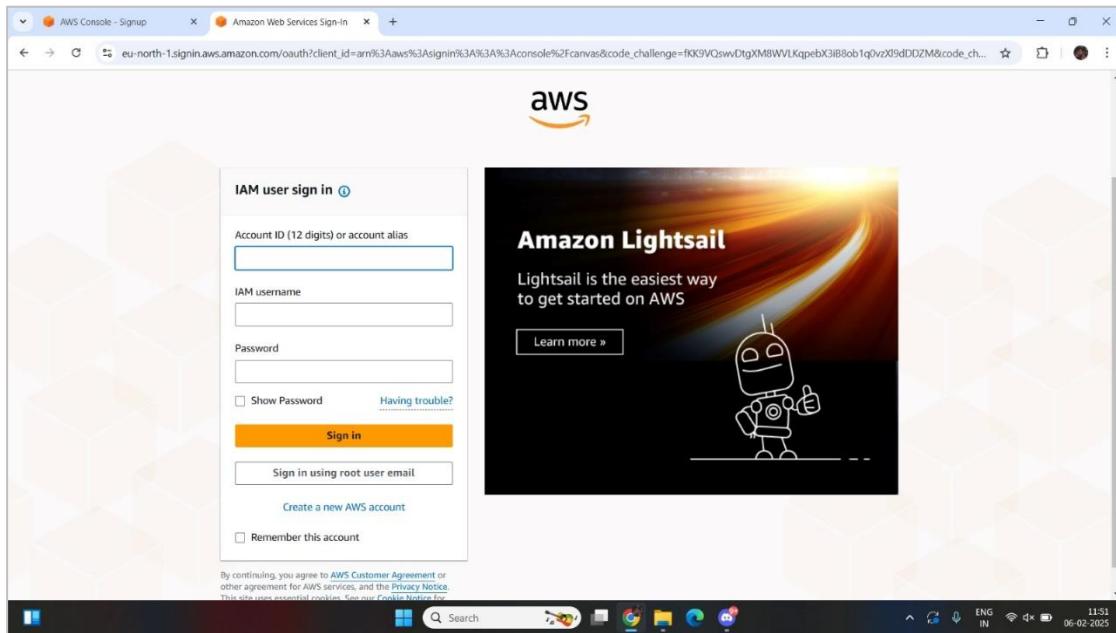


ASSIGNMENT 1:

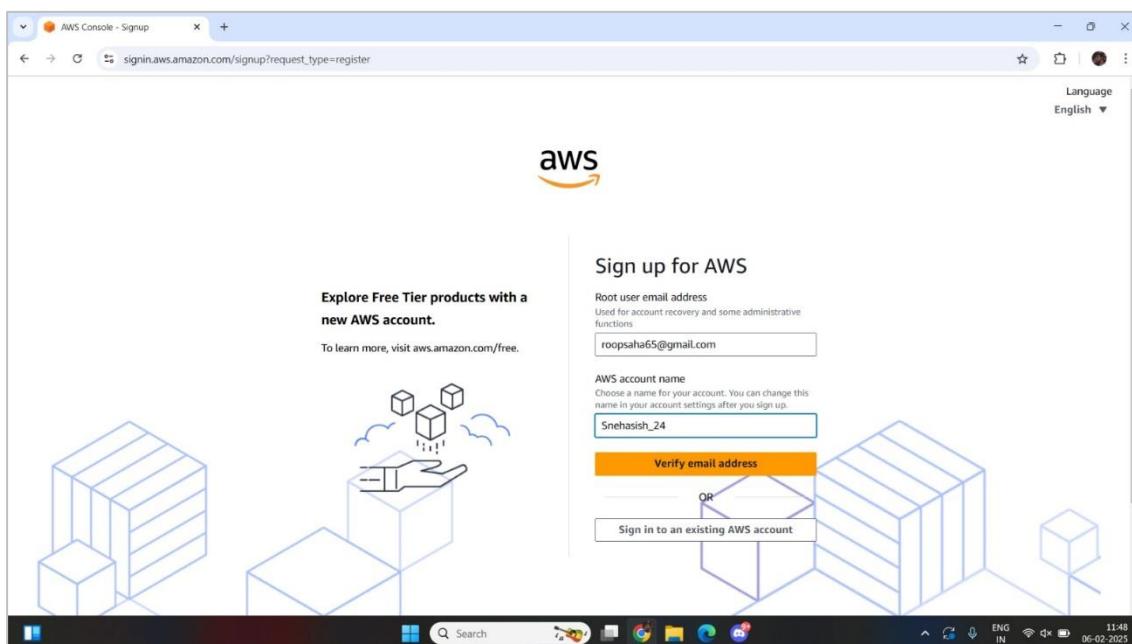
Problem Definition: Create an account in AWS and configure a budget

Account Creation:

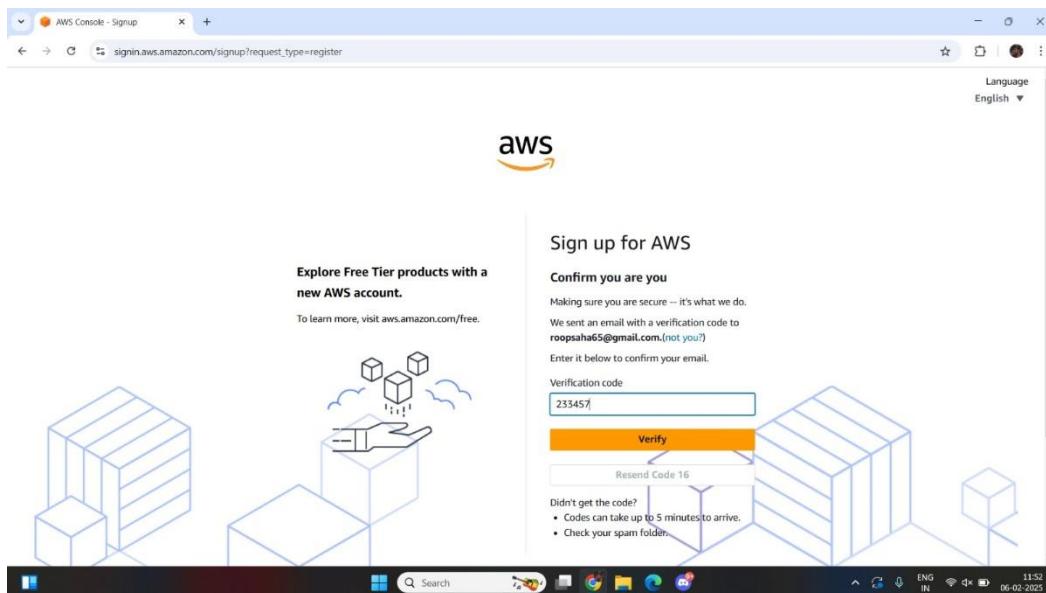
1. Go to **AWS Management Console** and click on “**Sign Up**”.



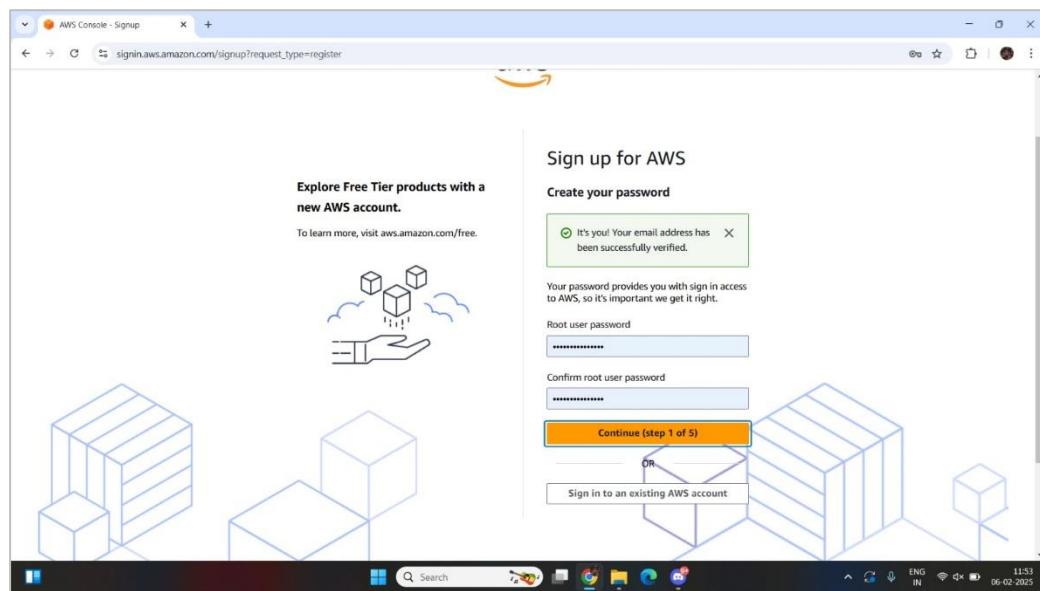
2. Enter root user email address and AWS account name.



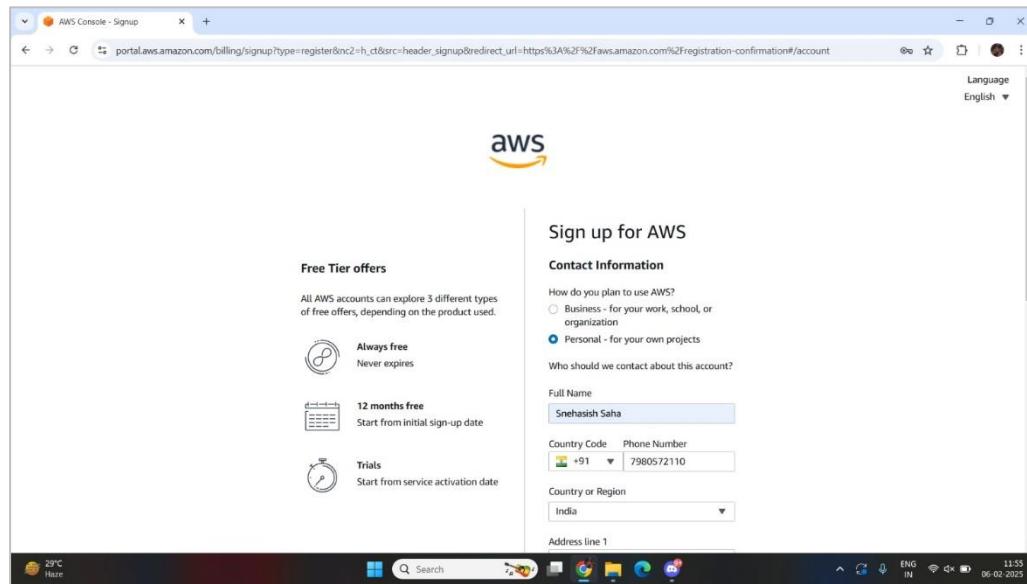
3. Verify your email address.



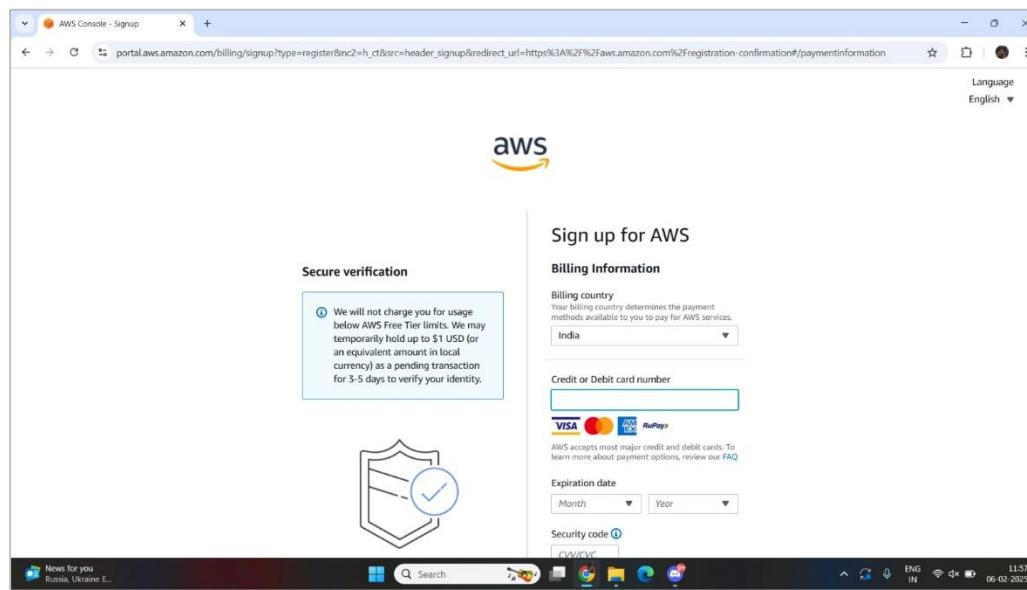
4. Set your password, then click on "**Continue (step 1 of 5)**".



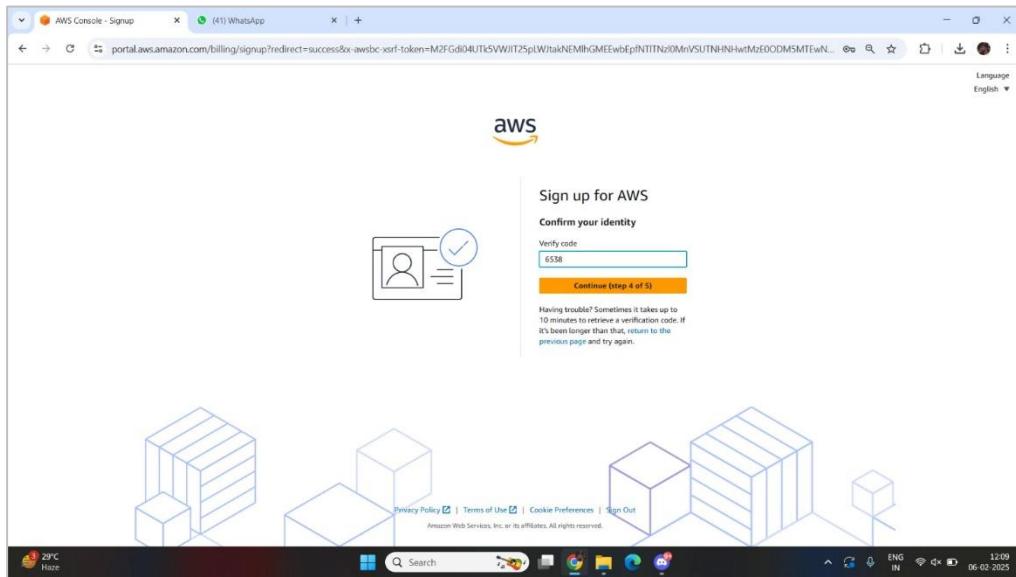
5. Fill the required details, tick off the check box and then click on “**Continue (step 2 of 5)**”.



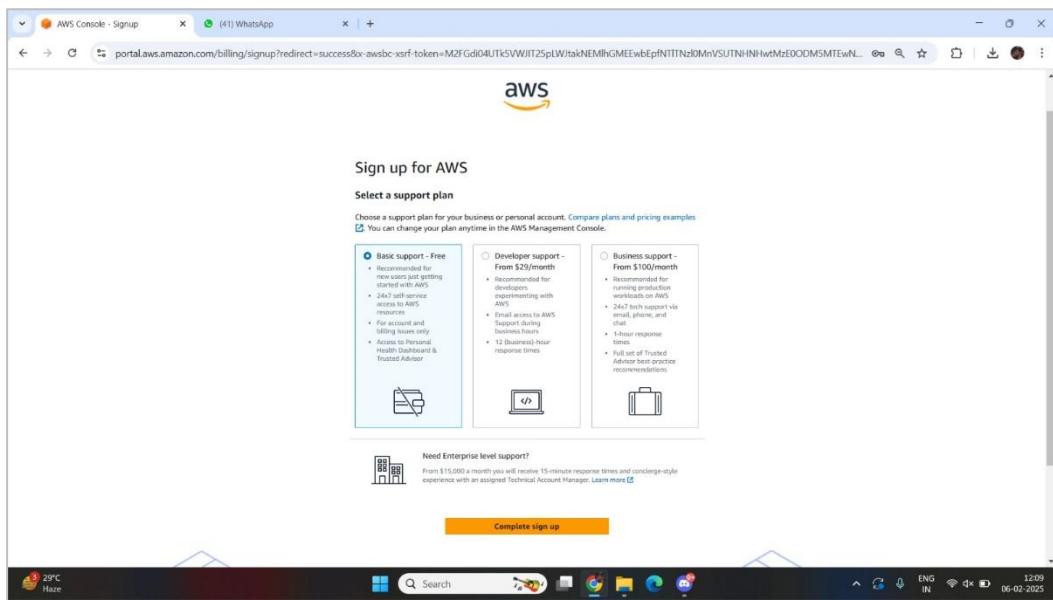
6. Enter billing information and complete payment, then click on “**Continue (step 3 of 5)**”.



