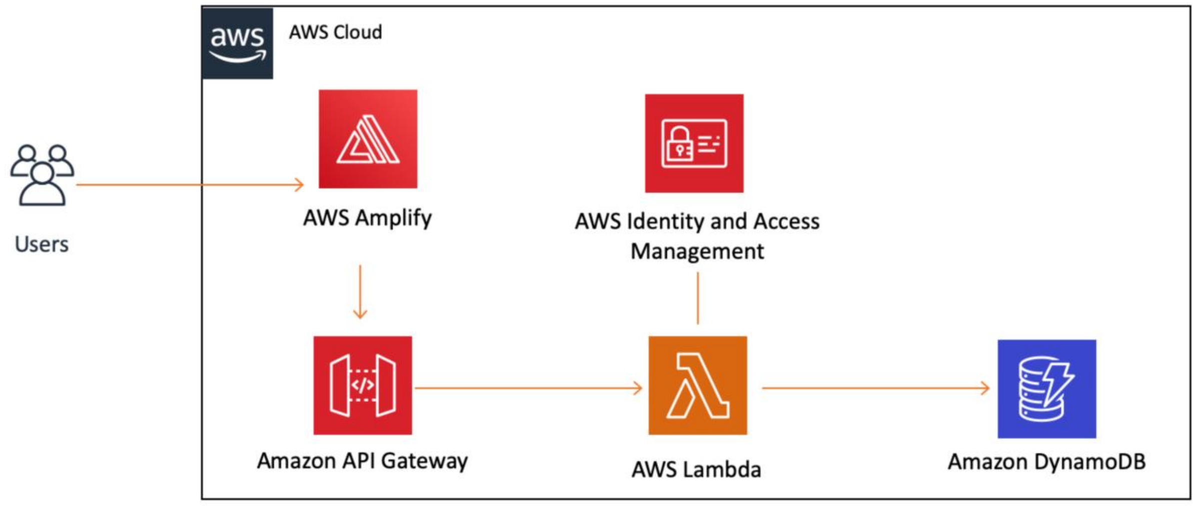
8. Choose the white Choose files button.

9. Select the ZIP file you created in Step 5.

10. When the file is uploaded, a deployment process will automatically begin. Once you see a green bar, your deployment will be complete.

b) Test the updated App:

1. Choose the URL under Domain.
2. Your updated web app should load in your browser.
3. Fill in your name (or whatever you prefer) and choose the Call API button.
4. You should see a message that starts with Hello from Lambda followed by the text you filled in.



RESULT:

Create a Cost-model for a web application using various services and do Cost-benefit analysis has been executed successfully and the output is verified.

EXP. NO: 03 **DATE:**	**BILLING ALERT**

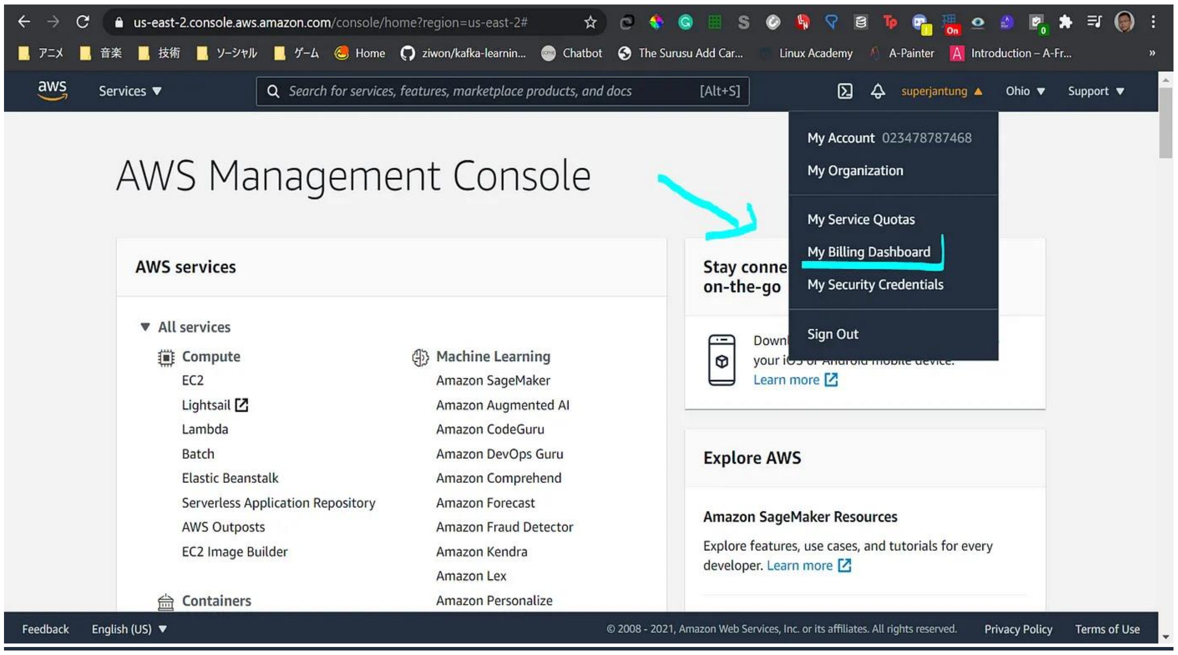
AIM:

To Create Billing alerts for your Cloud Organization.

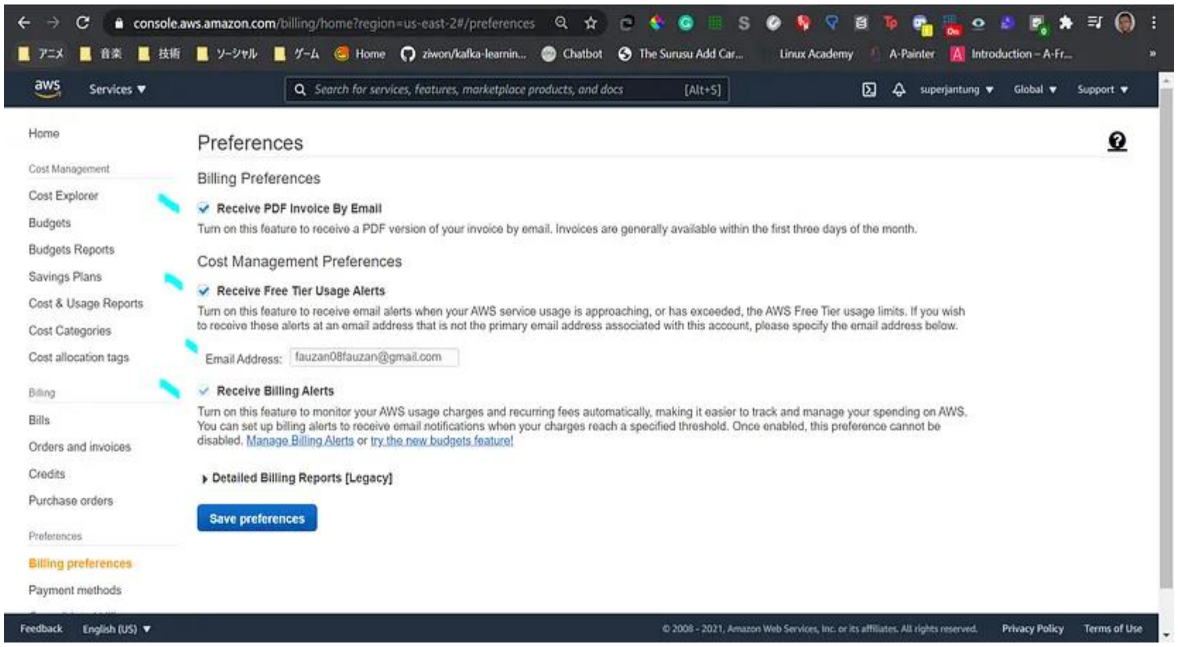
PROCEDURE:

a) Set Billing Preferences

Step-1: Go to the AWS console and login using your credentials. After that, go to the My Billing Dashboard menu.



Step-2: Go to the Billing Preferences and check on Receive PDF Invoice by Email and Receive Billing Alerts. Also, check on Receive Free Tier Usage Alerts and input your email if you are using AWS Free Tier.



Preferences

Billing Preferences

✓ Receive PDF Invoice By Email

Turn on this feature to receive a PDF version of your invoice by email. Invoices are generally available within the first three days of the month.

Cost Management Preferences

✓ Receive Free Tier Usage Alerts

Turn on this feature to receive email alerts when your AWS service usage is approaching, or has exceeded, the AWS Free Tier usage limits. If you wish to receive these alerts at an email address that is not the primary email address associated with this account, please specify the email address below.

Email Address: fauzan08fauzan@gmail.com

✓ Receive Billing Alerts

Turn on this feature to monitor your AWS usage charges and recurring fees automatically, making it easier to track and manage your spending on AWS. You can set up billing alerts to receive email notifications when your charges reach a specified threshold. Once enabled, this preference cannot be disabled. Manage Billing Alerts or try the new budgets feature!

▸ Detailed Billing Reports [Legacy]

Save preferences

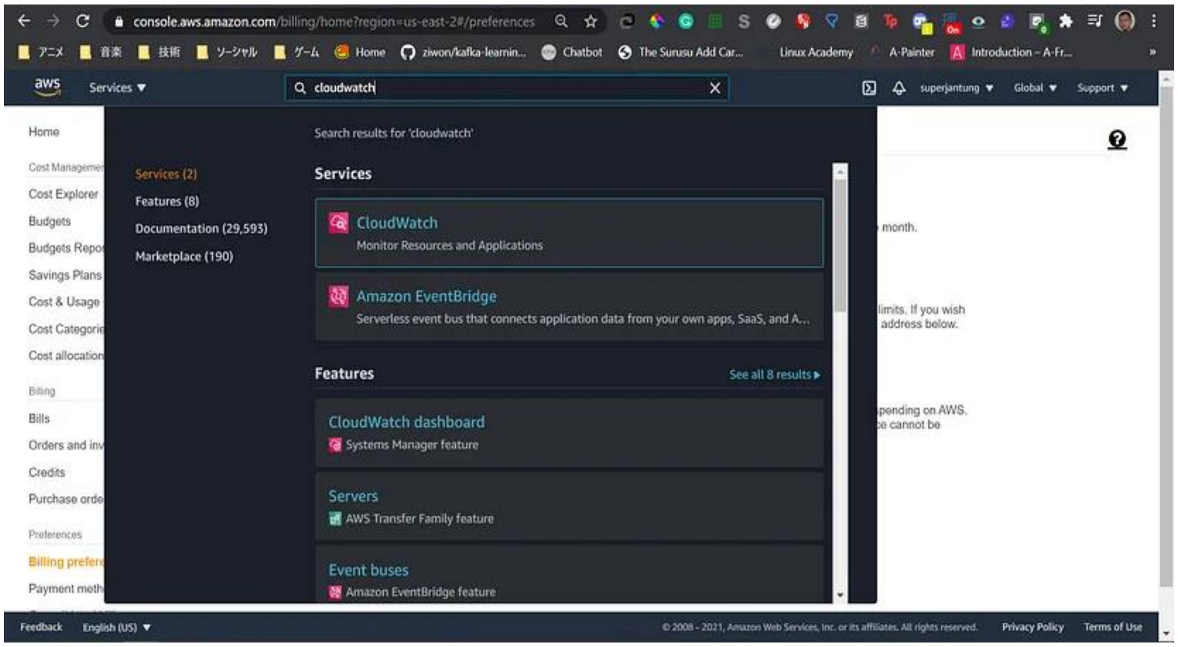
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.

Step-3: Click Save Preferences, after that you will receive email notifications when your charges reach a specified threshold.

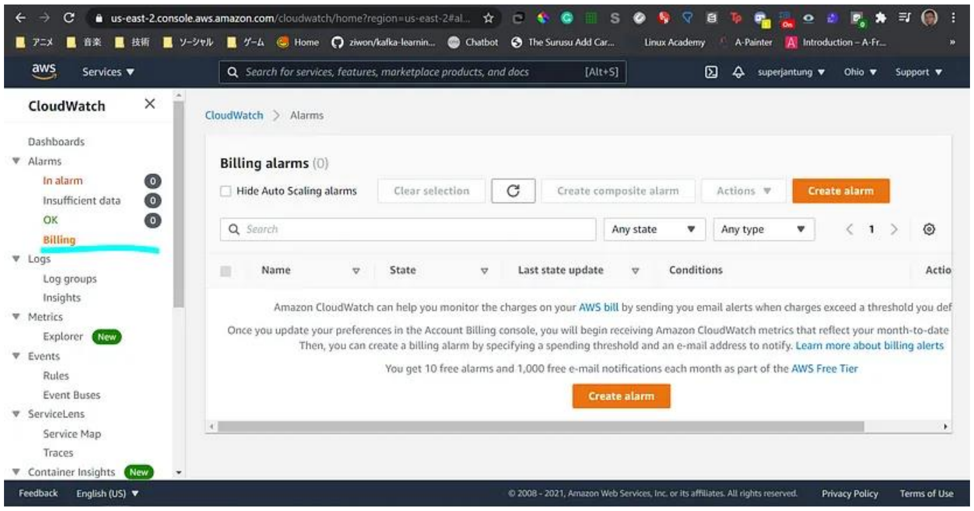
b) Create a Billing Alarm Using Amazon CloudWatch

Amazon CloudWatch is basically a monitoring service, so you can monitor your AWS services and trigger an alarm when something unexpected happens.

Step-1: Search CloudWatch on the search bar


cloudwatch
Search results for 'cloudwatch'
Services (2)
Features (8)
Documentation (29,593)
Marketplace (190)
Services
CloudWatch
Monitor Resources and Applications
Amazon EventBridge
Serverless event bus that connects application data from your own apps, SaaS, and A...
Features
See all 8 results
CloudWatch dashboard
Systems Manager feature
Servers
AWS Transfer Family feature
Event buses
Amazon EventBridge feature

Step-2: Go to the billing menu and click create alarm

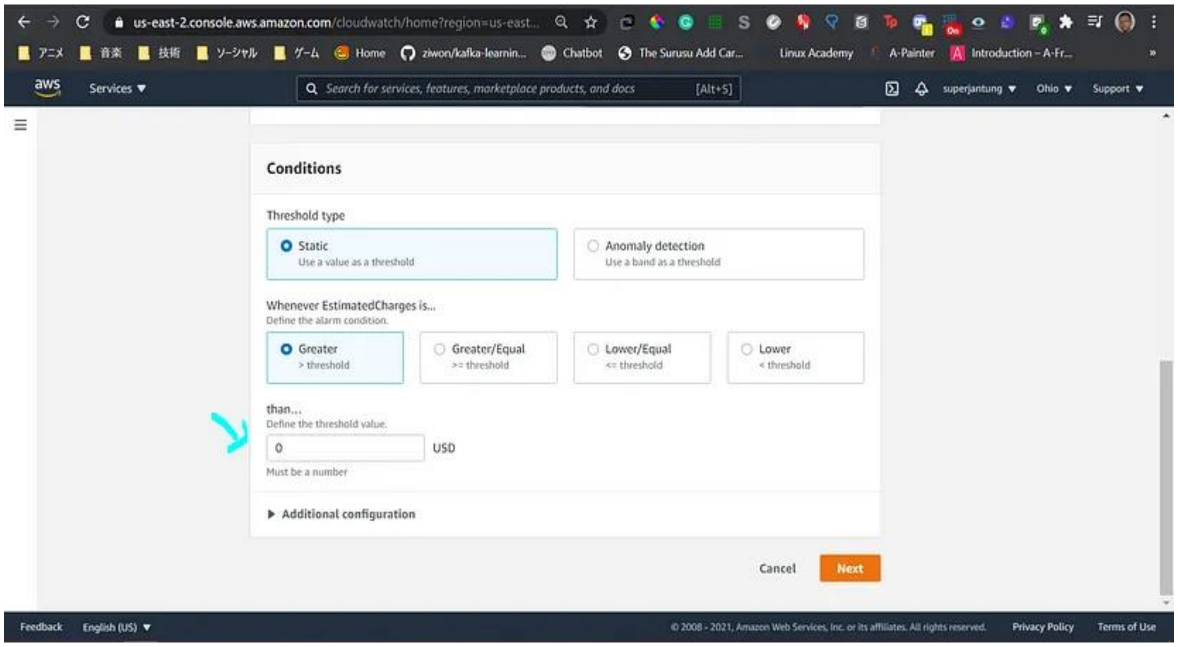


Amazon CloudWatch can help you monitor the charges on your AWS bill by sending you email alerts when charges exceed a threshold you def

Once you update your preferences in the Account Billing console, you will begin receiving Amazon CloudWatch metrics that reflect your month-to-date Then, you can create a billing alarm by specifying a spending threshold and an e-mail address to notify. Learn more about billing alerts

You get 10 free alarms and 1,000 free e-mail notifications each month as part of the AWS Free Tier

Step-3: You can set up the condition. For example, if you are using AWS free tier, you can set the value to 0. So you will get an alert if your estimated charges are greater than $0. After you finished setting up the condition, click next.


Conditions

Threshold type

- [✓ Static] Use a value as a threshold
- Anomaly detection — Use a band as a threshold

Whenever EstimatedCharges is...
Define the alarm condition.

- [✓ Greater] > threshold
- Greater/Equal >= threshold
- Lower/Equal <= threshold
- Lower < threshold

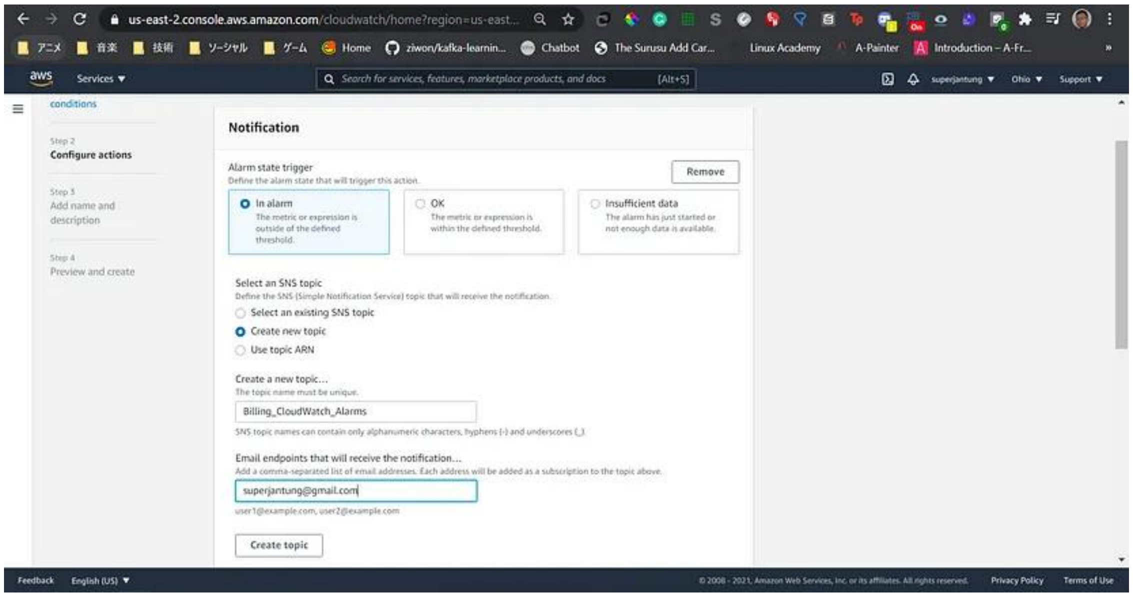
than...
Define the threshold value.