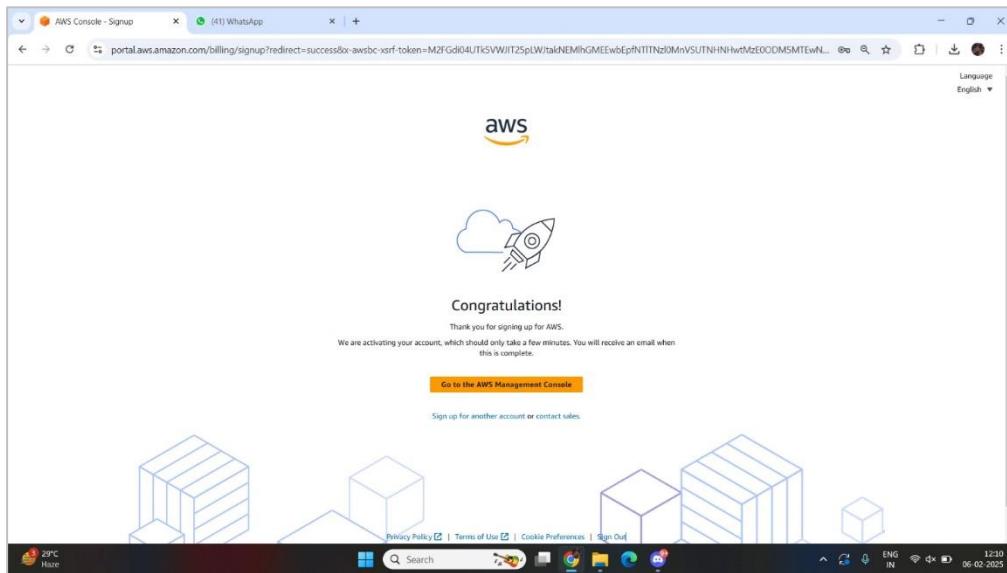
7. Verify your phone number then click “**Continue (step 4 of 5)**”.



8. Select a Support Plan then click on “**Complete sign up**”.

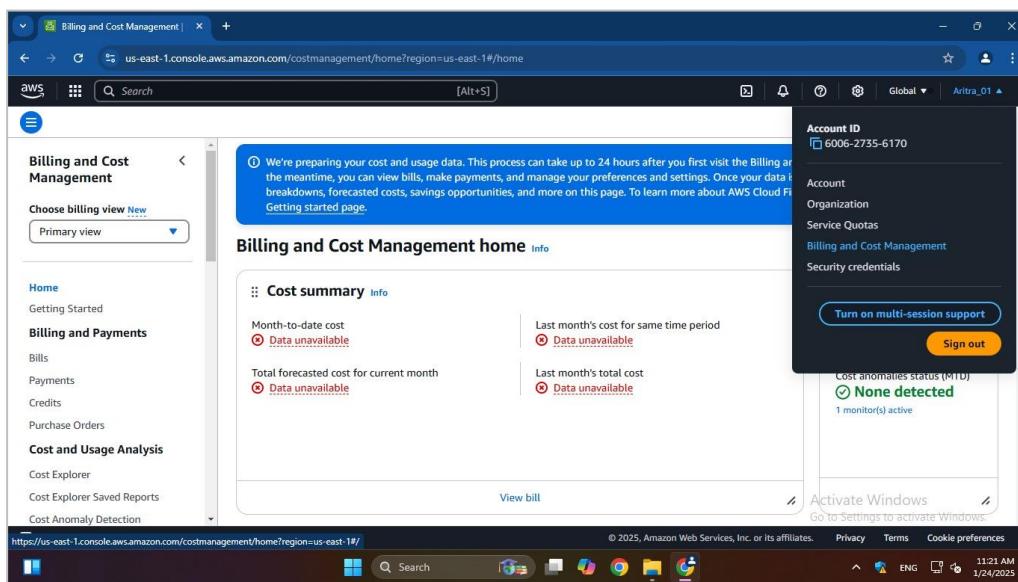


9. Congratulation page will be shown if account is created successfully.



Budget Configuration:

1. Login to **AWS Management Console**, click on your username at the top right corner and click on "**Billing and Cost Management**".

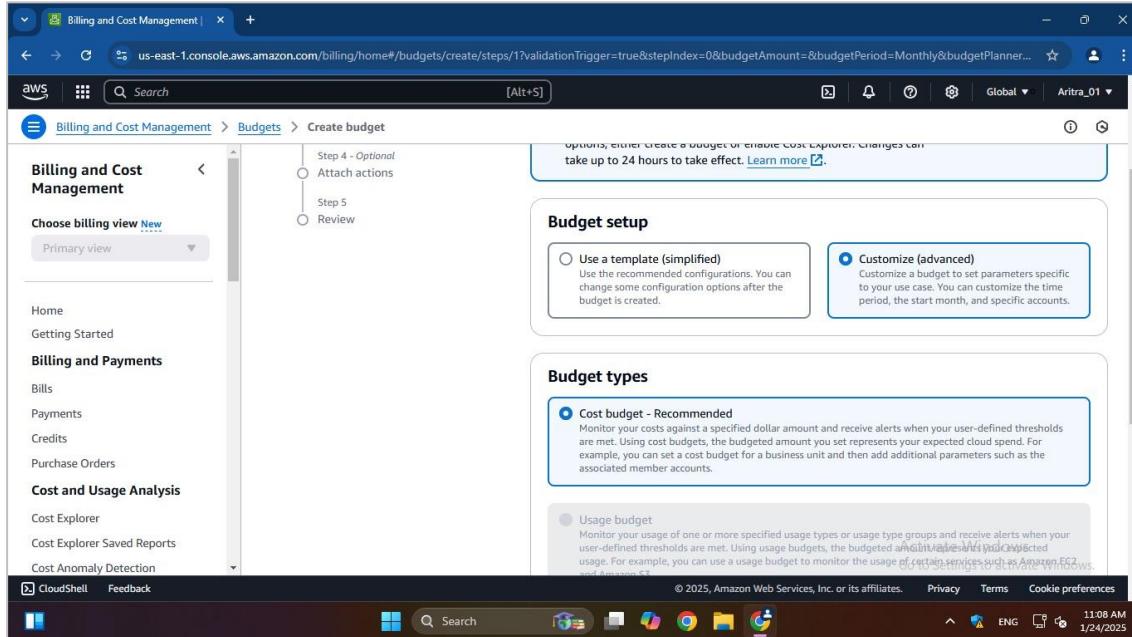


2. Under “**Budgets and Planning**”, click on “**Budgets**” and then click on “**Click on budget**”.

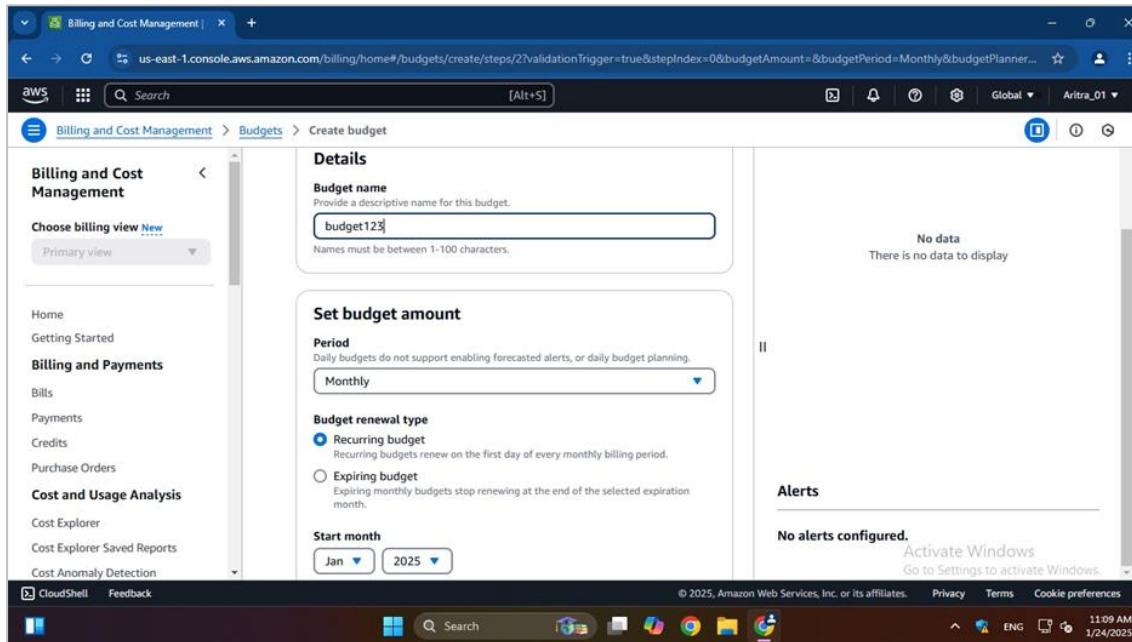
The screenshot shows the AWS Billing and Cost Management home page. On the left, there's a sidebar with various navigation options like Customer Carbon Footprint Tool, Cost Organization, Cost Categories, Cost Allocation Tags, Billing Conductor, Budgets and Planning, Savings and Commitments, and more. The main content area has a blue banner at the top stating: "We're preparing your cost and usage data. This process can take up to 24 hours after you first visit the Billing and Cost Management console. In the meantime, you can view bills, make payments, and manage your preferences and settings. Once your data is ready, you can view cost breakdowns, forecasted costs, savings opportunities, and more on this page. To learn more about AWS Cloud Financial Management, visit the Getting started page." Below the banner, there are two main sections: "Cost summary" and "Cost monitor". The "Cost summary" section displays month-to-date cost, total forecasted cost for current month, and last month's total cost, all marked as "Data unavailable". The "Cost monitor" section shows budgets status, setup required (marked as "None detected"), and cost anomalies status (MTD). A "View bill" button is located at the bottom of the main content area.

The screenshot shows the AWS Billing home page. On the left, there's a sidebar with options like Home, Getting Started, Billing and Payments, Bills, Payments, Credits, Purchase Orders, Cost and Usage Analysis, Cost Explorer, Cost Explorer Saved Reports, and Cost Anomaly Detection. The main content area features a large section titled "AWS Budgets" with the sub-headline "Set custom budgets that alert you when you exceed your budgeted thresholds". It includes a callout box for "Start tracking your AWS costs and usage" with a "Create a budget" button. Another callout box for "Pricing (US)" states: "There is no additional charge for using AWS Budgets. You only pay for configured actions that go beyond the free tier offer of 62 action-enabled budget days." A "How it works" diagram is shown below the main text.

3. Configure budget setup by selecting “**Customize (advanced)**” then click on “**Next**”.



4. Add budget details.



5. After selecting “**Add threshold**”, fill-in the necessary details and then click on “**Next**”.

Billing and Cost Management > Budgets > Edit budget

Set alert threshold

Threshold: When should this alert be triggered?
80 % of budgeted a... Trigger: How should this alert be triggered?
Actual

Summary: When your actual cost is greater than 80.00% (\$0.80) of your budgeted amount (\$1.00), the alert threshold will be exceeded.

Notification preferences: Select one or more notification preferences to receive alerts.

Email recipients: Specify the email recipients you want to notify when the threshold has exceeded.
akOley09@gmail.com

Cost Data: No data. There is no data to display.

6. Review the budget details and click on “**Save**” to finalize the budget.

Billing and Cost Management > Budgets > Edit budget

Review Info

Step 1: Set up your budget

Budget details

Name budget123	Start date Jan 2025	Budget amount \$1.00
Period Monthly	End date -	

Additional budget parameters

Tags

Key	Value
There are no tags associated with this resource.	

7. Budget is successfully created.

The screenshot shows the AWS Billing and Cost Management console. The left sidebar includes sections for Home, Getting Started, Billing and Payments (Bills, Payments, Credits, Purchase Orders), and Cost and Usage Analysis (Cost Explorer, Cost Explorer Saved Reports, Cost Anomaly Detection). The main content area displays a green success message: "Your budget budget123 has been updated successfully." Below this, a table lists one budget entry:

Name	Thresholds	Budget	Amount used	Forecasted ...	Current vs. bu...
budget123	OK	\$1.00	\$0.00		

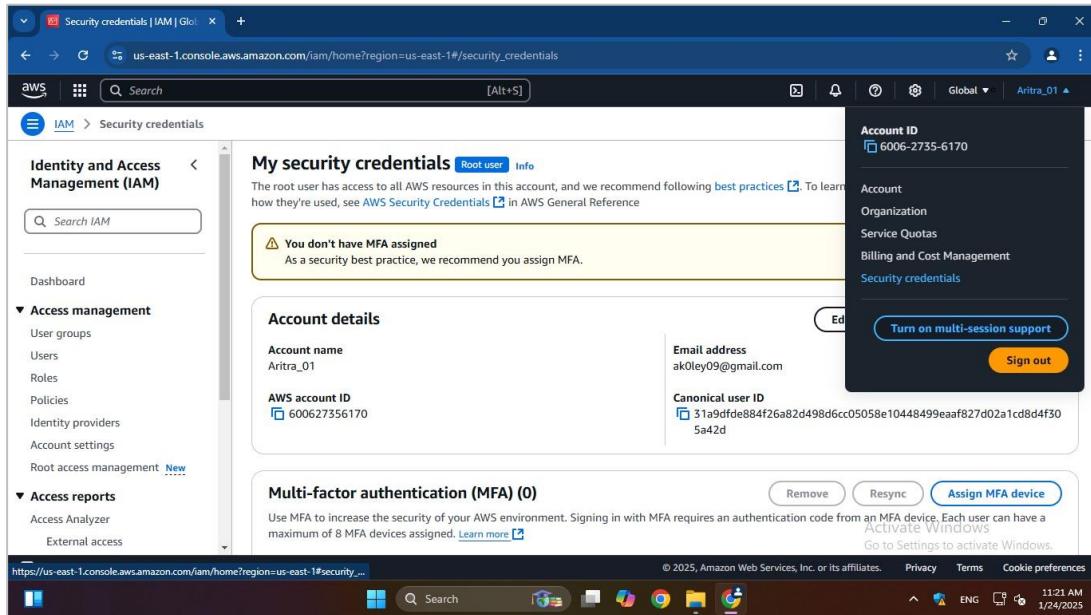
At the bottom right of the page, there is a watermark: "Activate Windows Go to Settings to activate Windows". The footer contains links for Privacy, Terms, and Cookie preferences, along with system status indicators for CloudShell, Feedback, and a timestamp of 11:17 AM on 1/24/2025.