0 USD

Must be a number

Additional configuration

Step-4: When an alarm is triggered we want to create a new topic. So you can select in alarm trigger, select create a new topic for the SNS topic, input the topic name, input your email that will receive the notification, click create topic and click next


Notification

Alarm state trigger
Define the alarm state that will trigger this action.

- [✓ In alarm] The metric or expression is outside of the defined threshold.
- OK — The metric or expression is within the defined threshold.
- Insufficient data — The alarm has just started or not enough data is available.

Select an SNS topic
Define the SNS (Simple Notification Service) topic that will receive the notification.

- Select an existing SNS topic
- [✓ Create new topic]
- Use topic ARN

Create a new topic...
The topic name must be unique.

Billing_CloudWatch_Alarms

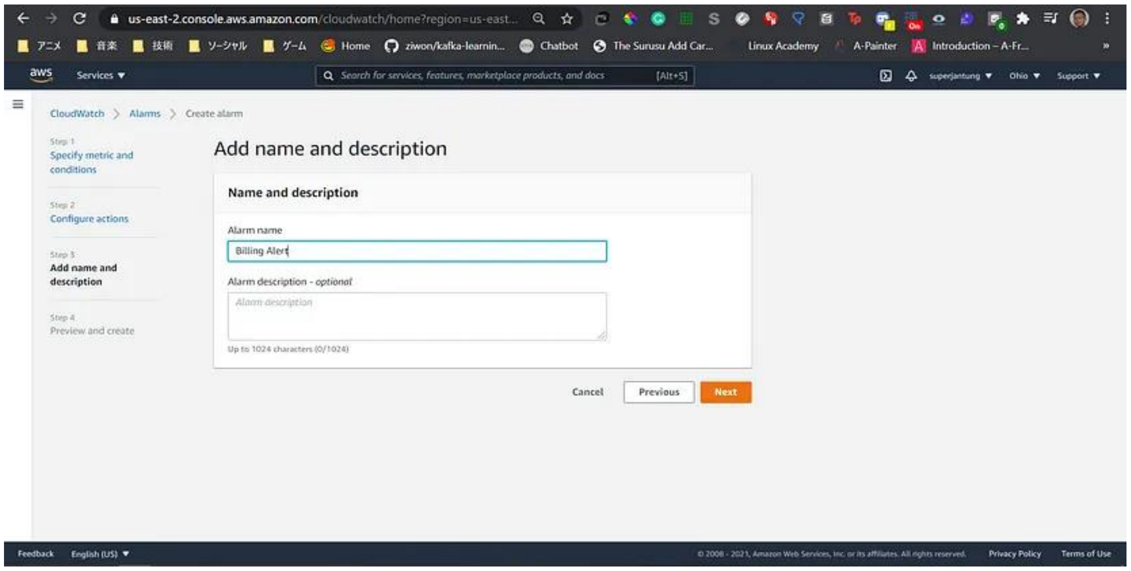
SNS topic names can contain only alphanumeric characters, hyphens (-) and underscores (_).

Email endpoints that will receive the notification...
Add a comma-separated list of email addresses. Each address will be added as a subscription to the topic above.

superjantung@gmail.com

user1@example.com, user2@example.com

Step-5: Input alarm name. For example, Billing alert. After that, click next.


us-east-2.console.aws.amazon.com/cloudwatch/home?region=us-east...
Services
Search for services, features, marketplace products, and docs [Alt+S]
CloudWatch > Alarms > Create alarm
Step 1
Specify metric and conditions
Step 2
Configure actions
Step 3
Add name and description
Step 4
Preview and create
Cancel
Previous
Next
Feedback
English (US)
Privacy Policy
Terms of Use

Add name and description

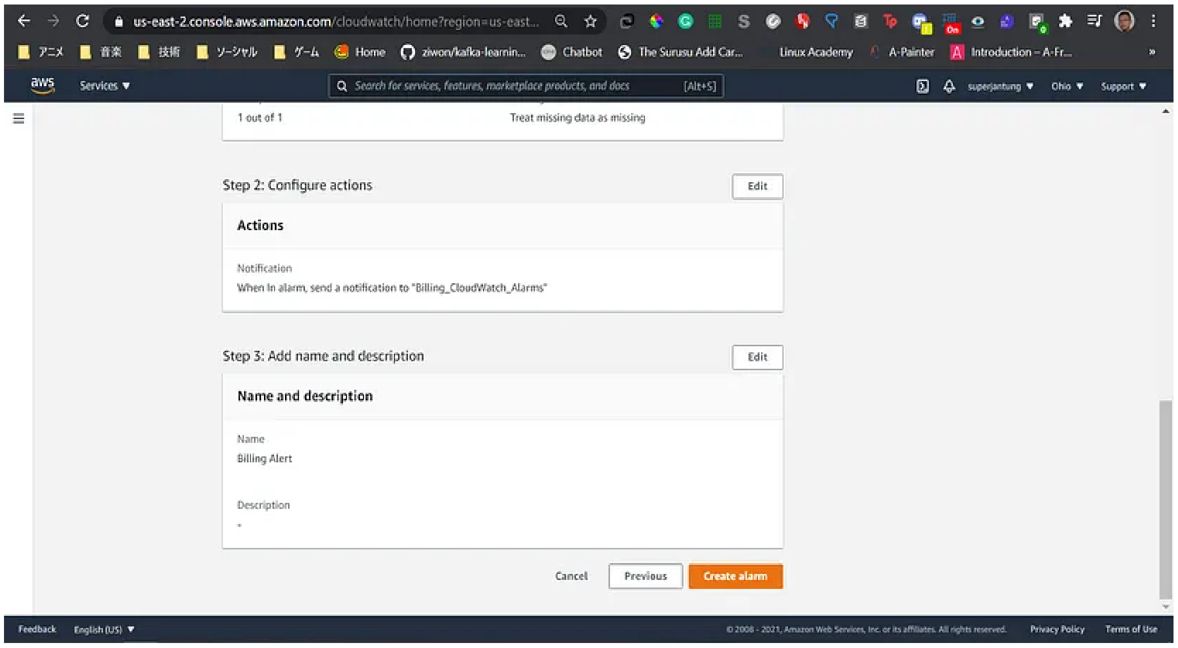
Name and description

Alarm name
Billing Alert

Alarm description - optional
Alarm description

Up to 1024 characters (0/1024)

Step-6: You can preview everything that you have input. If you are satisfied, click create alarm


us-east-2.console.aws.amazon.com/cloudwatch/home?region=us-east...
Services
Search for services, features, marketplace products, and docs [Alt+S]
Edit
Edit
Cancel
Previous
Create alarm
Feedback
English (US)
Privacy Policy
Terms of Use

1 out of 1 Treat missing data as missing

Step 2: Configure actions

Actions

Notification
When In alarm, send a notification to "Billing_CloudWatch_Alarms"

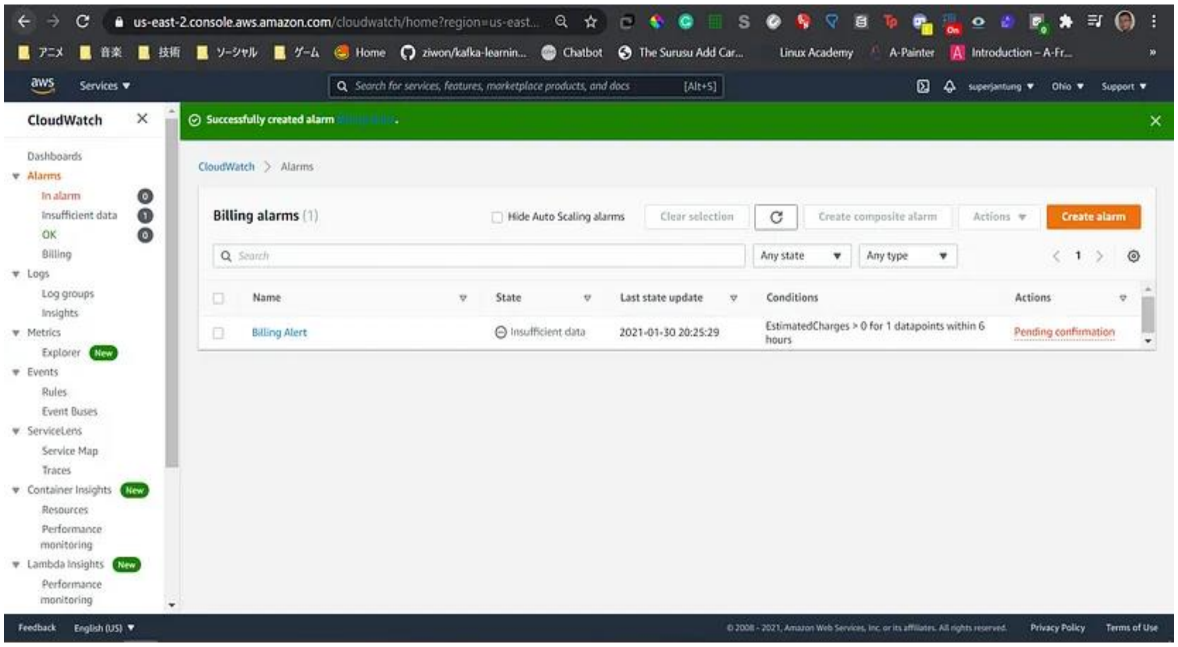
Step 3: Add name and description

Name and description

Name
Billing Alert

Description
-

If successful, you will see an alert "Successfully created alarm" and the billing alarm is created ,



Successfully created alarm [illegible].

CloudWatch > Alarms

Billing alarms (1)

Name	State	Last state update	Conditions	Actions
Billing Alert	Insufficient data	2021-01-30 20:25:29	EstimatedCharges > 0 for 1 datapoints within 6 hours	Pending confirmation

RESULT:

Create Billing alerts for your Cloud Organization has been created executed successfully and the output is verified.