ASSIGNMENT 2:

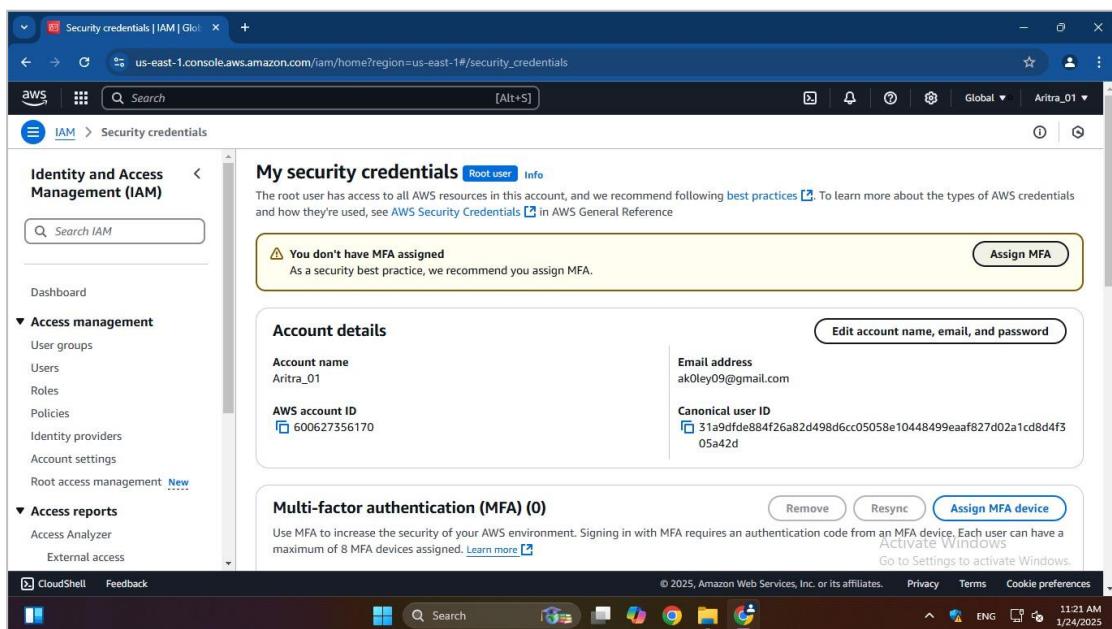
Problem Definition: Create MFA (Multi-Factor Authentication) for authentication.

Instructions:

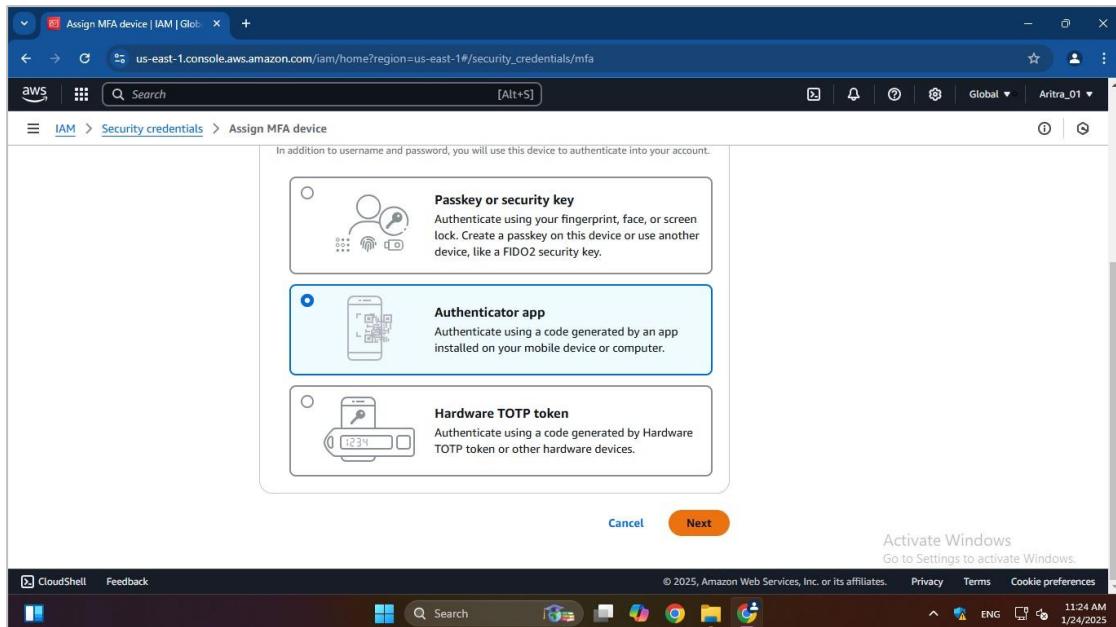
1. Log into AWS account and go to “**Security credentials**”.



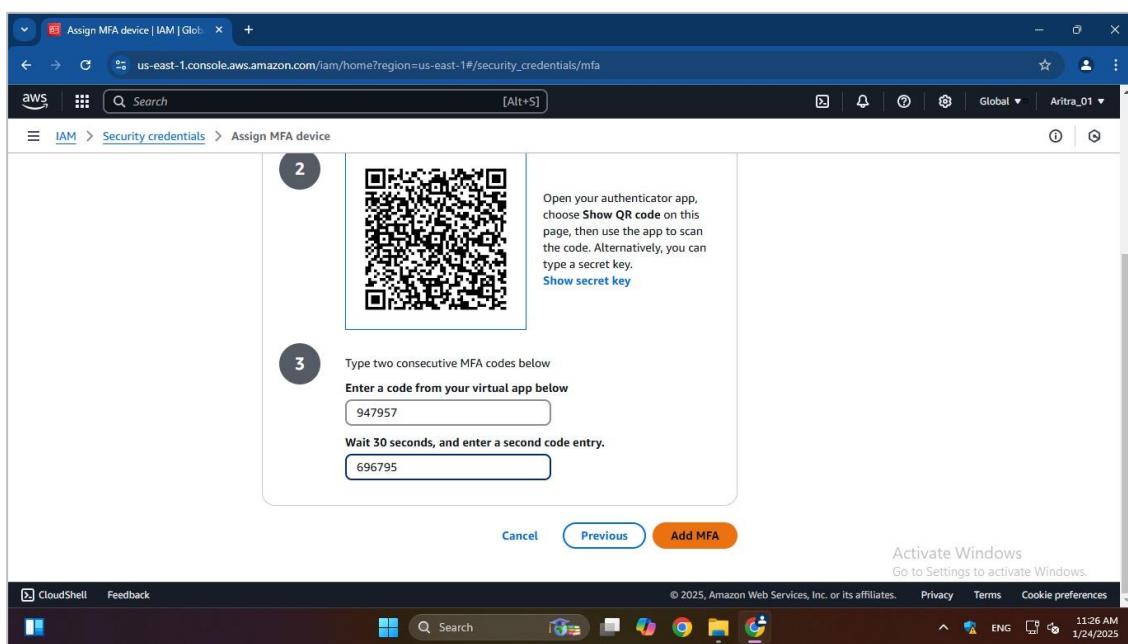
2. Add MFA device by clicking on “**Assign MFA device**”.



3. Enter a desired device name, MFA device and then click on “**Next**”.



4. Enter code using authenticator app and click on “**Add MFA**” to complete the process.



5. MFA device assigned successfully.

The screenshot shows the AWS IAM Security Credentials page. A green success message at the top states: "MFA device assigned. You can register up to 8 MFA devices of any combination of the currently supported MFA types with your AWS account root and IAM user. With multiple MFA devices, you only need one MFA device to sign in to the AWS console or create a session through the AWS CLI with that user." Below this, the "Multi-factor authentication (MFA) (1)" section displays a single entry:

Type	Identifier	Certifications	Created on
Virtual	arn:aws:iam::600627356170:mfa/Aritra@123	Not Applicable	Fri Jan 24 2025

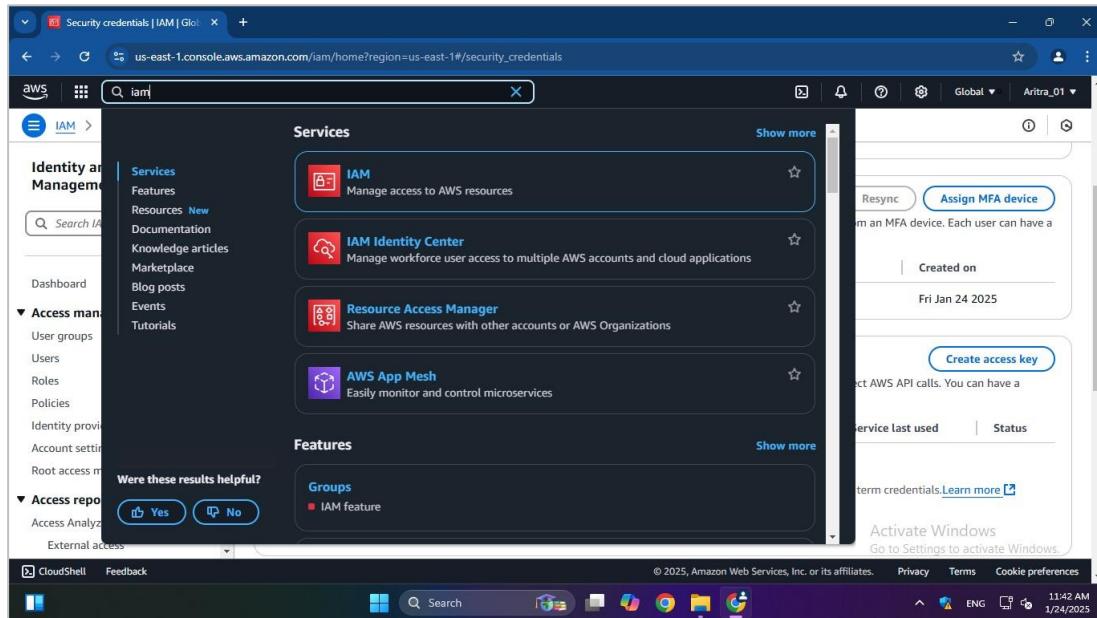
Below this, the "Access keys (0)" section indicates "No access keys". The page includes a sidebar with navigation links for Identity and Access Management (IAM), Access management, and Access reports. The status bar at the bottom shows the date and time as 1/24/2025 and 11:27 AM.

ASSIGNMENT 3:

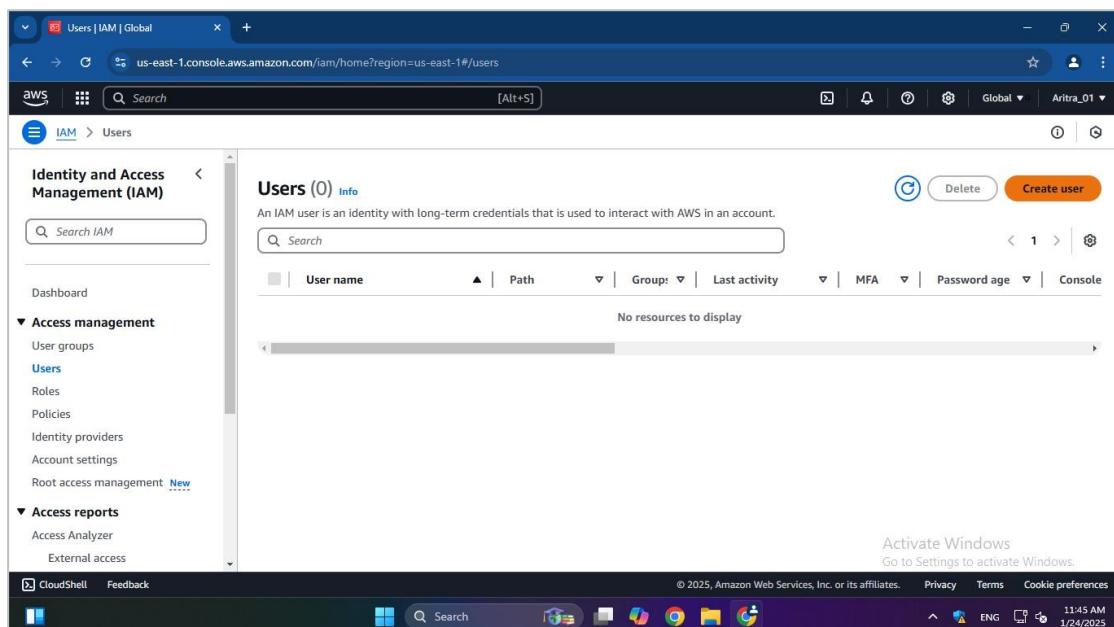
Problem Definition: Create IAM user and give full access to S3.

Instructions:

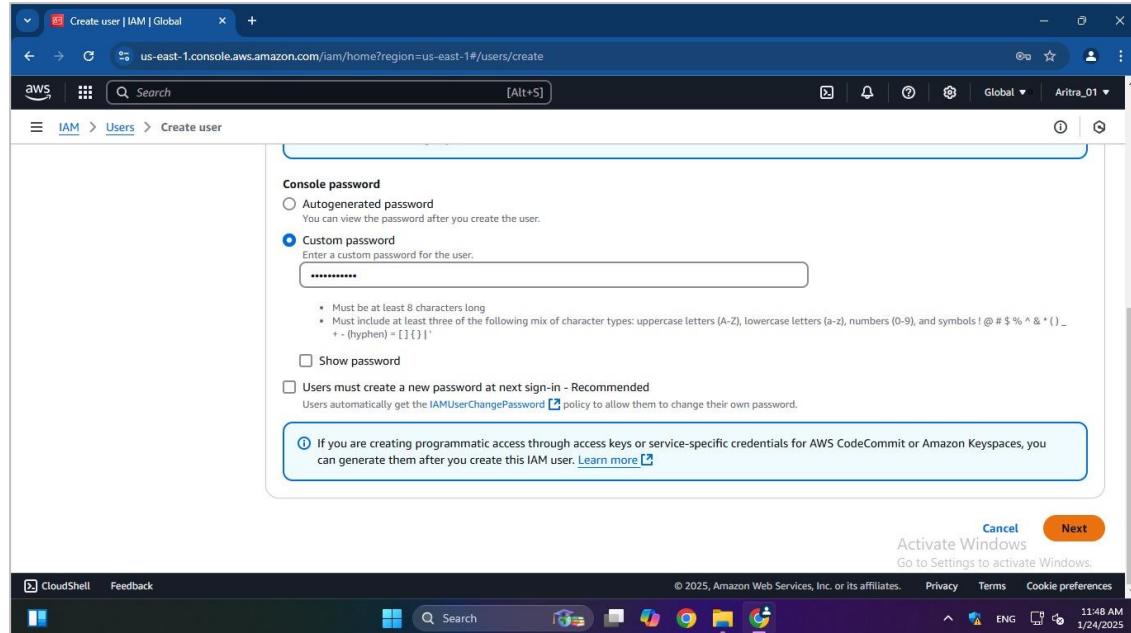
1. From AWS console, search for “**IAM**” and click on it.



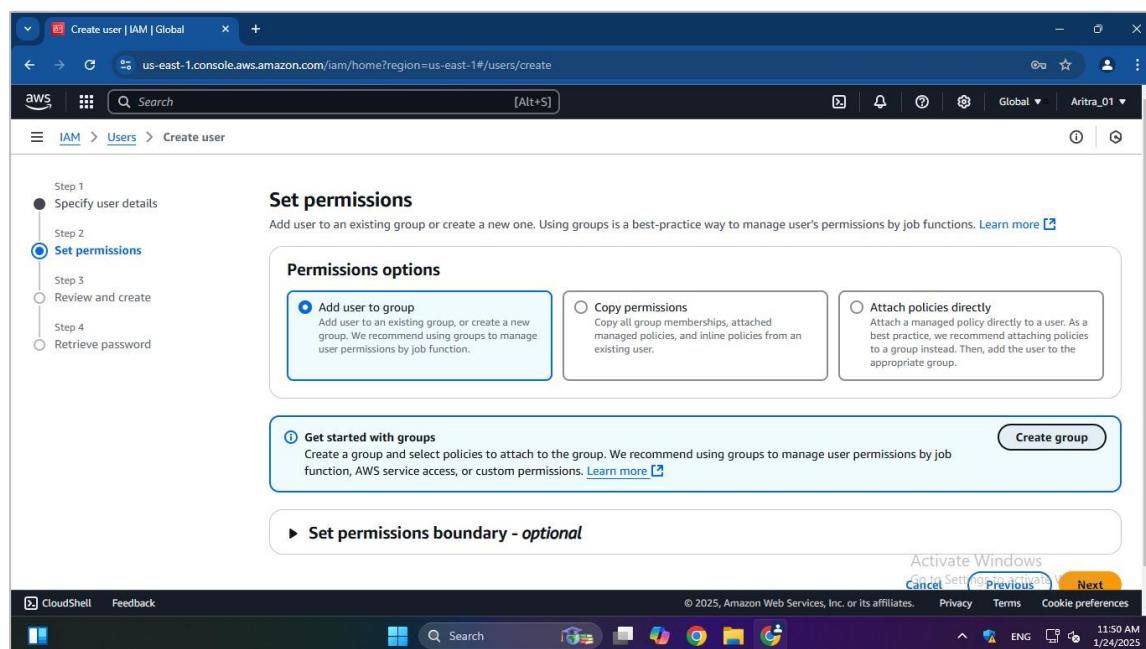
2. Under “**IAM resources**” click on “**Users**”.



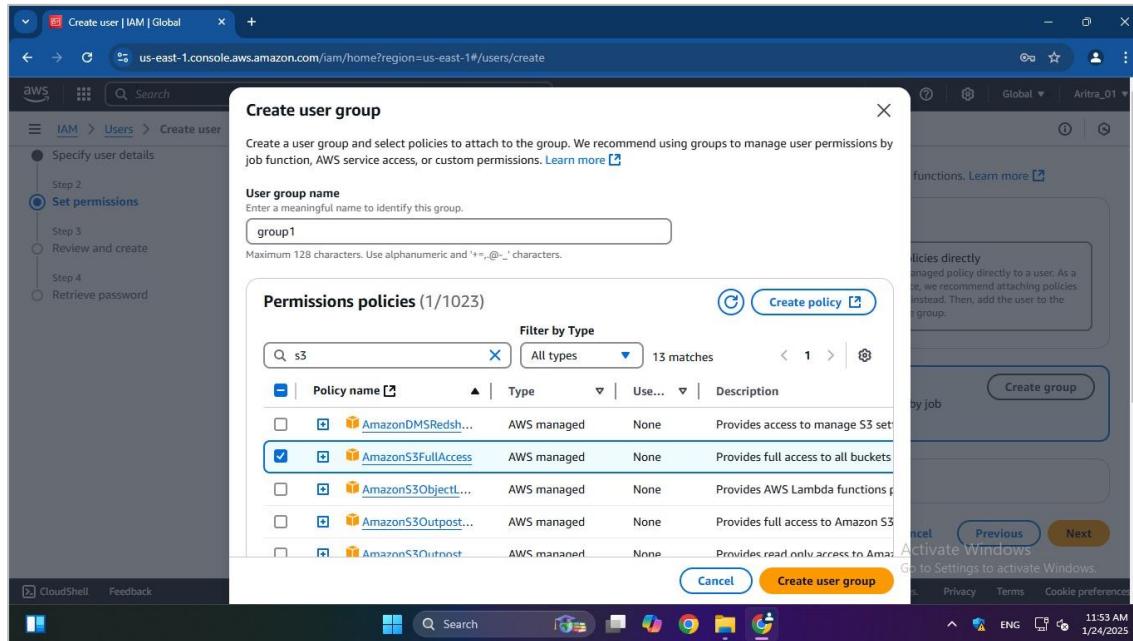
3. Click on “**Create User**”. Provide User details (name, password, etc.) and click on “**Next**” to proceed.



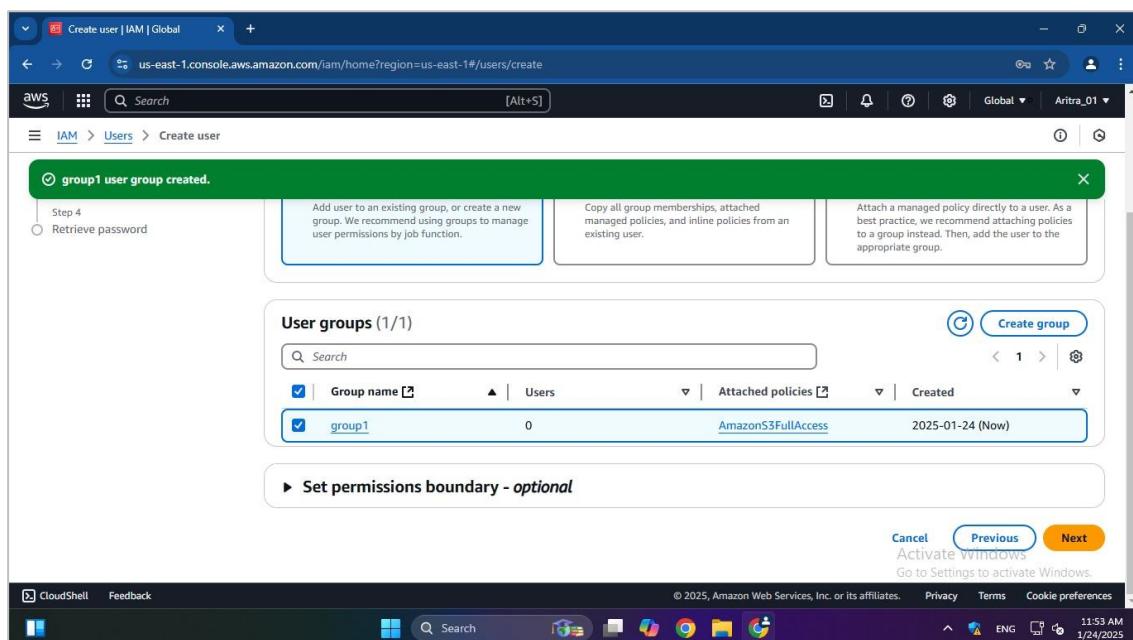
4. Under “**Permissions options**”, click on “**Add user to group**” and go to “**Create group**”.



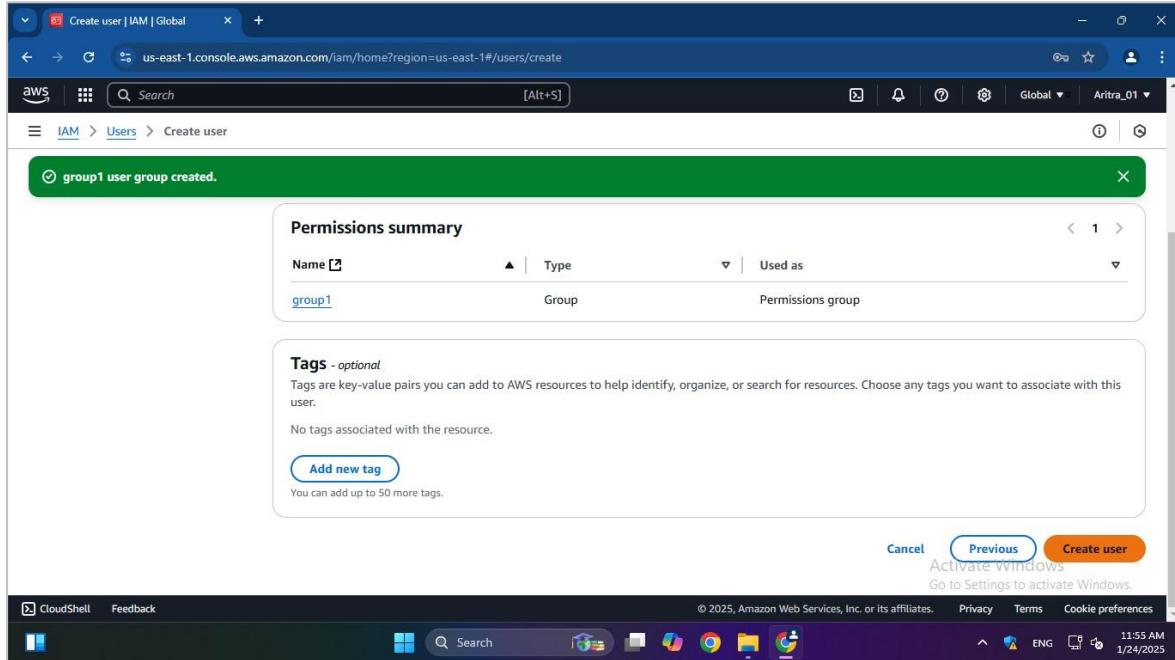
5. To create user group, first provide “**User group name**”, then from “**Permission policies**” search for “**S3**” and select full access of the particular permission we want to give to that group. Then click on “**Create User Group**”.



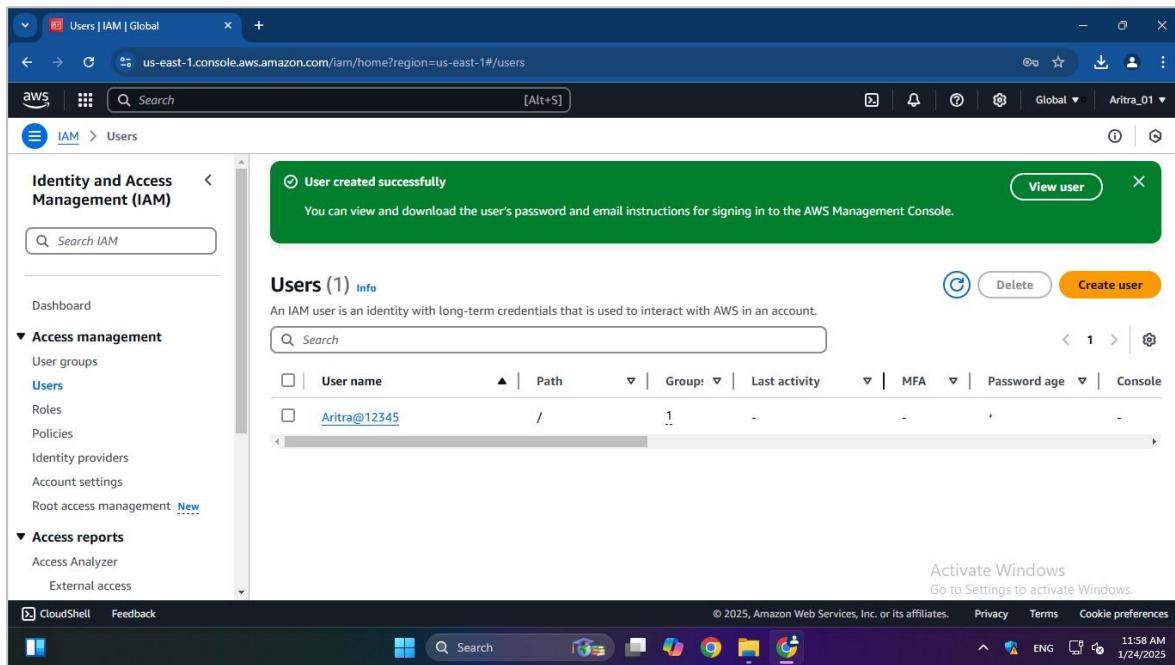
6. The group is successfully created. Tick the checkbox of the created group in “**User groups**”, so that our IAM user is connected to this group and gets the required permission. Then click on “**Next**”.



7.Under “**Review and create**”, click on “**Create user**” to create our IAM user.



8. IAM user is created successfully.

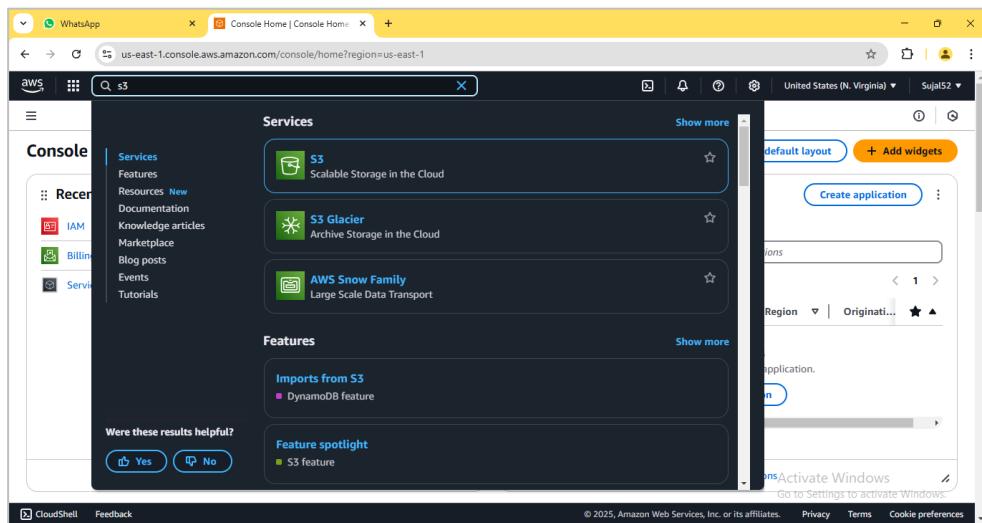


ASSIGNMENT 4:

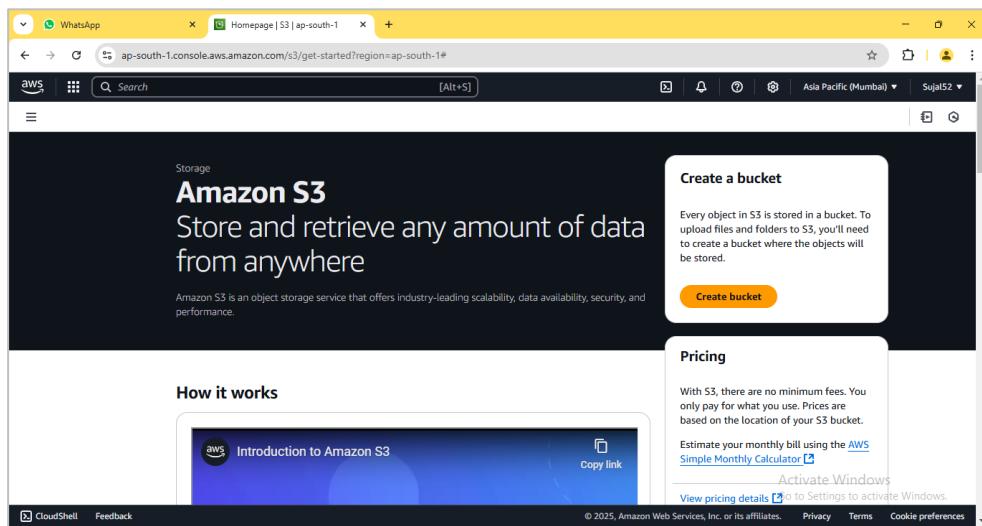
Problem Definition: Create a private bucket in AWS. Upload a file and check by reassigned URL whether you can access the file or not

Instructions:

1. Access the **AWS console**, search for **S3**, and select the top option from the search results.



2. Click on "**Create bucket**".



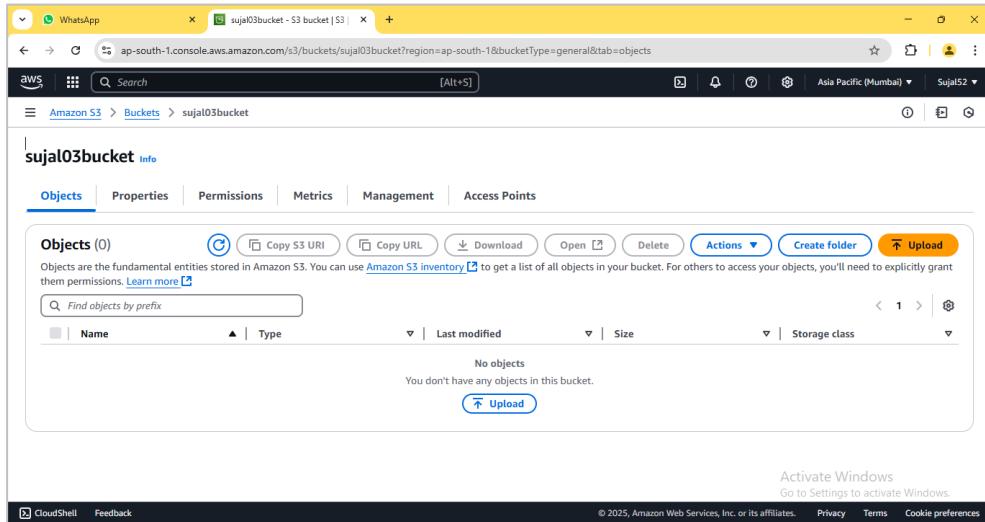
3. Enter bucket name. Leave the other options set to default as we are creating a private bucket. Then click on “**Create bucket**”.

The screenshots show the 'Create S3 bucket' wizard. In the first step, 'General configuration', the 'Bucket type' is set to 'General purpose'. In the second step, 'Encryption type', the 'Encryption type' is set to 'Server-side encryption with Amazon S3 managed keys (SSE-S3)' and 'Bucket Key' is enabled. Both steps include a note about activating Windows.