EXP. NO: 04 **DATE:**	**SETTING UP BUDGET ALERT**

AIM:

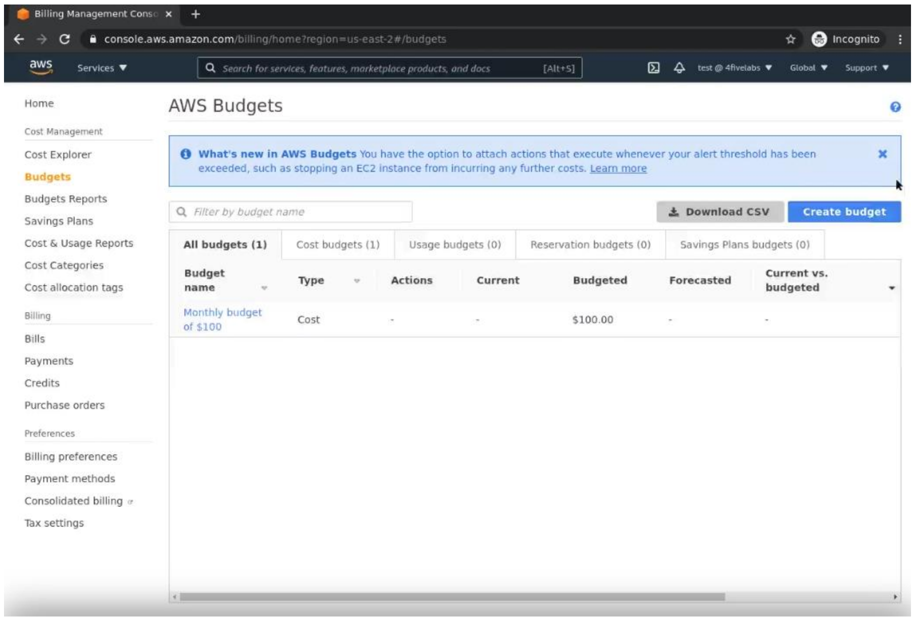
To Create alerts for usage of Cloud resources.

PROCEDURE:

<u>**Budget Alerts**</u>

Step-1 : Go to the billing section, explore details like EC2 usage, instance types, and data transfer metrics to understand the diverse usage measurements (hours vs. GB).

Step-2 : Head back to the budget section and click on "Create budget." and Select "Usage Budget" and proceed to set up the budget details.



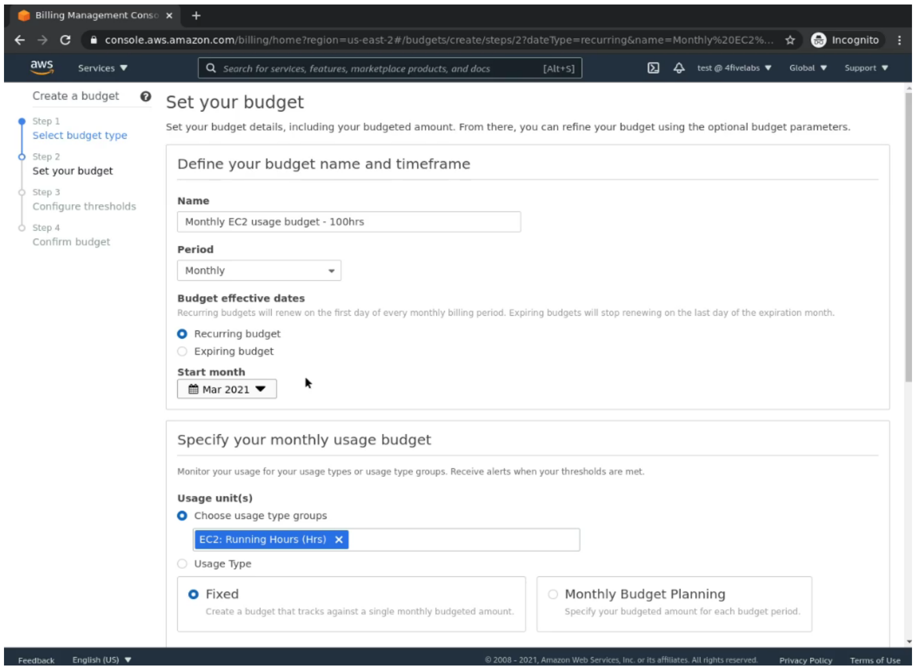
What's new in AWS Budgets You have the option to attach actions that execute whenever your alert threshold has been exceeded, such as stopping an EC2 instance from incurring any further costs. Learn more

Budget name	Type	Actions	Current	Budgeted	Forecasted	Current vs. budgeted
Monthly budget of $100	Cost	-	-	$100.00	-	-

Step-3 : Define the budget name (e.g., "Monthly EC2 Usage Budget") and specify the usage limit (e.g., 100 hours for EC2 instances) and the monitoring period (e.g., monthly).

Step-4 : opt for a recurring budget starting from a chosen month (e.g. March).

Step-5 : Define the usage type (e.g., EC2 running hours) and set the budget amount (e.g., 100 hours).


Billing Management Conso
console.aws.amazon.com/billing/home?region=us-east-2#/budgets/create/steps/2?dateType=recurring&name=Monthly%20EC2%...
Incognito
Services
Search for services, features, marketplace products, and docs [Alt+S]
test @ 4fivelabs
Global
Support
Create a budget
Step 1 Select budget type
Step 2 Set your budget
Step 3 Configure thresholds
Step 4 Confirm budget
Feedback
English (US)
© 2008 - 2021, Amazon Web Services, Inc. or its affiliates. All rights reserved.
Privacy Policy
Terms of Use

Set your budget

Set your budget details, including your budgeted amount. From there, you can refine your budget using the optional budget parameters.

Define your budget name and timeframe

Name: Monthly EC2 usage budget - 100hrs

Period: Monthly

Budget effective dates

Recurring budgets will renew on the first day of every monthly billing period. Expiring budgets will stop renewing on the last day of the expiration month.

[✓ Recurring budget] Expiring budget

Start month: Mar 2021

Specify your monthly usage budget

Monitor your usage for your usage types or usage type groups. Receive alerts when your thresholds are met.

Usage unit(s)

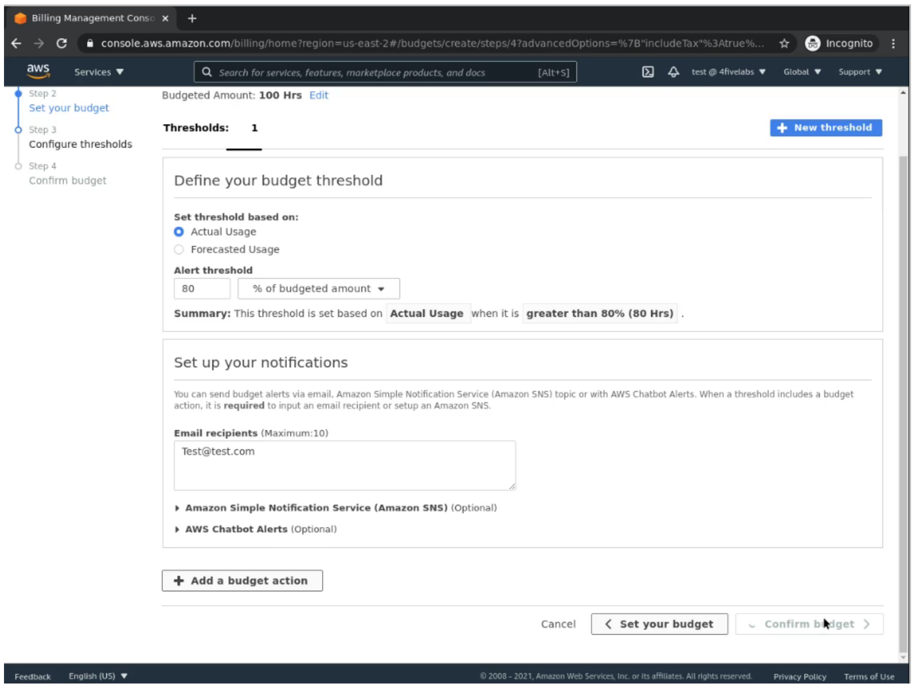
[✓ Choose usage type groups]: EC2: Running Hours (Hrs)

Usage Type

[✓ Fixed] Create a budget that tracks against a single monthly budgeted amount.

Monthly Budget Planning Specify your budgeted amount for each budget period.

Step-6 : Configure the threshold for alerts (e.g., 80% of the defined budget) and Add email recipients for alerts.



Step 2
Set your budget
Step 3
Configure thresholds
Step 4
Confirm budget

Budgeted Amount: **100 Hrs** Edit

Thresholds: 1

Define your budget threshold

Set threshold based on:
Actual Usage [✓]
Forecasted Usage

Alert threshold
80 | % of budgeted amount

Summary: This threshold is set based on **Actual Usage** when it is **greater than 80% (80 Hrs)** .

Set up your notifications

You can send budget alerts via email, Amazon Simple Notification Service (Amazon SNS) topic or with AWS Chatbot Alerts. When a threshold includes a budget action, it is **required** to input an email recipient or setup an Amazon SNS.

Email recipients (Maximum:10)
Test@test.com

Amazon Simple Notification Service (Amazon SNS) (Optional)
AWS Chatbot Alerts (Optional)

Step-7 : Review the budget summary. Click on "Create" to establish the usage budget alert successfully.

Verify the created usage budget in the AWS Cost Explorer and check other details.

RESULT:

Create alerts for usage of Cloud resources has been executed successfully and the output is verified.

EXP. NO: 05 **DATE:**	**DEPLOYING A WEB APPLICATION USING S3**

AIM:

To Deploy a Web Application using S3.

PROCEDURE:

S3:

Deploying a web application on AWS S3 involves creating a bucket for resources, uploading files, configuring static web hosting, managing access controls, and verifying functionality through the provided endpoint URL.