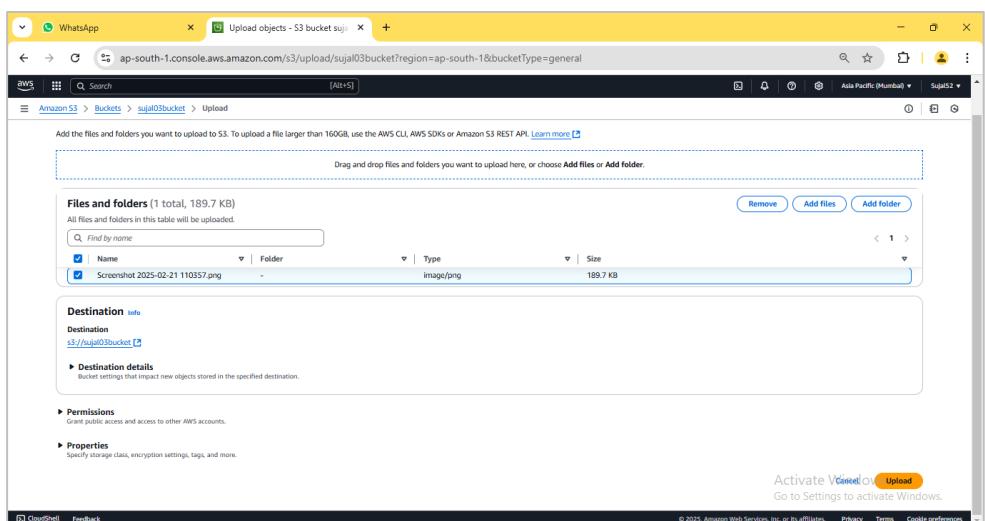
4. A green coloured pop-up dialogue box shall confirm the creation of our bucket. It shall also be displayed on the bucket list.

The screenshot shows the 'Amazon S3 > Buckets' page. A green success message at the top states 'Successfully created bucket "sujal03bucket"'. Below it, there is an 'Account snapshot - updated every 24 hours' section. The main list shows one 'General purpose buckets' entry named 'sujal03bucket', which was created on 'February 21, 2025, 10:59:18 (UTC+05:30)'. The 'Create bucket' button is visible at the top right of the list.

5. Click on the name of our newly created bucket. On the bucket detail page, click on “**Upload**”.



6. Click on “**Add files**” then select the file we want to upload to our bucket. Select our file from the list and click on “**Upload**” button.



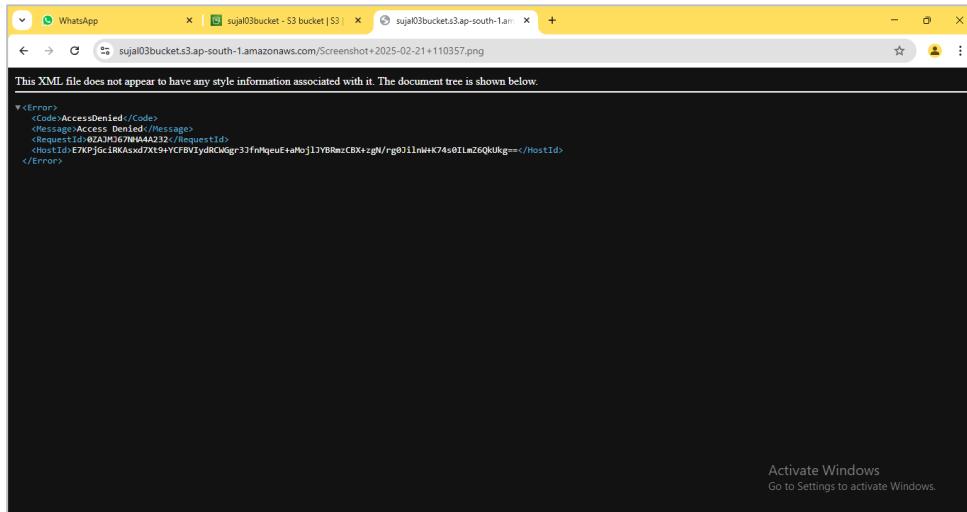
7. The notification confirms that the file has been successfully uploaded. Click on “**Close**” to return to our bucket details page.

The screenshot shows the AWS S3 console interface. At the top, a green success message box displays "Upload succeeded" and "For more information, see the Files and folders table." Below this, the "Summary" section shows the destination as "s3://sujal03bucket" and lists one succeeded file: "Screenshot 2025-02-21 110357.png" (189.7 KB). The "Files and folders" tab is selected, showing a table with one row of data. The table columns are Name, Folder, Type, Size, and Status. The status for the file is "Succeeded". At the bottom of the page, there are links for CloudShell, Feedback, and cookie preferences.

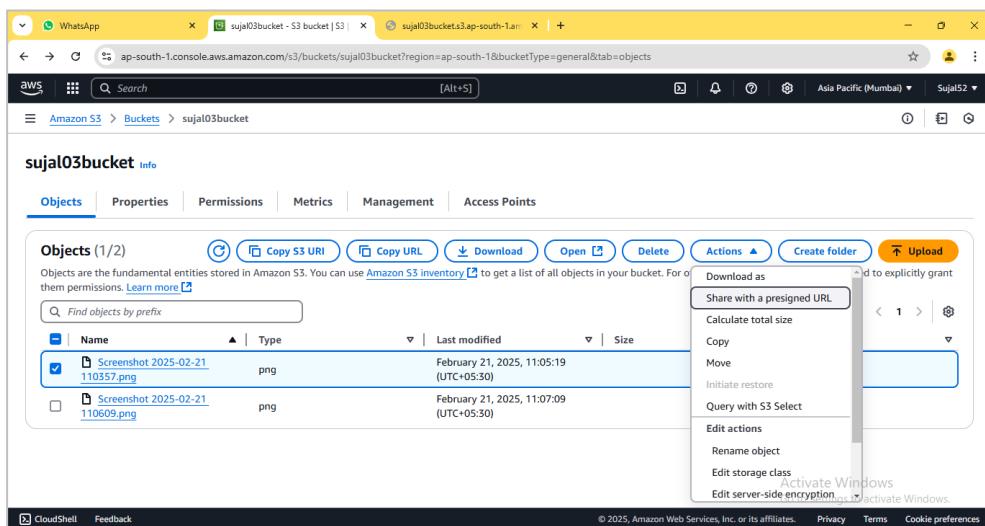
8. Click on our previously uploaded file from the list. Copy the “**Object URL**” from the file details page that appears.

The screenshot shows the AWS S3 console interface. The top navigation bar shows the path "Amazon S3 > Buckets > sujal03bucket". The main content area displays the "Objects" list for the "sujal03bucket". There is one object listed: "Screenshot 2025-02-21 110357.png" (png type, 189.7 KB, last modified February 21, 2025, 11:05:19 UTC+05:30). Below this, the "Screenshot 2025-02-21 110357.png" file details page is shown. The "Properties" tab is selected. On the right side, the "Object URI" is displayed as "https://s3.ap-south-1.amazonaws.com/sujal03bucket/Screenshot+2025-02-21+110357.png". Other tabs include "Permissions" and "Versions". The bottom of the page includes links for CloudShell, Feedback, and cookie preferences.

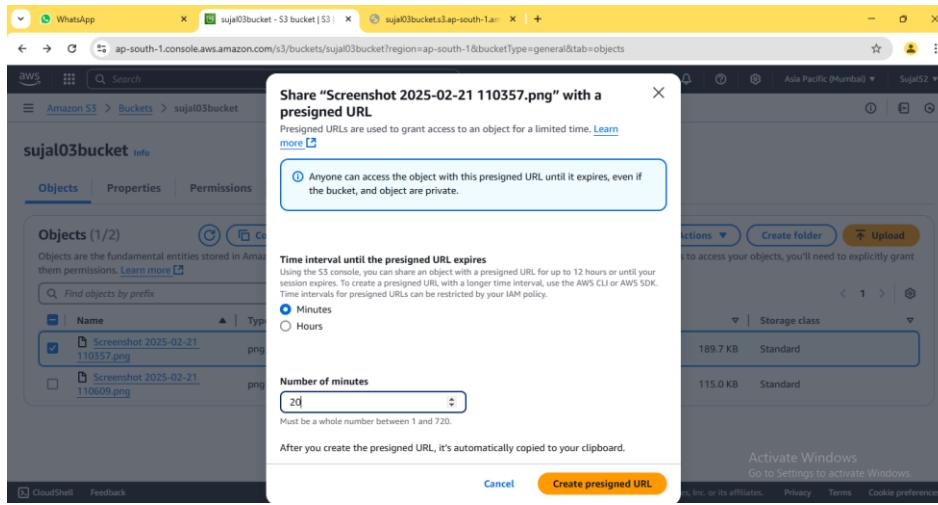
9. Open the copied URL in a new browser window. An error appears indicating that access to the contents is restricted since this is a private bucket.



10. Go back to bucket details page, select our file from the list, click on “**Actions**” and select “**Share with a presigned URL**” from the dropdown.



11. Select the desired duration until you want the presigned URL to expire then click on “**Create presigned URL**”.



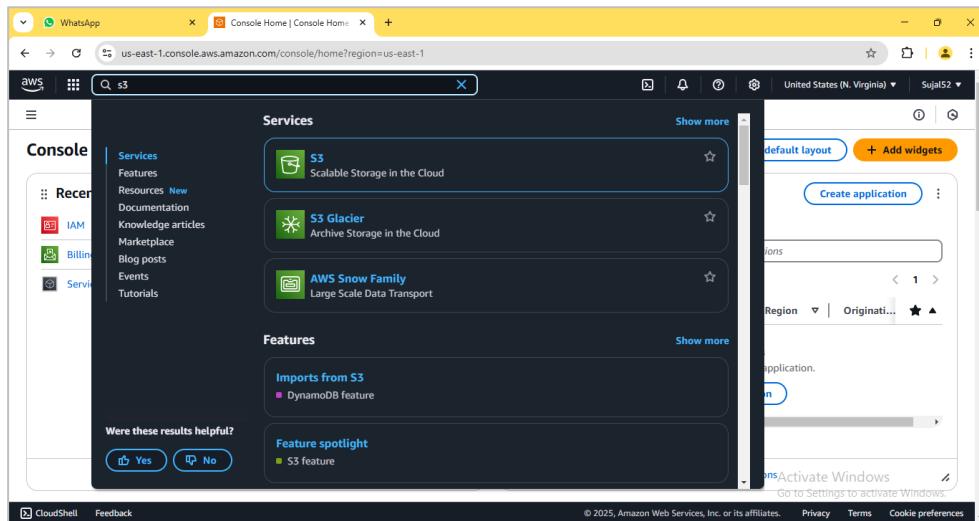
12. Copy the pre-assigned URL and paste it on a new window. Now, you can view the uploaded file.

ASSIGNMENT 5:

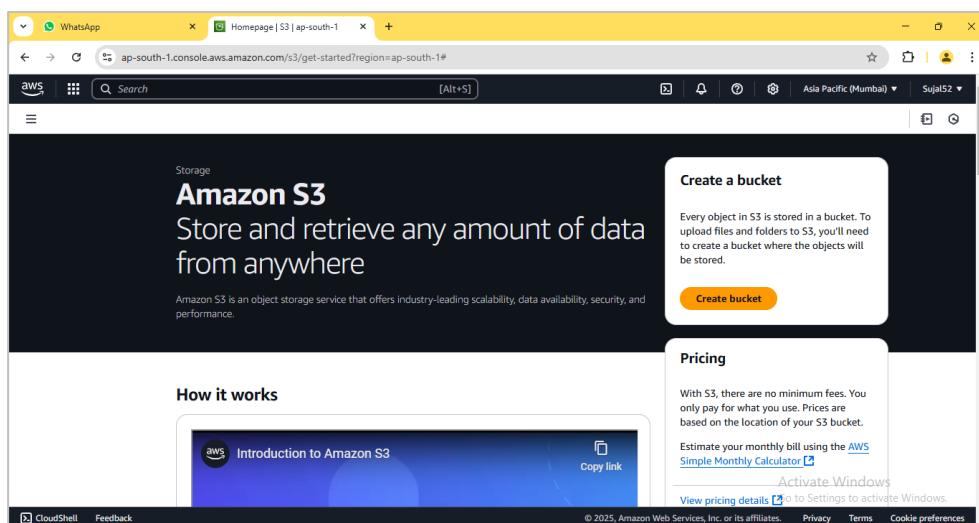
Problem Definition: Create a public Bucket in AWS. Upload a file and give the necessary permission to check whether file the URL is working.

Instructions:

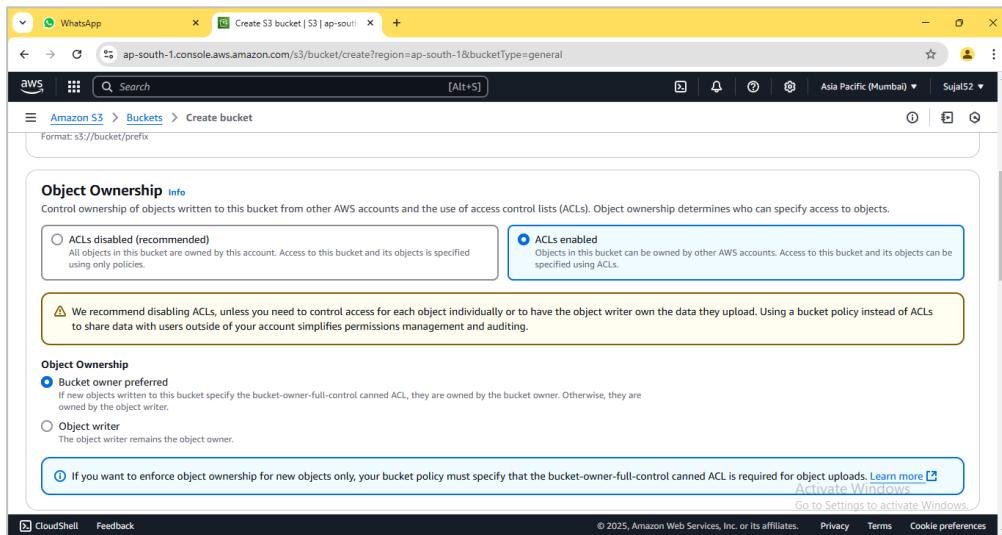
1. Access the **AWS console**, search for **S3**, and select the top option from the search results.



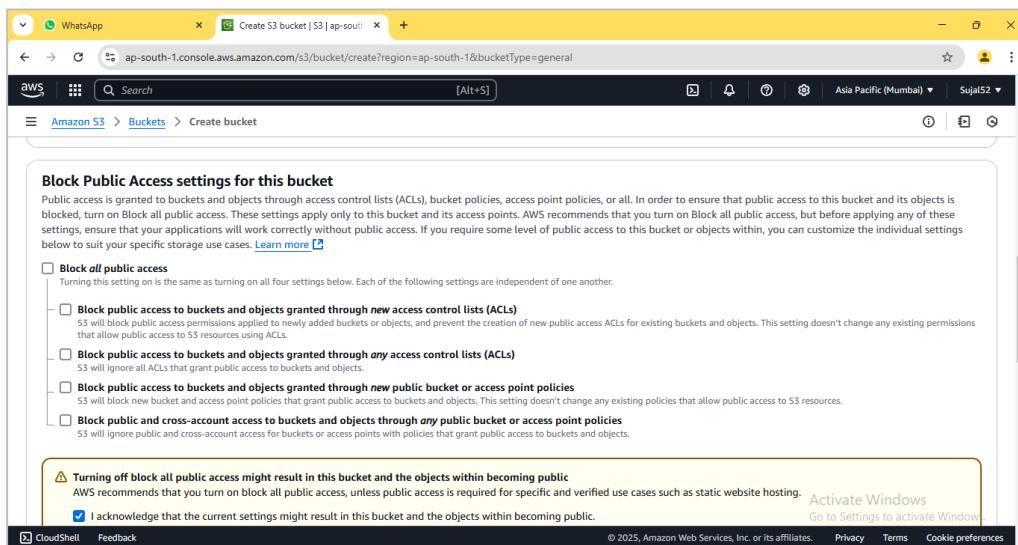
2. Click on “**Create bucket**”.