S3
S3 bucket
Website

Overview:

1.) Create an AWS S3 bucket to store static web application resources.

2.) Upload HTML, CSS, JavaScript, and other static files constituting the web application to the designated S3 bucket.

3.) Configure the S3 bucket for static website hosting to allow direct access to the web application files via an endpoint URL.

4.) Fine-tune access controls and permissions within the S3 bucket to manage user access and security settings for the deployed web application.

5.) Verify the deployment by accessing the web application through the provided endpoint URL to ensure proper functionality and accessibility.

Step 1: Create an AWS S3 bucket:

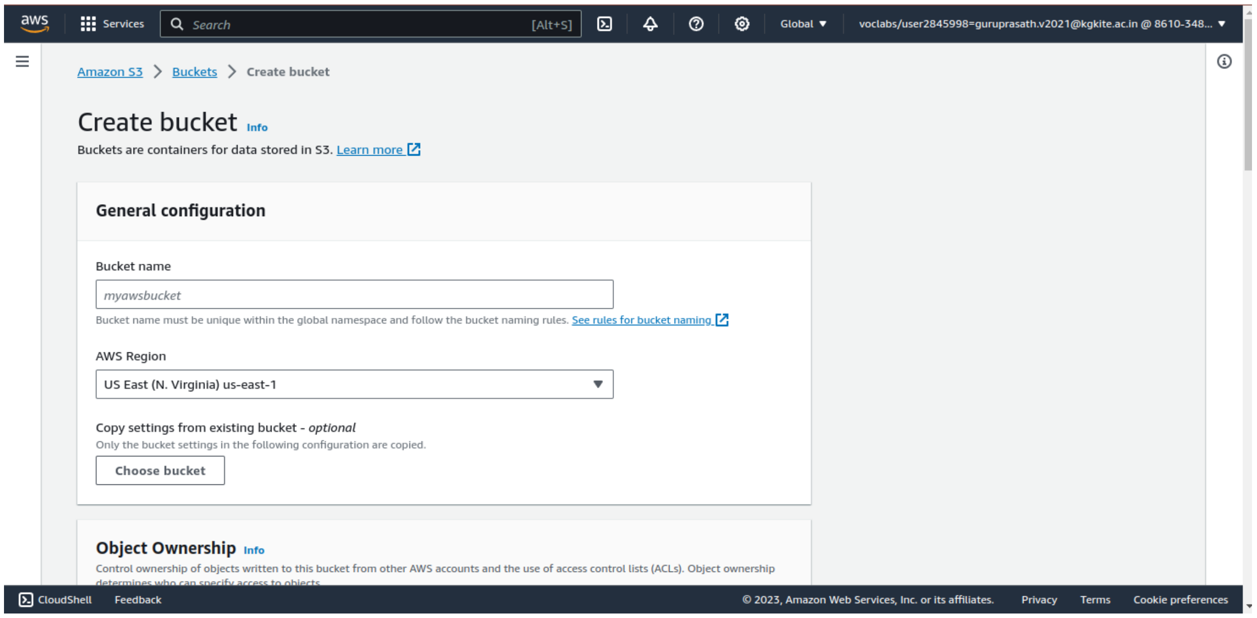
1.) In a new browser tab, log in to the AWS console.

2.) In there search for S3 and click the link with S3 with description of scalable storage in the cloud.

3.) In that UI you will see the Buckets section.

4.) In that section click "Create Bucket".

5.) After clicked create bucket button, you will see the UI for configuration.


aws
Services
Search
[Alt+S]
Global
voclabs/user2845998=guruprasath.v2021@kgkite.ac.in @ 8610-348...
Amazon S3 > Buckets > Create bucket

Create bucket Info

Buckets are containers for data stored in S3. Learn more

General configuration

Bucket name

myawsbucket

Bucket name must be unique within the global namespace and follow the bucket naming rules. See rules for bucket naming

AWS Region

US East (N. Virginia) us-east-1

Copy settings from existing bucket - *optional*

Only the bucket settings in the following configuration are copied.

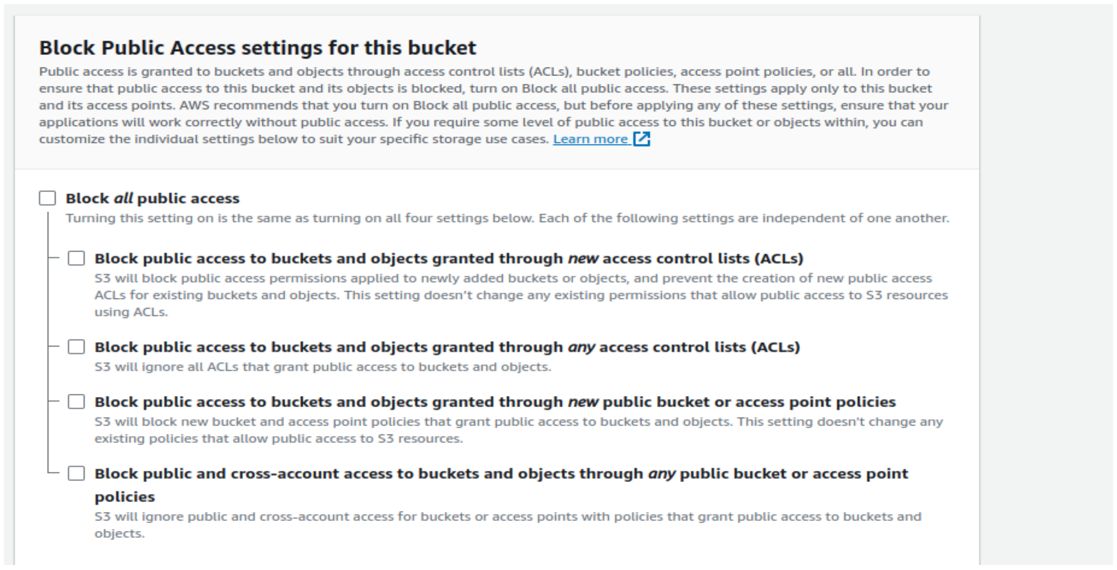
Choose bucket

Object Ownership Info

Control ownership of objects written to this bucket from other AWS accounts and the use of access control lists (ACLs). Object ownership determines who can specify access to objects.

6.) Choose a unique bucket name (it must be globally unique), select your preferred AWS region (Leave it default).

7.) Under Block public access section make sure to untick the checkbox showing block all public access



Block Public Access settings for this bucket

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to this bucket and its objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to this bucket or objects within, you can customize the individual settings below to suit your specific storage use cases. Learn more

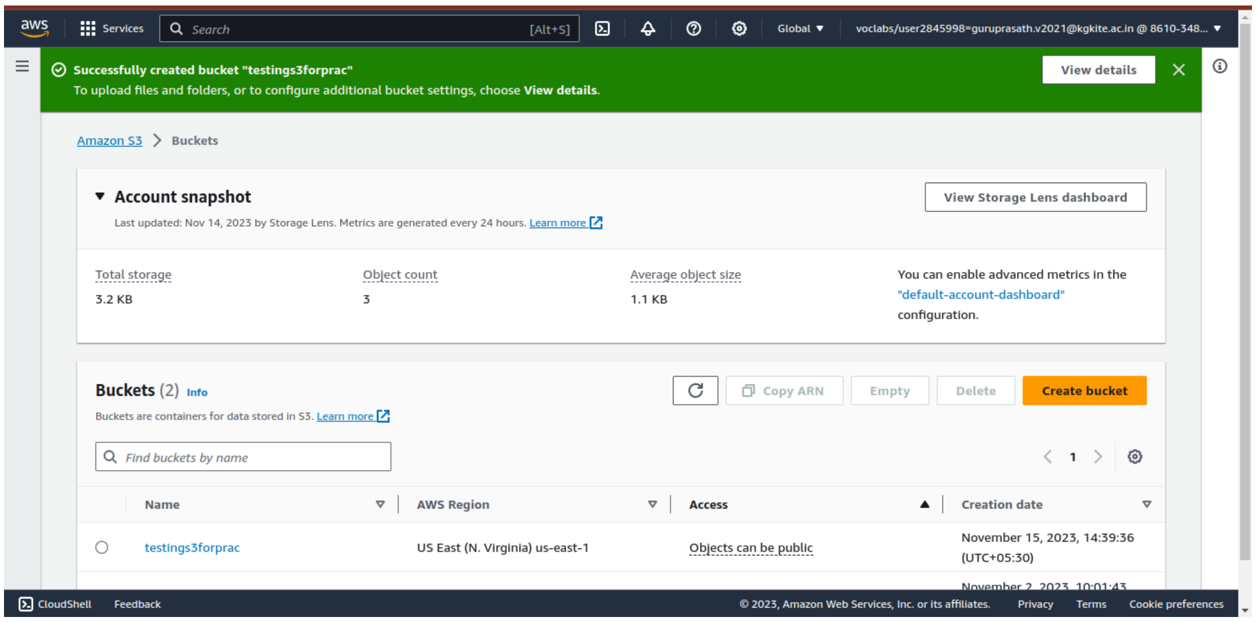
☐ **Block *all* public access**
Turning this setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

- ☐ **Block public access to buckets and objects granted through *new* access control lists (ACLs)**
  S3 will block public access permissions applied to newly added buckets or objects, and prevent the creation of new public access ACLs for existing buckets and objects. This setting doesn't change any existing permissions that allow public access to S3 resources using ACLs.
- ☐ **Block public access to buckets and objects granted through *any* access control lists (ACLs)**
  S3 will ignore all ACLs that grant public access to buckets and objects.
- ☐ **Block public access to buckets and objects granted through *new* public bucket or access point policies**
  S3 will block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.
- ☐ **Block public and cross-account access to buckets and objects through *any* public bucket or access point policies**
  S3 will ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

8.) And tick the checkbox inside the alert box saying "I acknowledge that the current settings might result in this bucket and the objects within becoming public".