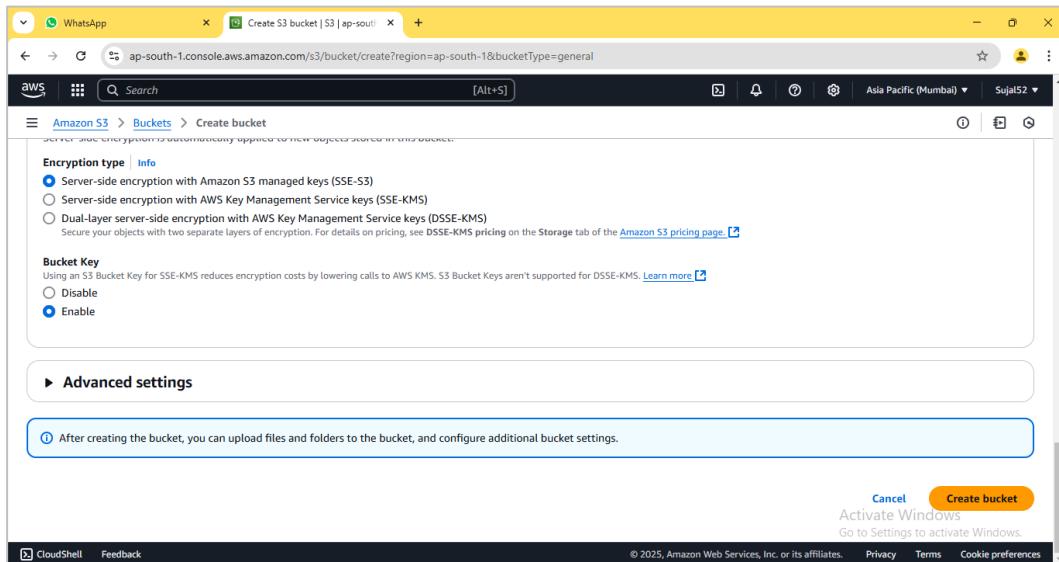
3. Enter bucket name. Select “**ACLs enabled**” since we are creating a public bucket.



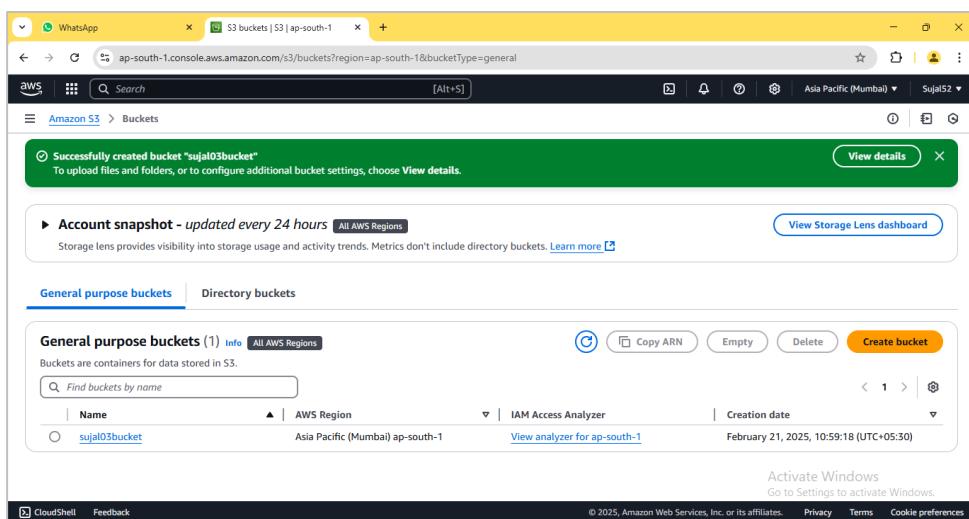
4. Now, uncheck the box for “**Block all public access**”. At the bottom of the window, check the box confirming that the current setting changes may result in a public bucket.



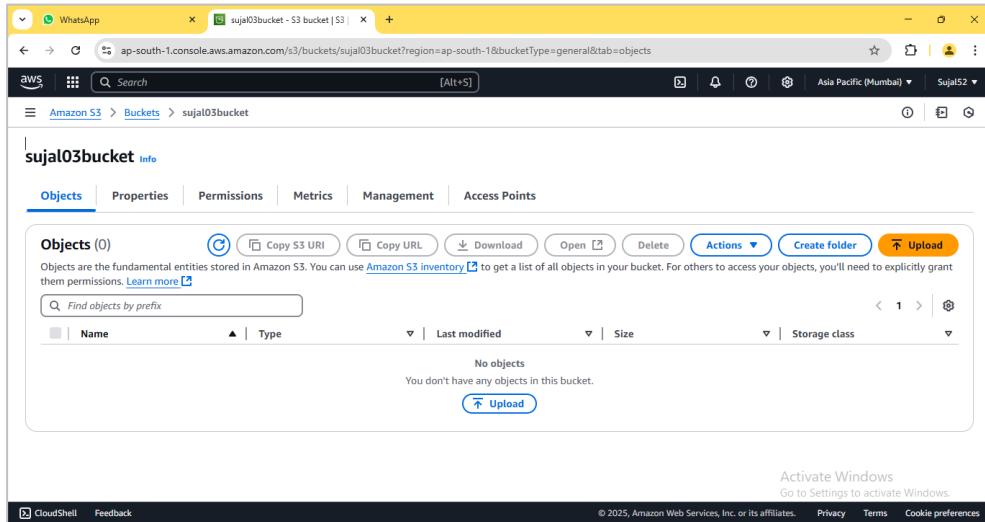
5. Scroll down without any further modifications and proceed to click on “**Create Bucket**”.



6. A green coloured pop-up dialogue box shall confirm the creation of our bucket. It shall also be displayed on the bucket list.

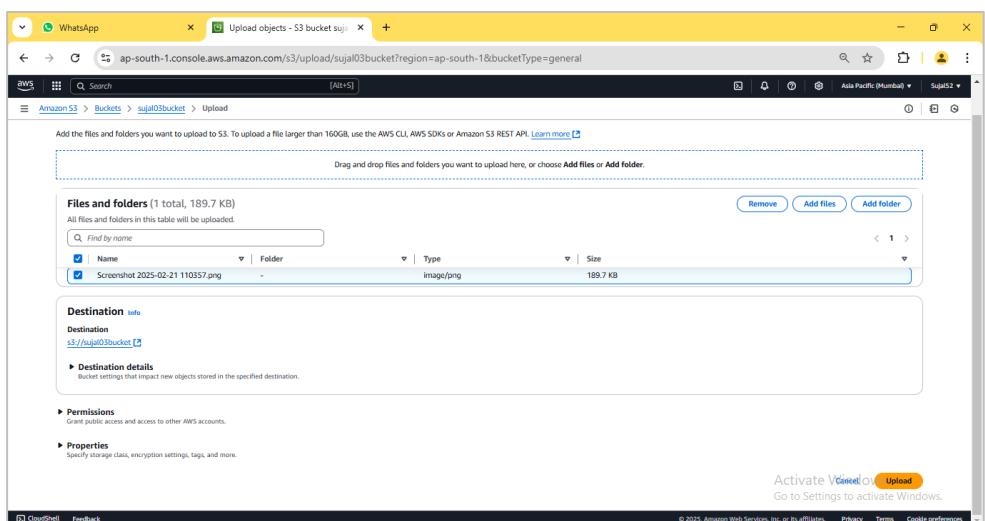


7. Click on the name of our newly created bucket. On the bucket detail page, click on “**Upload**”.



The screenshot shows the AWS S3 console. The URL is ap-south-1.console.aws.amazon.com/s3/buckets/sujal03bucket?region=ap-south-1&bucketType=general&t=tab=objects. The page title is "sujal03bucket - S3 bucket | S3". The top navigation bar includes "aws", "Search [Alt+S]", and "Actions". The main content area shows the bucket name "sujal03bucket" and tabs for "Objects", "Properties", "Permissions", "Metrics", "Management", and "Access Points". Under the "Objects" tab, there is a table with one row: "No objects". A large orange "Upload" button is located at the bottom right of the table. The status bar at the bottom right says "Activate Windows Go to Settings to activate Windows."

8. Click on “**Add files**” then select the file we want to upload to our bucket. Select our file from the list and click on “**Upload**” button.



The screenshot shows the "Upload objects" page for the "sujal03bucket" bucket. The URL is ap-south-1.console.aws.amazon.com/s3/upload/sujal03bucket?region=ap-south-1&bucketType=general. The top navigation bar includes "aws", "Search [Alt+S]", and "Actions". The main content area has a message "Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDKs or Amazon S3 REST API." Below this is a "Drag and drop Files and folders you want to upload here, or choose Add files or Add folder." section. A table lists "Files and folders" (1 total, 189.7 KB) with one item: "Screenshot 2025-02-21 110357.png" (image/png, 189.7 KB). Buttons for "Remove", "Add files", and "Add folder" are available. Below the table are sections for "Destination info" (set to "s3://sujal03bucket") and "Destination details". The status bar at the bottom right says "Activate Windows Go to Settings to activate Windows."

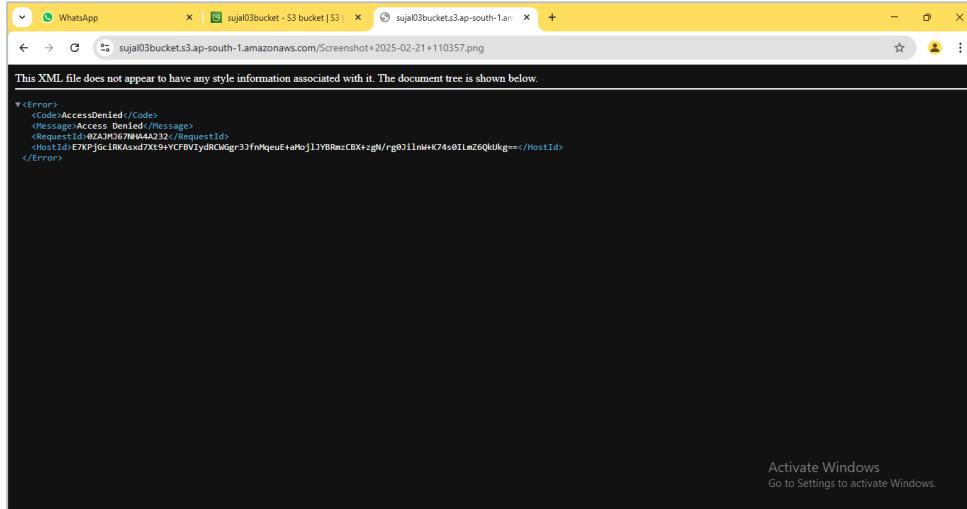
9. The notification confirms that the file has been successfully uploaded. Click on “**Close**” to return to our bucket details page.

The screenshot shows the AWS S3 console interface. At the top, a green banner displays "Upload succeeded" and "For more information, see the Files and folders table." Below this, the "Summary" section shows "Destination s3://sujal03bucket" with "Succeeded" status (1 file, 189.7 KB (100.00%)) and "Failed" status (0 files, 0 B (0%)). The "Files and folders" tab is selected, showing a table with one item: "Screenshot 2025-02-21 110357.png" (image/png, 189.7 KB, Succeeded). The bottom of the screen includes standard AWS navigation links like CloudShell, Feedback, and cookie preferences.

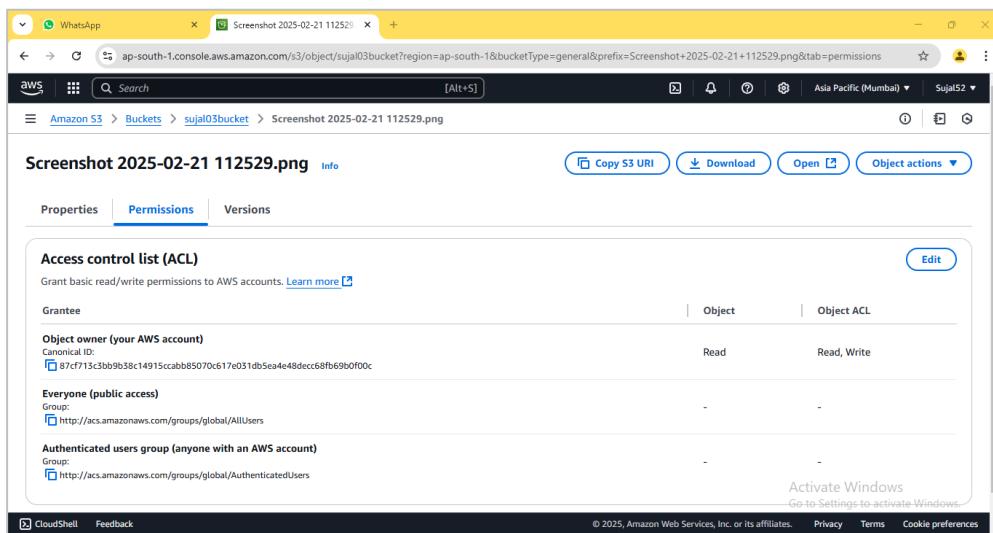
10. Click on our previously uploaded file from the list. Copy the “**Object URL**” from the file details page that opens.

The first screenshot shows the "Objects" list in the S3 console, displaying one item: "Screenshot 2025-02-21 110357.png" (png, 189.7 KB, Standard storage class). The second screenshot shows the file details page for "Screenshot 2025-02-21 110357.png", with tabs for Properties, Permissions, and Versions. The "Properties" tab is selected, showing the "Object overview" section with fields like Owner, AWS Region (Asia Pacific (Mumbai) ap-south-1), Last modified (February 21, 2025, 11:05:19 (UTC+05:30)), Size (189.7 KB), Type (png), and Key (Screenshot 2025-02-21 110357.png). On the right, the "Object URI" is listed as "https://s3.ap-south-1.amazonaws.com/s3/object/sujal03bucket/Screenshot+2025-02-21+110357.png". The third screenshot shows the "Object actions" dropdown menu open, with options including "Copy S3 URI", "Download", "Open", and "Object actions".

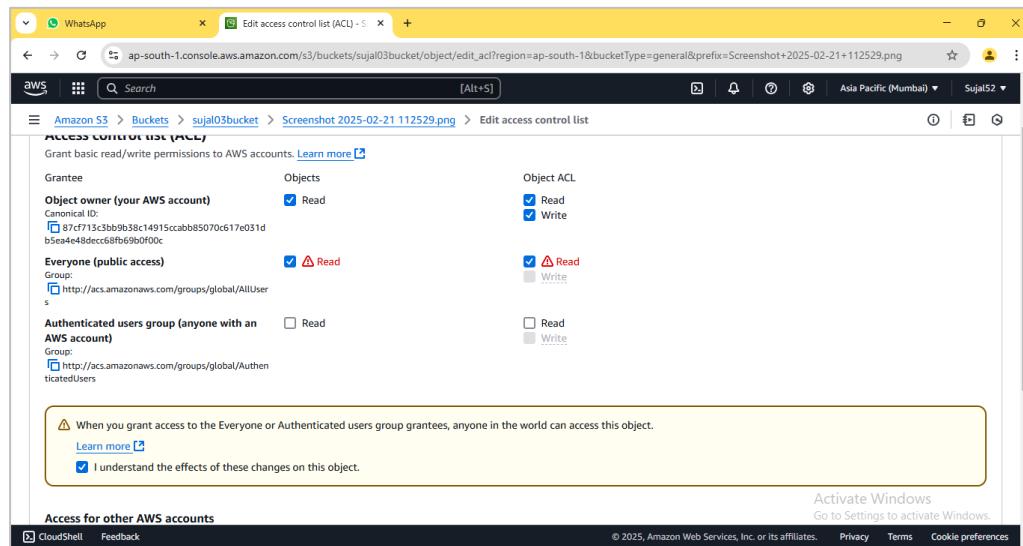
11. Open the copied URL in a new browser window. An error appears indicating that access to the contents is restricted since this is a private bucket.



12. Go back to file details page and click on “**Permissions**” tab. Then click on “**Edit**” button.



13. Edit the permissions as follow then click on “**Save changes**”.



The screenshot shows the 'Edit access control list (ACL)' page for an S3 bucket. Under the 'Everyone (public access)' section, the 'Read' checkbox is checked, while 'Write' is unchecked. A warning message states: '⚠️ When you grant access to the Everyone or Authenticated users group grantees, anyone in the world can access this object.' Below the warning, there is a checkbox for 'I understand the effects of these changes on this object.' At the bottom of the page, there is a 'Save changes' button.

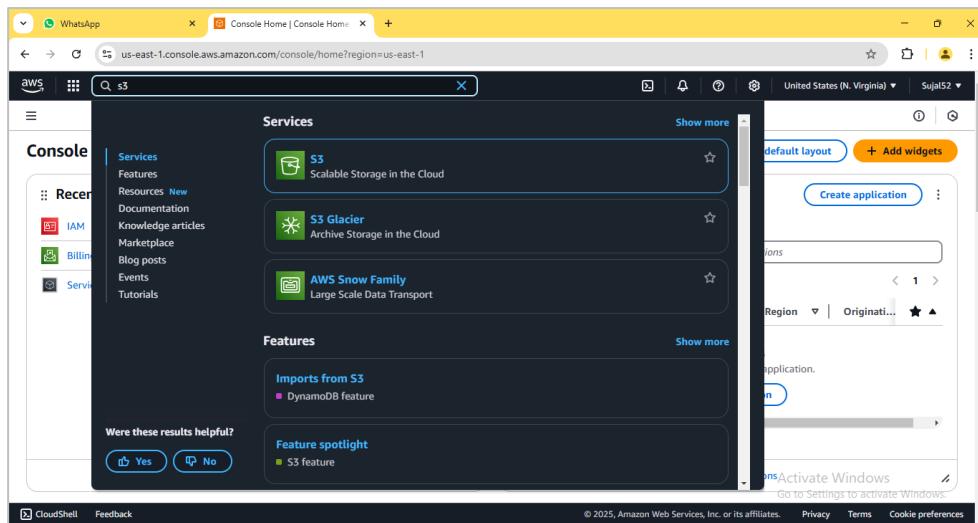
14. Copy the URL as mentioned previously and open it on new window.

ASSIGNMENT 6:

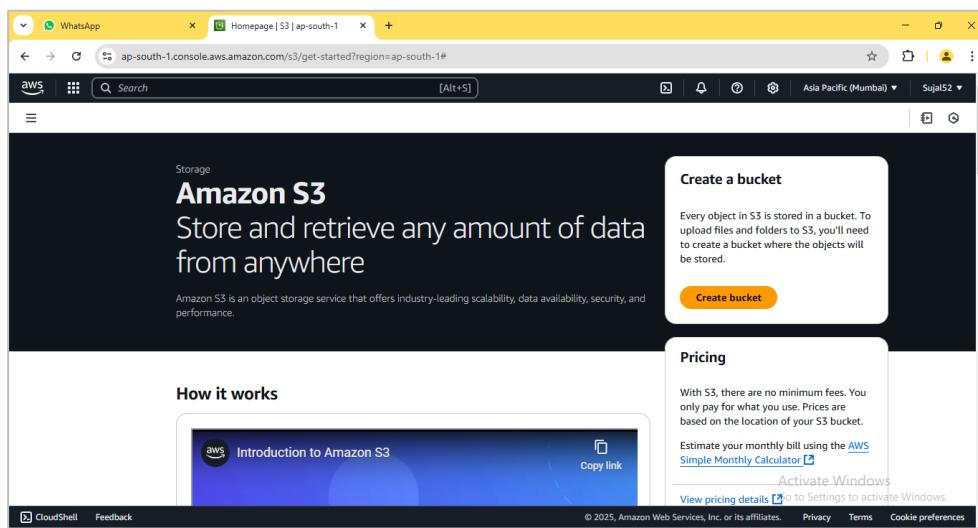
Problem Definition: Upload a static website on S3.

Instructions:

1. Access the **AWS console**, search for **S3**, and select the top option from the search results.



2. Click on “**Create bucket**”.



3. Enter bucket name. Select “**ACLs enabled**” since we are creating a public bucket. Uncheck the box for “**Block all public access**”. Leave other options as default and click on “**Create bucket**”.

The screenshots show the 'Create bucket' wizard in the AWS Management Console. The first step shows 'Object Ownership' where 'ACLs enabled' is selected. The second step shows 'Block Public Access settings for this bucket' with the 'Block all public access' checkbox unselected. The third step shows 'Encryption type' where 'Server-side encryption with Amazon S3 managed keys (SSE-S3)' is selected. All steps include a note about activating Windows.

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.

- ACLs disabled (recommended)**
All objects in this bucket are owned by this account. Access to this bucket and its objects is specified using only policies.
- ACLs enabled**
Objects in this bucket can be owned by other AWS accounts. Access to this bucket and its objects can be specified using ACLs.

⚠️ We recommend disabling ACLs, unless you need to control access for each object individually or to have the object writer own the data they upload. Using a bucket policy instead of ACLs to share data with users outside of your account simplifies permissions management and auditing.

Object Ownership

- Bucket owner preferred**
If new objects written to this bucket specify the bucket-owner-full-control canned ACL, they are owned by the bucket owner. Otherwise, they are owned by the object writer.
- Object writer**
The object writer remains the object owner.

ⓘ If you want to enforce object ownership for new objects only, your bucket policy must specify that the bucket-owner-full-control canned ACL is required for object uploads. [Learn more](#) [Activate Windows](#) [Go to Settings to activate Windows](#)

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.

⚠️ Turning off block all public access might result in this bucket and the objects within becoming public
AWS recommends that you turn on block all public access, unless public access is required for specific and verified use cases such as static website hosting. [Activate Windows](#) [Go to Settings to activate Windows](#)

I acknowledge that the current settings might result in this bucket and the objects within becoming public.

Encryption type

- Server-side encryption with Amazon S3 managed keys (SSE-S3)**
- Server-side encryption with AWS Key Management Service keys (SSE-KMS)
- Dual-layer server-side encryption with AWS Key Management Service keys (DSS-E-KMS)
Secure your objects with two separate layers of encryption. For details on pricing, see [DSSE-KMS pricing](#) on the Storage tab of the [Amazon S3 pricing page](#).

Bucket Key
Using an S3 Bucket Key for SSE-KMS reduces encryption costs by lowering calls to AWS KMS. S3 Bucket Keys aren't supported for DSSE-KMS. [Learn more](#)

- Disable**
- Enable**

Advanced settings

ⓘ After creating the bucket, you can upload files and folders to the bucket, and configure additional bucket settings.

Create bucket

4. A green coloured pop-up dialogue box shall confirm the creation of our bucket. It shall also be displayed on the bucket list.

The screenshot shows the AWS S3 console in a web browser. At the top, there's a green success message: "Successfully created bucket 'sujal03bucket'. To upload files and folders, or to configure additional bucket settings, choose View details." Below this, there's an "Account snapshot" section with a note about storage usage and activity trends. The main area shows a table of "General purpose buckets". One row is highlighted with a blue background, showing the bucket name "sujal03bucket", its AWS Region "Asia Pacific (Mumbai) ap-south-1", and its creation date "February 21, 2025, 10:59:18 (UTC+05:30)". There are buttons for "Copy ARN", "Empty", "Delete", and "Create bucket".

5. Click on the name of our newly created bucket. On the bucket detail page, click on “**Upload**”.

The screenshot shows the AWS S3 console on the "Objects" tab of the "sujal03bucket" detail page. At the top, there's a header with tabs for "Objects", "Properties", "Permissions", "Metrics", "Management", and "Access Points". Below this, there's a toolbar with buttons for "Copy S3 URI", "Copy URL", "Download", "Open", "Delete", "Actions", "Create folder", and "Upload". The main area shows a table with one row: "No objects". A message below the table says "You don't have any objects in this bucket." At the bottom right of the table area, there's a large blue "Upload" button.

6. Click on “**Add files**”. Select our html files from the list. Expand “**Permissions**” tab, select “**Grant public read access**” and check the confirmation popup. Finally, click on “**Upload**” button.

Access control list (ACL)

AWS recommends using S3 bucket policies or IAM policies for access control.

Access control list (ACL)

- Choose from predefined ACLs
- Specify individual ACL permissions

Predefined ACLs

- Private (recommended) Only the object owner will have read and write access.
- Grant public-read access Anyone in the world will be able to access the specified objects. The object owner will have read and write access.

Granting public-read access is not recommended

Anyone in the world will be able to access the specified objects.

I understand the risk of granting public-read access to the specified objects.

Properties

Specify storage class, encryption settings, tags, and more.

Upload

Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDKs or Amazon S3 REST API.

Drag and drop files and folders you want to upload here, or choose **Add files** or **Add folder**.

Files and folders (2 total, 652.0 B)

Name	Type	Size
about.html	text/html	326.0 B
index.html	text/html	326.0 B

Destination

Destination

s3://staticwebsitesujal03

Destination details

Bucket settings that impact new objects stored in the specified destination.

Permissions

Grant public access and access to other AWS accounts.

Activate Windows

Go to Settings to activate Windows.

7. The notification confirms that the file has been successfully uploaded. Click on “***Close***” to return to our bucket details page.

Upload objects - S3 bucket stat

ap-south-1.console.aws.amazon.com/s3/upload/staticwebsitesujal03?region=ap-south-1&bucketType=general

aws Search [Alt+S]

Upload succeeded
For more information, see the Files and folders table.

Upload: status

After you navigate away from this page, the following information is no longer available.

Summary

Destination	Succeeded	Failed
s3://staticwebsitesujal03	2 files, 652.0 B (100.00%)	0 files, 0 B (0%)

Files and folders Configuration

Files and folders (2 total, 652.0 B)

Name	Folder	Type	Size	Status	Error
about.html	-	text/html	326.0 B	SUCCEEDED	-
index.html	-	text/html	326.0 B	SUCCEEDED	-

Activate Windows
Go to Settings to activate Windows.

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

8. From the bucket details page, click on “***Properties***” tab, scroll down and click on “***Edit***” next to “***Static website hosting***”.

staticwebsitesujal03 - S3 bucket

ap-south-1.console.aws.amazon.com/s3/buckets/staticwebsitesujal03?region=ap-south-1&tab=properties

Amazon S3 > Buckets > staticwebsitesujal03

Disabled

Object Lock
Store objects using a write-once-read-many (WORM) model to help you prevent objects from being deleted or overwritten for a fixed amount of time or indefinitely. Object Lock works only in versioned buckets. [Learn more](#)

Object Lock
Disabled

Requester pays
When enabled, the requester pays for requests and data transfer costs, and anonymous access to this bucket is disabled. [Learn more](#)

Requester pays
Disabled

Static website hosting
Use this bucket to host a website or redirect requests. [Learn more](#)

We recommend using AWS Amplify Hosting for static website hosting
Deploy a fast, secure, and reliable website quickly with AWS Amplify Hosting. Learn more about [Amplify Hosting](#) or [View your existing Amplify apps](#).

Create Amplify app

S3 static website hosting
Disabled

Activate Windows
Go to Settings to activate Windows.

CloudShell Feedback © 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

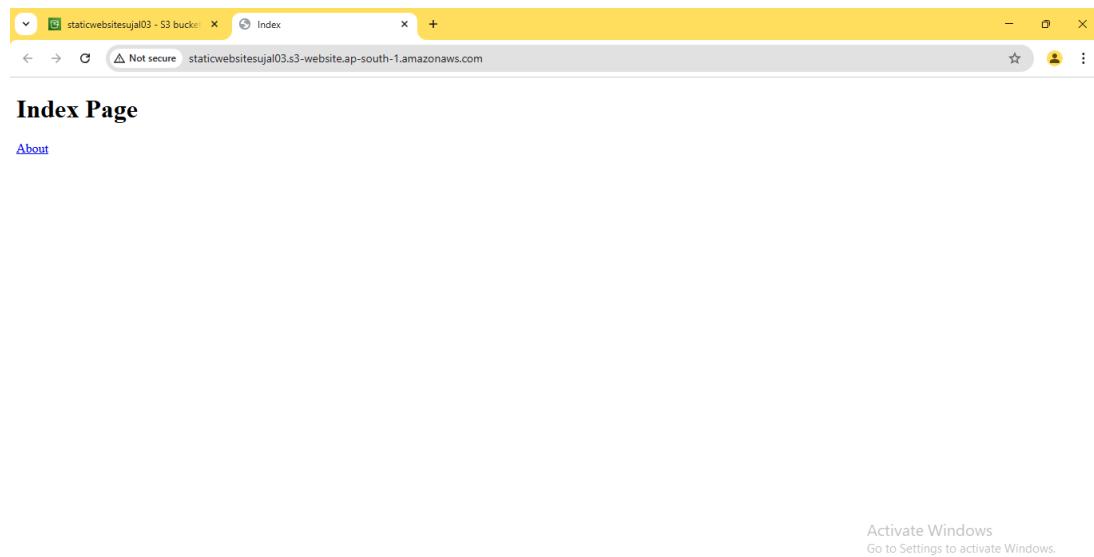
9. Under “**Static website hosting**”, click on “**Enable**”, then fill in the name of index document and click on “**Save changes**”.

The screenshot shows the 'Edit static website hosting' configuration for the 'staticwebsitelujal03' bucket. The 'Static website hosting' section is open, showing the 'Enable' radio button selected. The 'Index document' field contains 'index.html'. A note at the bottom states: '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.' The 'Activate Windows' watermark is visible in the top right corner.

10. Once again, scroll down to “**Static website hosting**” and copy the URL under “**Bucket website endpoint**”.

The screenshot shows the 'Properties' tab of the 'staticwebsitelujal03' bucket. The 'Static website hosting' section is expanded, showing the 'Enabled' status and the 'Bucket website endpoint' URL: 'http://staticwebsitelujal03.s3-website.ap-south-1.amazonaws.com'. A note at the bottom of this section says: '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'.

11. Open the copied URL in a new browser window. Our website should be accessible now.

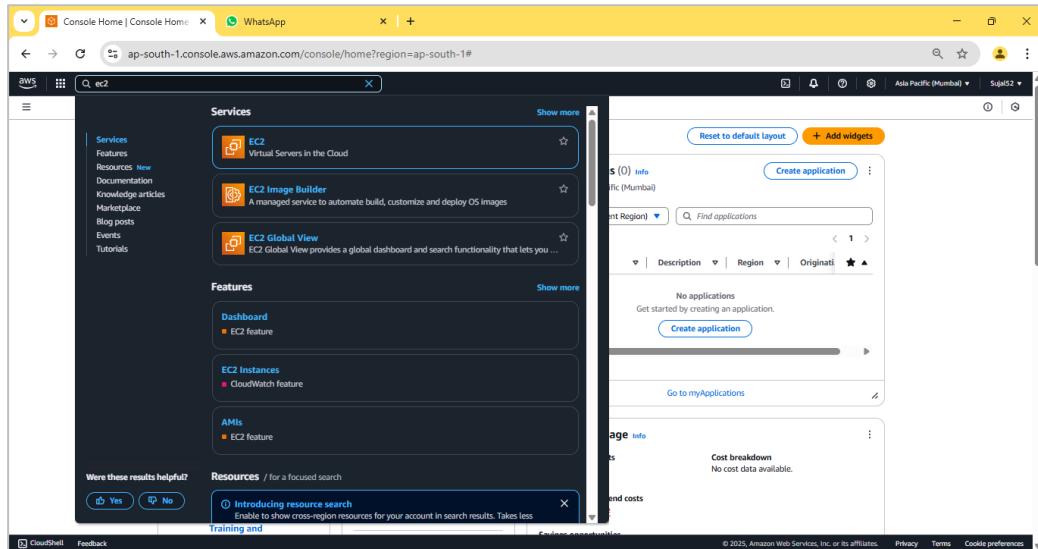


ASSIGNMENT 7:

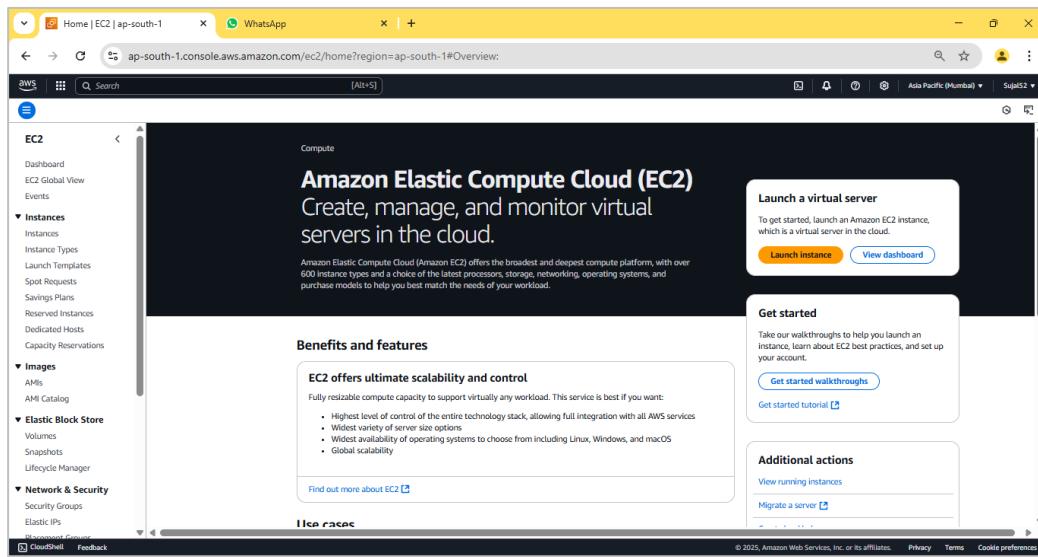
Problem Definition: Hosting a website on EC2.