9.) Now click create bucket button in the bottom.

10.) Now you can see this alert like "successfully created" and you can also view details of your bucket by clicking view details button.



Successfully created bucket "testings3forprac"
To upload files and folders, or to configure additional bucket settings, choose **View details**.

Amazon S3 > Buckets

Account snapshot
Last updated: Nov 14, 2023 by Storage Lens. Metrics are generated every 24 hours. Learn more

Total storage	Object count	Average object size
3.2 KB	3	1.1 KB

You can enable advanced metrics in the "default-account-dashboard" configuration.

Buckets (2) Info
Buckets are containers for data stored in S3. Learn more

Name	AWS Region	Access	Creation date
testings3forprac	US East (N. Virginia) us-east-1	Objects can be public	November 15, 2023, 14:39:36 (UTC+05:30)
			November 2, 2023, 10:01:43

© 2023, Amazon Web Services, Inc. or its affiliates.

Step 2: Upload Files into Bucket:

1.) Under the list of your buckets. Click the bucket you have now created.
2.) Then it will show the objects section. In this section we gonna upload web files for hosting and deploying.
3.) Open any code editor and create three files name it as “index.html”, “style.css”.
4.) Add this code in html file:

```
<!DOCTYPE html>
<html lang="en">
<head>
 <meta charset="UTF-8">
 <meta name="viewport" content="width=device-width, initial-scale=1.0">
 <meta http-equiv="X-UA-Compatible" content="ie=edge">
 <title>Recording</title>
 <link rel="stylesheet" href="main.css">
</head>
```

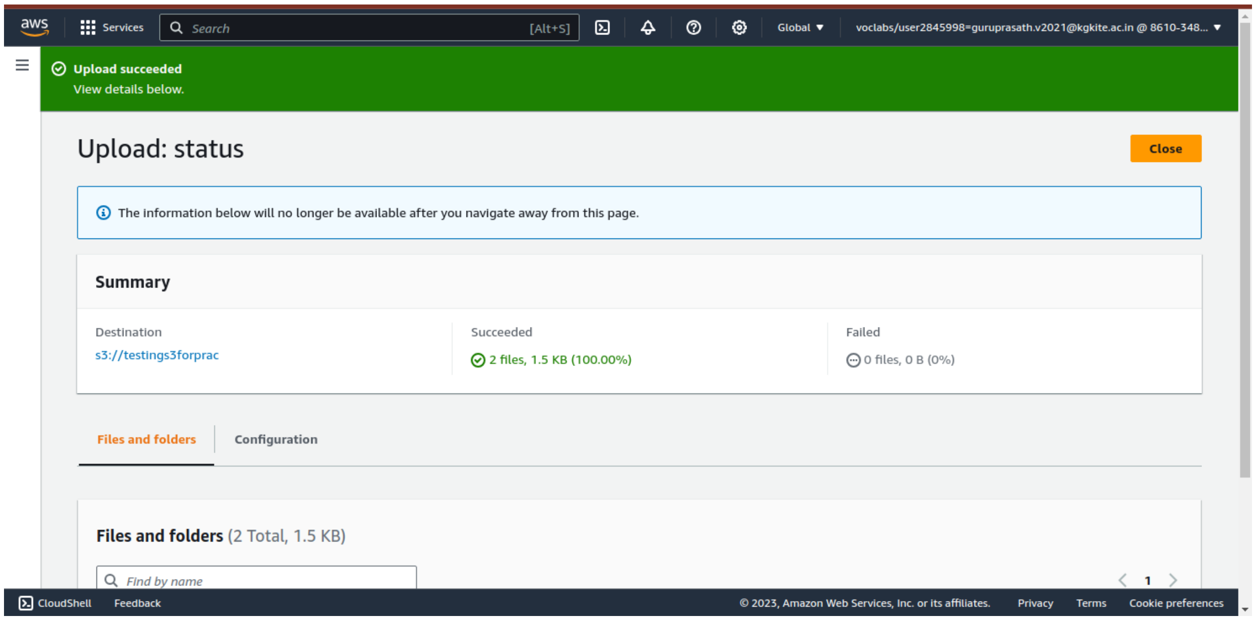
```
<body>
 <div class="center">
    SUCCESSFULLY DEPLOYED IN <span>"AWS"</span> S3 BUCKET
 </div>
</body>
</html>
```

5.) Add this following code in your css file:

```
@import url(https://fonts.googleapis.com/css?family=Agbalumo);
@import url(https://fonts.googleapis.com/css?family=Nunito);
{
 margin: 0;
 padding: 0;
 box-sizing: border-box;
}
.center{
    margin-left: auto;
    margin-right: auto;
    background-color: hsl(119, 40%, 51%);
    color: rgb(41, 37, 37);
    width: max-content;
    height: 5rem;
    font-size: 2em;
    padding: 5em;
    font-family: 'Agbalumo';
}
.center span{
    color: white;
    font-family: 'Nunito';
}
```

6.) Come to S3 object space and click upload button and click “Add files” and then select the html and css files and click upload.

7.) After uploading the files you will see this success box.



aws
Services
Search
[Alt+S]
Global
voclabs/user2845998=guruprasath.v2021@kgkite.ac.in @ 8610-348...

Upload succeeded
View details below.

Upload: status

The information below will no longer be available after you navigate away from this page.

Summary

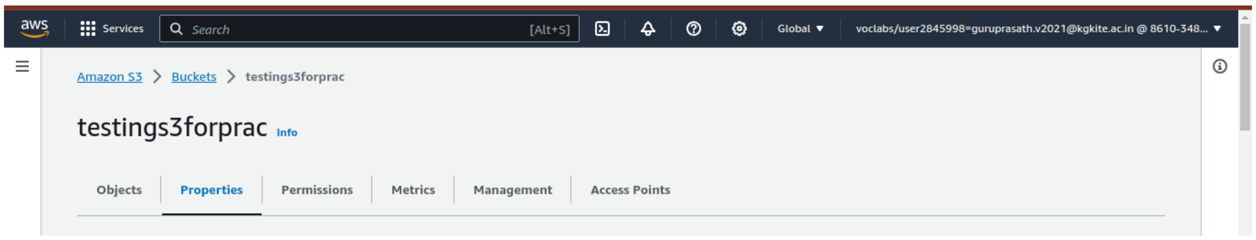
Destination	Succeeded	Failed
s3://testings3forprac	2 files, 1.5 KB (100.00%)	0 files, 0 B (0%)

Files and folders (2 Total, 1.5 KB)

8.) Now you can see the files that you uploaded.

Step 3: Configure S3 bucket for static web hosting:

1) In your S3 bucket, go to the "Properties" tab.

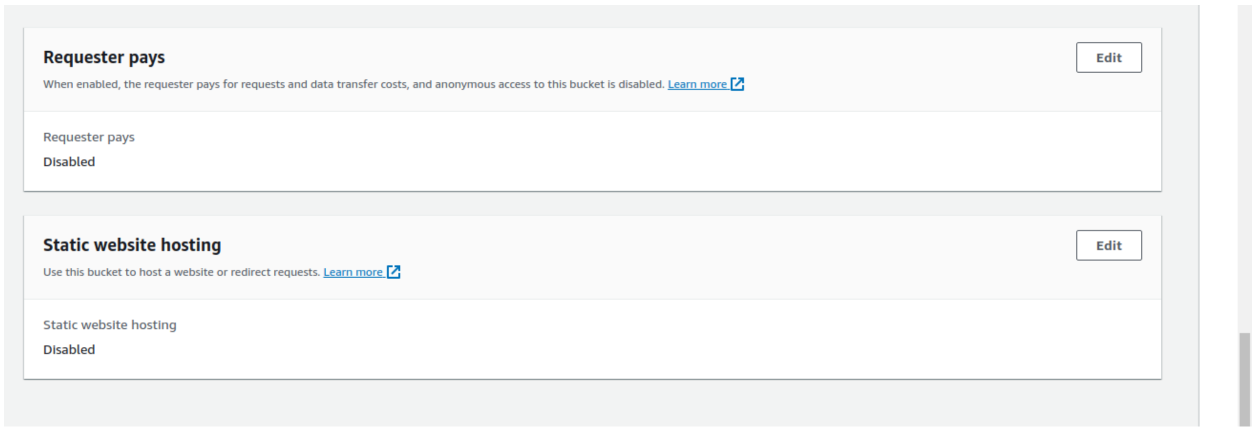

aws Services Search [Alt+S] Global voclabs/user2845998=guruprasath.v2021@kgkite.ac.in @ 8610-348...

Amazon S3 > Buckets > testings3forprac

testings3forprac Info

Objects | Properties | Permissions | Metrics | Management | Access Points

2) Scroll down to the "Static website hosting" section.


Requester pays

When enabled, the requester pays for requests and data transfer costs, and anonymous access to this bucket is disabled. Learn more

Edit

Requester pays

Disabled

Static website hosting

Use this bucket to host a website or redirect requests. Learn more

Edit

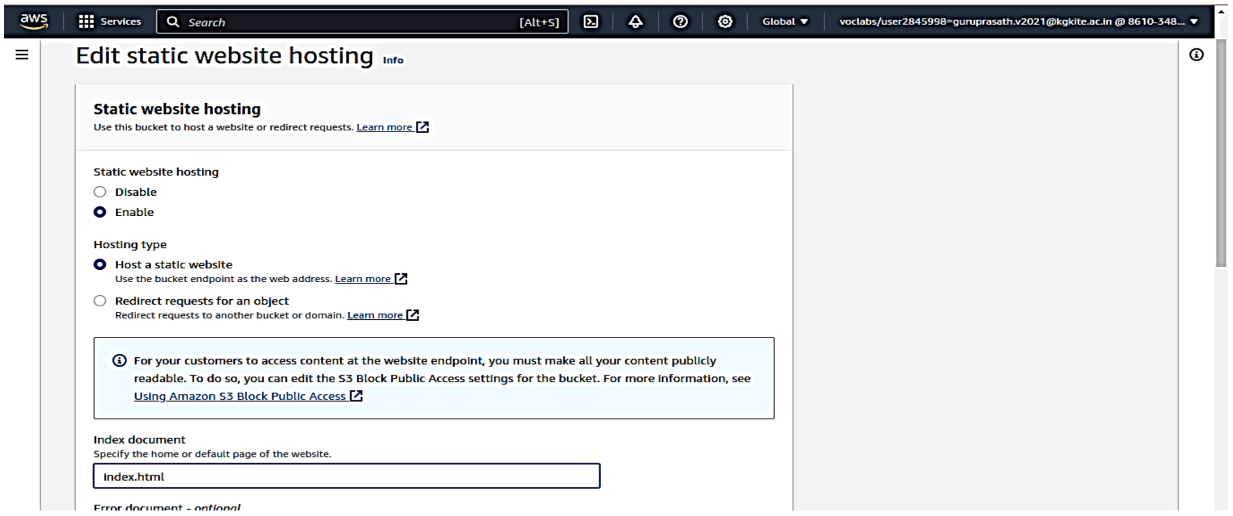
Static website hosting

Disabled

3) Click the "Edit" button.

4) Select “**Enable**” for “Use this bucket to host a website.”

5) Set the "Index document" to the name of your HTML file (e.g., "index.html").


aws Services Search [Alt+S] Global voclabs/user2845998=guruprasath.v2021@kgkite.ac.in @ 8610-348...

Edit static website hosting Info

Static website hosting

Use this bucket to host a website or redirect requests. Learn more

Static website hosting

Disable [✓ Enable]

Hosting type

[✓ Host a static website]
Use the bucket endpoint as the web address. Learn more

Redirect requests for an object
Redirect requests to another bucket or domain. Learn more

For your customers to access content at the website endpoint, you must make all your content publicly readable. To do so, you can edit the S3 Block Public Access settings for the bucket. For more information, see Using Amazon S3 Block Public Access

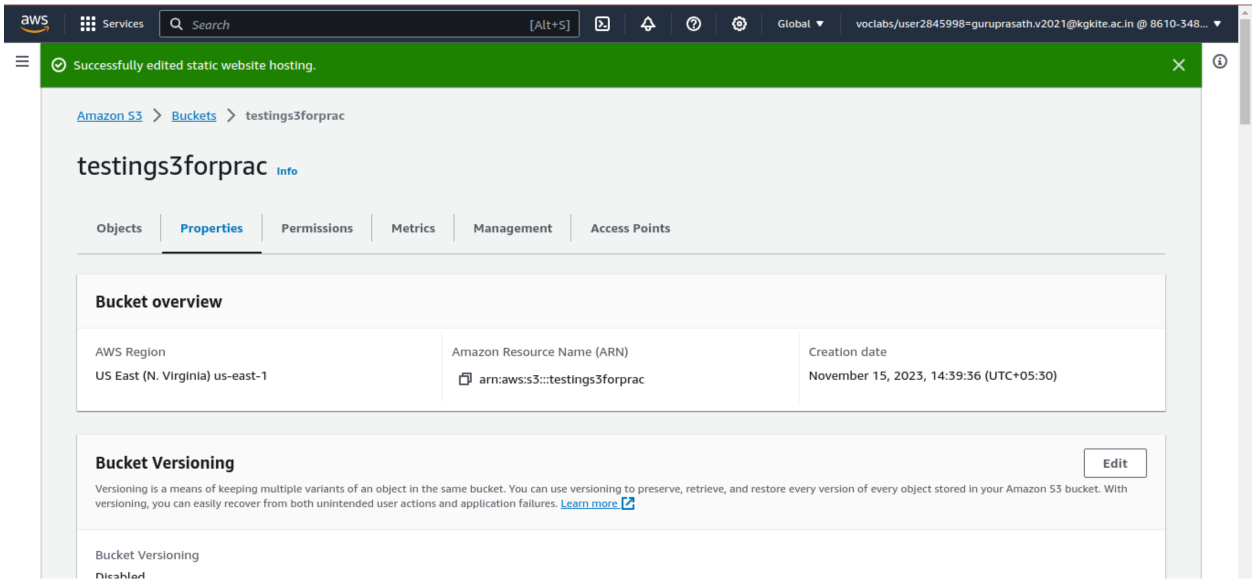
Index document
Specify the home or default page of the website.

index.html

Error document - optional

6) Click "Save."

7) You will see this alert message.



Successfully edited static website hosting.

Amazon S3 > Buckets > testings3forprac

testings3forprac Info

Objects | Properties | Permissions | Metrics | Management | Access Points

Bucket overview

AWS Region	Amazon Resource Name (ARN)	Creation date
US East (N. Virginia) us-east-1	arn:aws:s3:::testings3forprac	November 15, 2023, 14:39:36 (UTC+05:30)

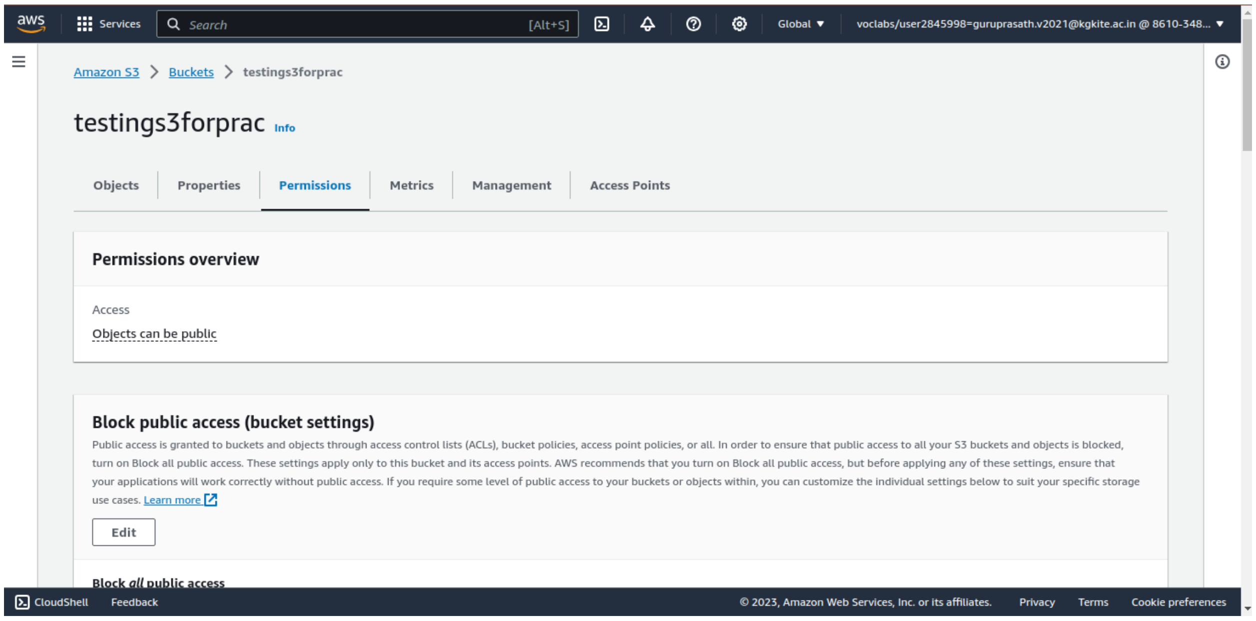
Bucket Versioning

Versioning is a means of keeping multiple variants of an object in the same bucket. You can use versioning to preserve, retrieve, and restore every version of every object stored in your Amazon S3 bucket. With versioning, you can easily recover from both unintended user actions and application failures. Learn more

Bucket Versioning
Disabled

Step 4: Access controls and permissions within S3 bucket:

1.) Go to the **Permissions** section of your Bucket (Next to properties tab).



Amazon S3 > Buckets > testings3forprac

testings3forprac Info

Objects | Properties | Permissions | Metrics | Management | Access Points

Permissions overview

Access
Objects can be public

Block public access (bucket settings)

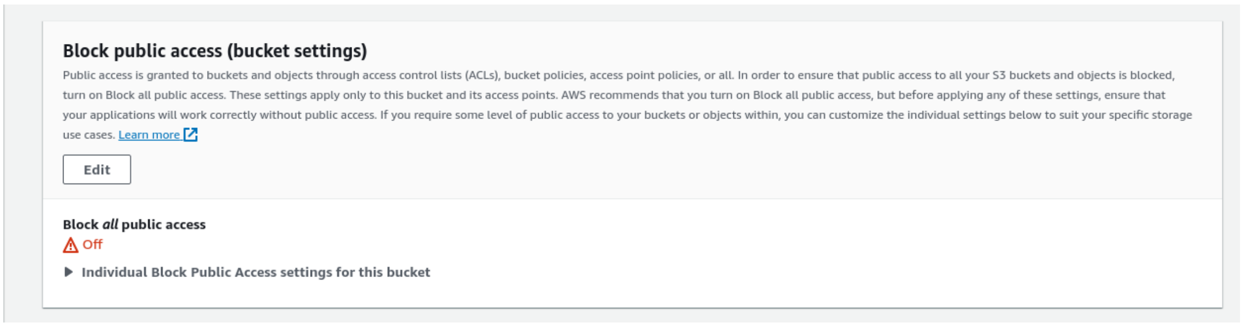
Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. Learn more

Edit

Block *all* public access

© 2023, Amazon Web Services, Inc. or its affiliates.

2.) Ensure that block public access is in off state.



Block public access (bucket settings)

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. Learn more

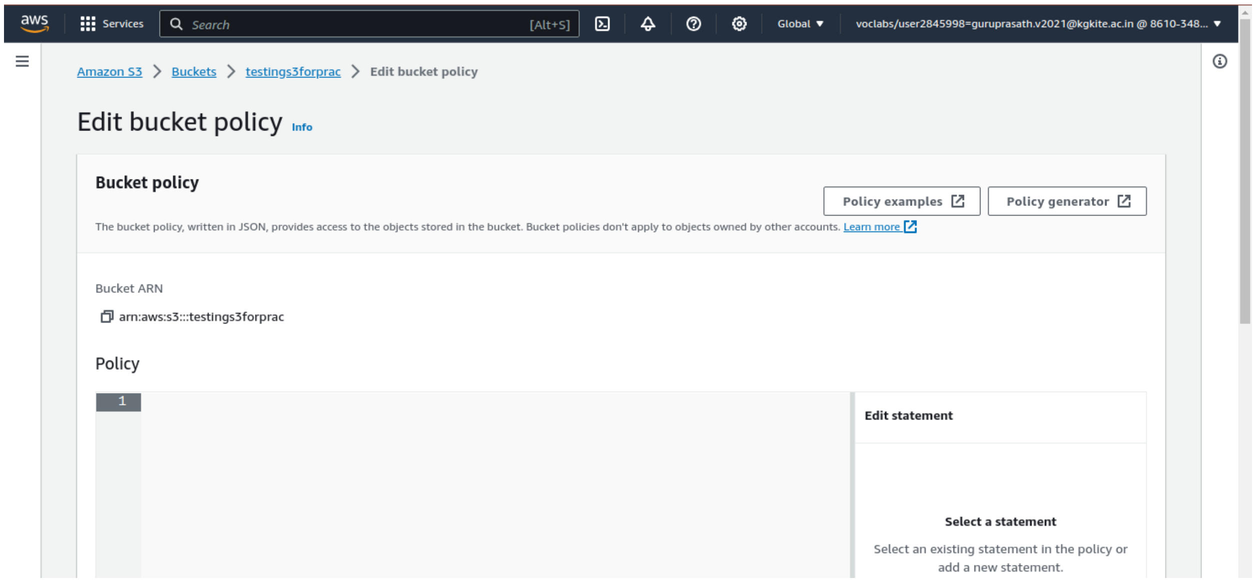
Edit

Block *all* public access
Off

Individual Block Public Access settings for this bucket

3.) Under that there will be **Bucket policy.**

4.) Click on "edit".



aws
Services
Search [Alt+S]
Global
voclabs/user2845998=guruprasath.v2021@kgkite.ac.in @ 8610-348...

Amazon S3 > Buckets > testings3forprac > Edit bucket policy

Edit bucket policy Info

Bucket policy

Policy examples
Policy generator

The bucket policy, written in JSON, provides access to the objects stored in the bucket. Bucket policies don't apply to objects owned by other accounts. Learn more

Bucket ARN

arn:aws:s3:::testings3forprac

Policy

Edit statement

Select a statement

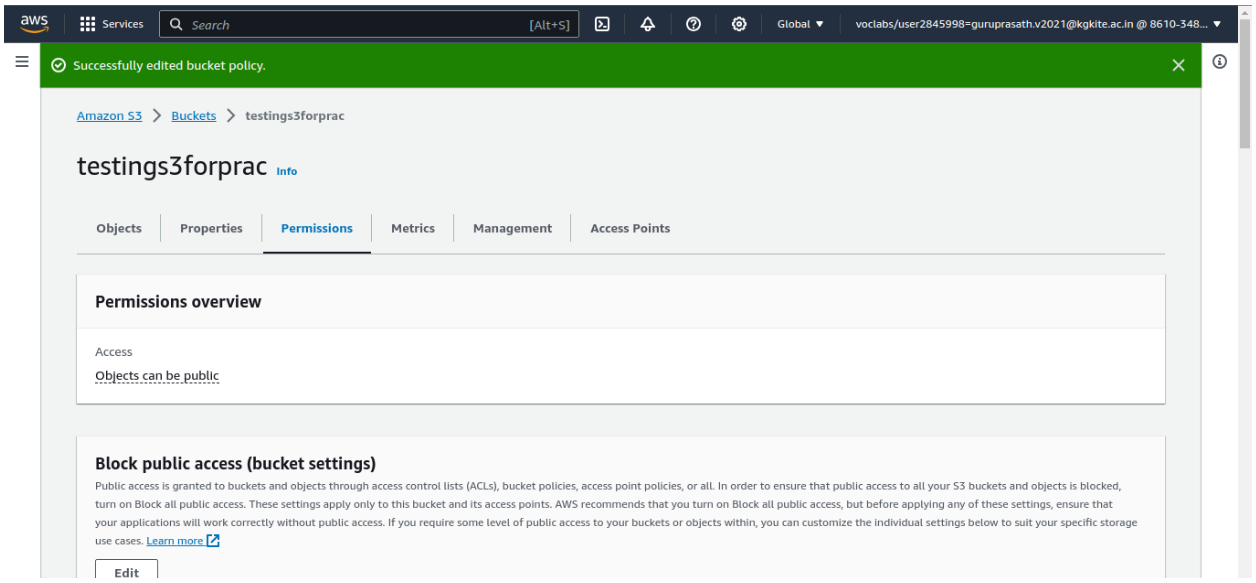
Select an existing statement in the policy or add a new statement.

5.) Add these following lines to that policy content section

```
{
   "Version": "2012-10-17",
   "Statement": [
      {
         "Sid": "PublicReadGetObject",
         "Effect": "Allow",
            "Principal": "*",
            "Action": "s3:GetObject",
            "Resource": "arn:aws:s3:::your-bucket-name/*"
        }
    ]
}
```

6.) Change the "your-bucket-name" with your actual bucket name.

7.) Click "save". you will see this alert.


aws
Services
Search
[Alt+S]
Global
voclabs/user2845998=guruprasath.v2021@kgkite.ac.in @ 8610-348...

Successfully edited bucket policy.

Amazon S3 > Buckets > testings3forprac

testings3forprac Info

Permissions overview

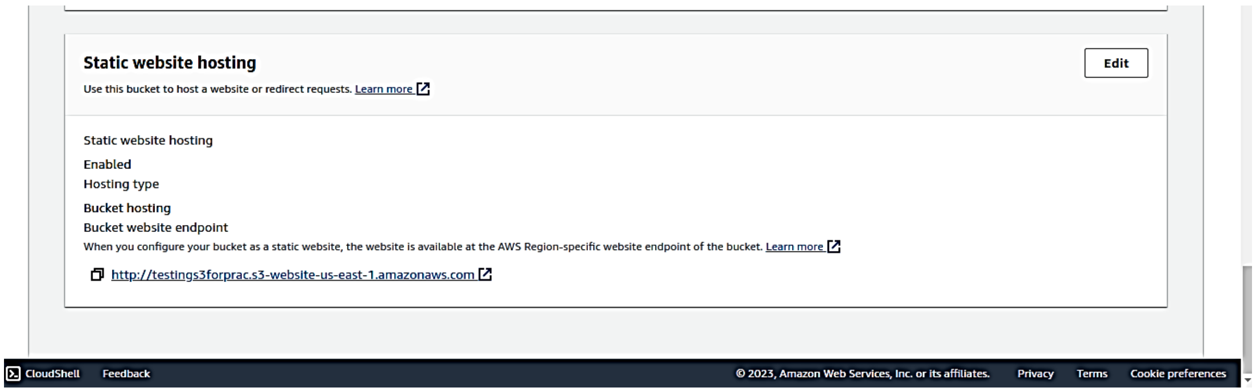
Access
Objects can be public

Block public access (bucket settings)

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. Learn more

Step 5: Verify the deployment by endpoint URL:

1.) In the properties section scroll down to reach "**Static website hosting**".



Static website hosting

Use this bucket to host a website or redirect requests. Learn more

Static website hosting

Enabled

Hosting type

Bucket hosting

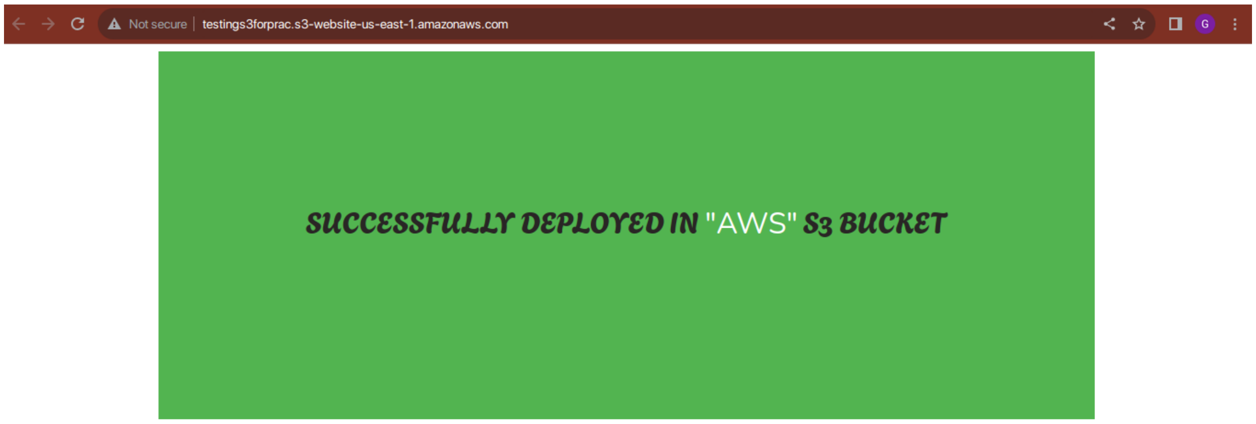
Bucket website endpoint

When you configure your bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket. Learn more

http://testings3forprac.s3-website-us-east-1.amazonaws.com

2.) There will be a link for your deployed website.

3.) Click that to verify your deployment.



Not secure | testings3forprac.s3-website-us-east-1.amazonaws.com
SUCCESSFULLY DEPLOYED IN "AWS" S3 BUCKET

That's it you've deployed a website in cloud (S3).

RESULT:

Create a Web Application using S3 has been executed successfully and the output is verified.