Instructions:

1. Access the **AWS console**, search for **EC2**, and select the top option from the search results.



2. Click on "**Launch instance**".



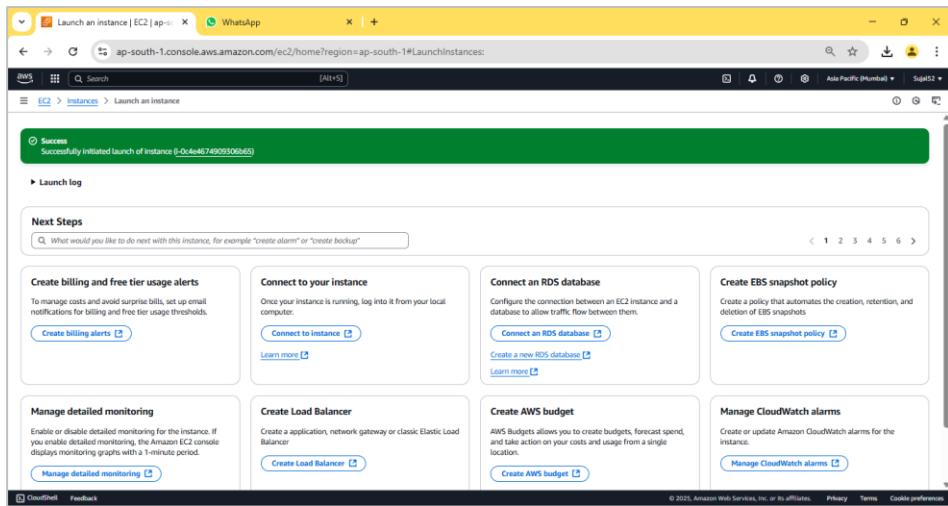
3. Enter instance name and select **Ubuntu** under “**Application and OS Images**”. Under “**Key Pair (login)**” click on “**Create new key pair**”, enter key pair name then click on “**Create key pair**” button, a *.pem file will be downloaded. Then click on “**Launch instance**” button

The image consists of two vertically stacked screenshots of the AWS CloudFormation 'Launch an instance' wizard.

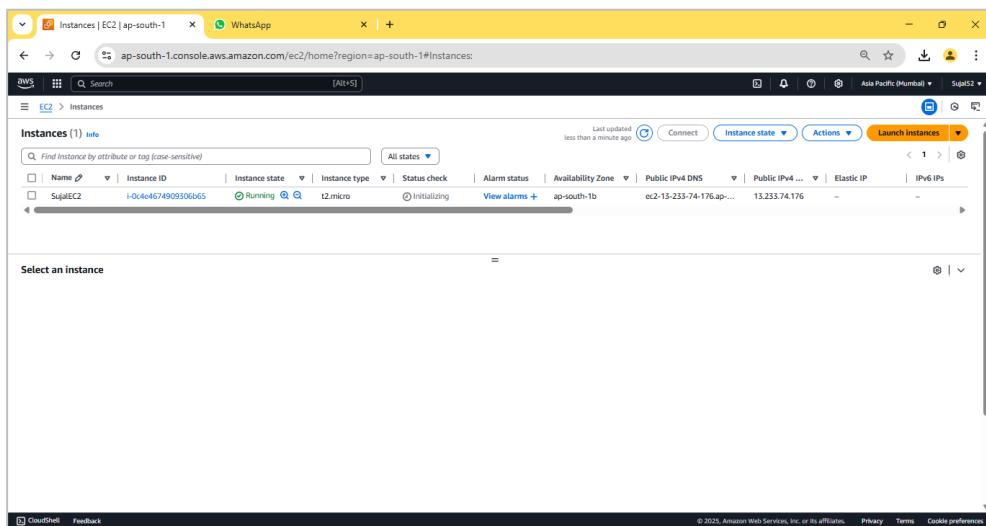
Top Screenshot: This screenshot shows the 'Application and OS Images (Amazon Machine Image)' step. It displays a grid of icons for various AMIs, including Amazon Linux, macOS, Ubuntu, Windows, Red Hat, SUSE Linux, and Debian. The 'Ubuntu' icon is selected. On the right side of the screen, there is a summary panel showing the configuration details: Software Image (AMI) Canonical, Ubuntu, 24.04, amd64, Virtual server type (instance type) t2.micro, and Storage (volumes) 1 volume(s) - 8 GiB. At the bottom right of this panel is a yellow 'Launch instance' button.

Bottom Screenshot: This screenshot shows the 'Create key pair' step. It has a form for entering a key pair name ('mykey123') and two radio buttons for 'Key pair type': 'RSA' (selected) and 'EC2SSN19'. Below the form is a note: 'When prompted, store the private key in a secure and accessible location on your computer. You will need it later to connect to your instance.' A yellow 'Create key pair' button is located at the bottom right of this section. The rest of the interface is identical to the top screenshot, including the summary panel and the yellow 'Launch instance' button at the bottom right.

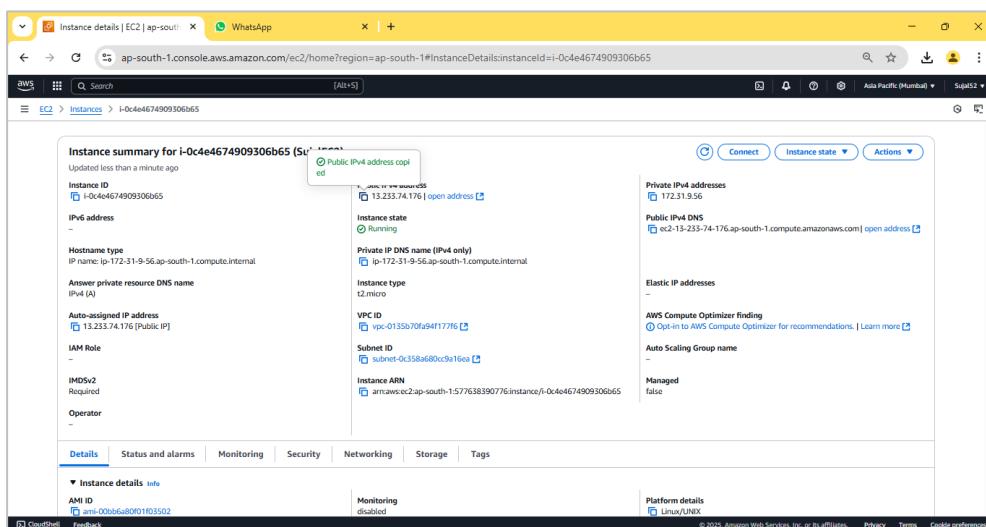
4. Our instance is created successfully.



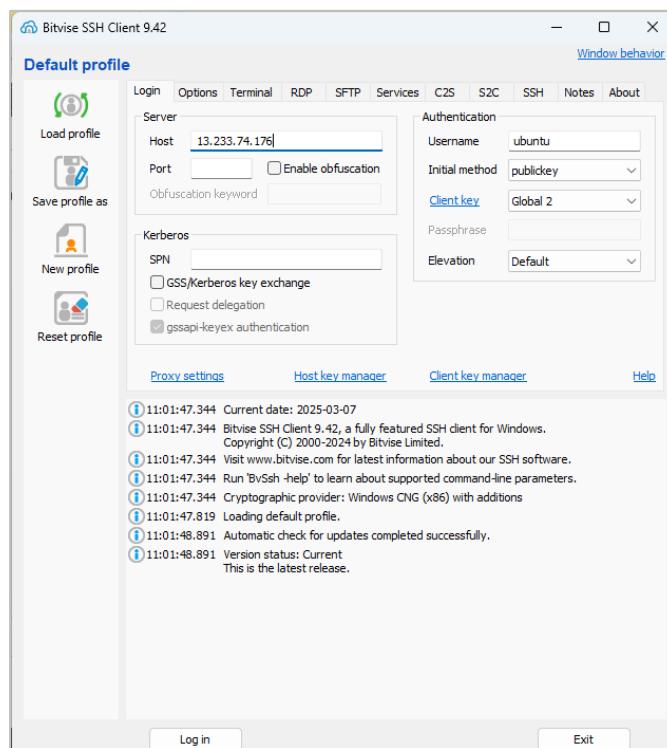
5. Go to “**Instance**” from the sidebar, then click on **Instance ID** of our newly created instance.



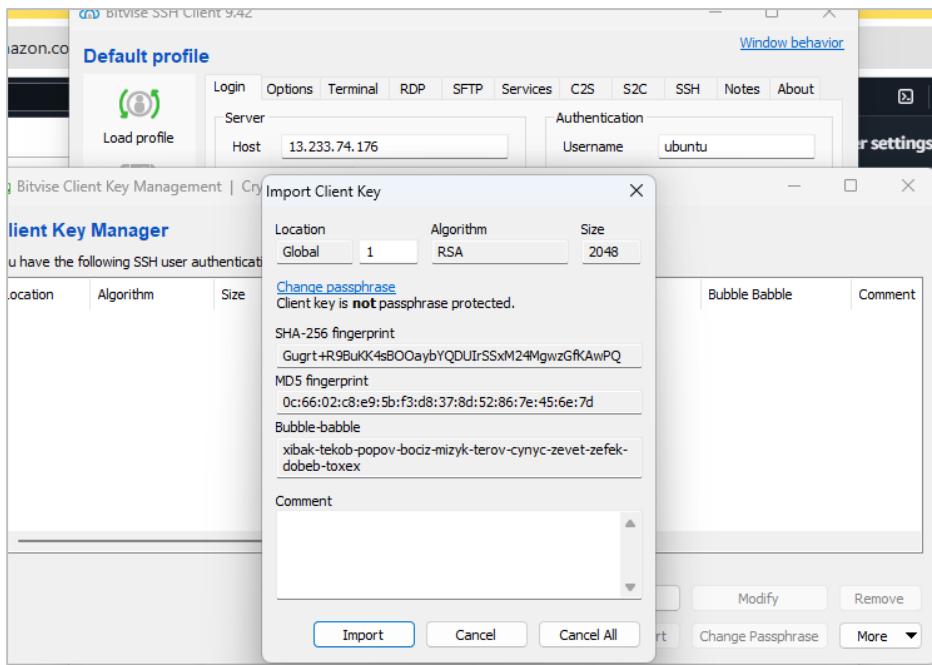
6. Copy “**Public IPV4 address**”.



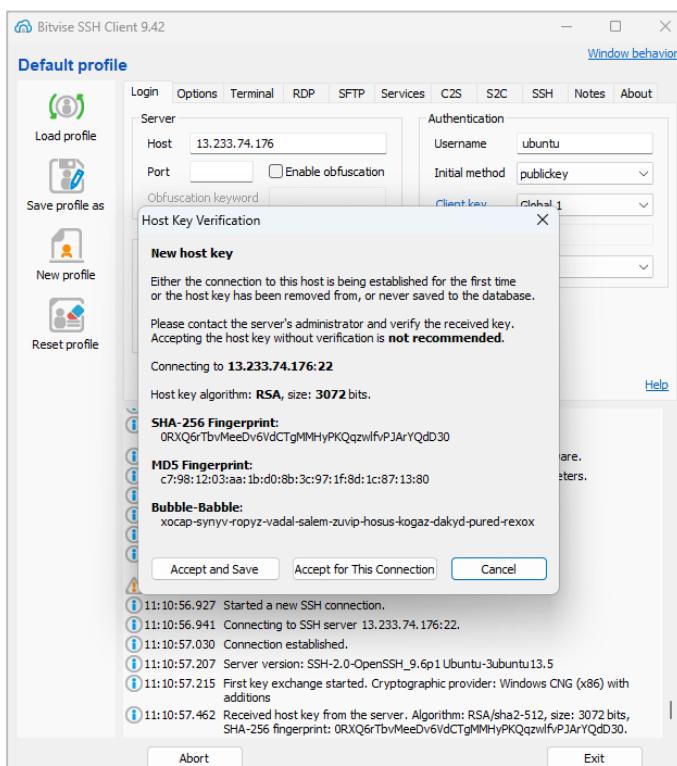
7. Open **Bitvise SSH Client**, paste the previously copied address in host field.



8. Click "**Client key manager**", click on "**Import**" and select our previously downloaded key pair file. Click on "**Import**" button. Our key pair should be imported successfully.

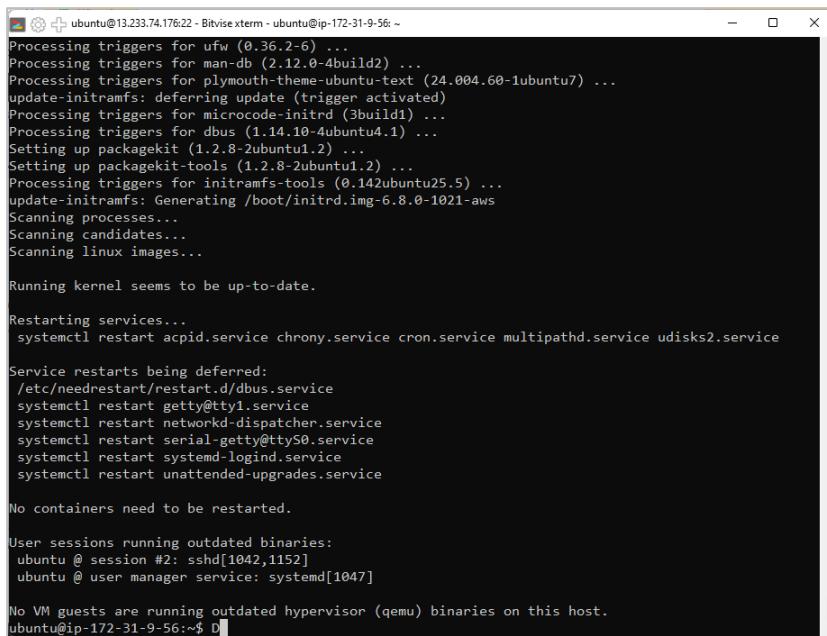


9. In “**Bitvise SSH Client**”, under “**Authentication**” give the username as “**ubuntu**”, select “**Global 1**” under “**Client Key**” then click on “**Log in**” & “**Accept and save**”.



10. Click on “**New terminal console**” from the sidebar. In the terminal enter the following commands:
 sudo apt-get update && sudo apt-get upgrade

```
sudo apt-get install nginx
```



```
ubuntu@13.233.74.176:22 - Bitvise XTerm - ubuntu@ip-172-31-9-56: ~
Processing triggers for ufw (0.36.2-6) ...
Processing triggers for man-db (2.12.0-4build2) ...
Processing triggers for plymouth-theme-ubuntu-text (24.004.60-1ubuntu7) ...
update-initramfs: deferring update (trigger activated)
Processing triggers for microcode-initrd (3build1) ...
Processing triggers for dbus (1.14.10-4ubuntu4.1) ...
Setting up packagekit (1.2.8-2ubuntu1.2) ...
Setting up packagekit-tools (1.2.8-2ubuntu1.2) ...
Processing triggers for initramfs-tools (0.142ubuntu25.5) ...
update-initramfs: Generating /boot/initrd.img-6.8.0-1021-aws
Scanning processes...
Scanning candidates...
Scanning linux images...

Running kernel seems to be up-to-date.

Restarting services...
systemctl restart acpid.service chrony.service cron.service multipathd.service udisks2.service

Service restarts being deferred:
/etc/needrestart/restart.d/dbus.service
systemctl restart getty@tty1.service
systemctl restart networkd-dispatcher.service
systemctl restart serial-getty@ttyS0.service
systemctl restart systemd-logind.service
systemctl restart unattended-upgrades.service

No containers need to be restarted.

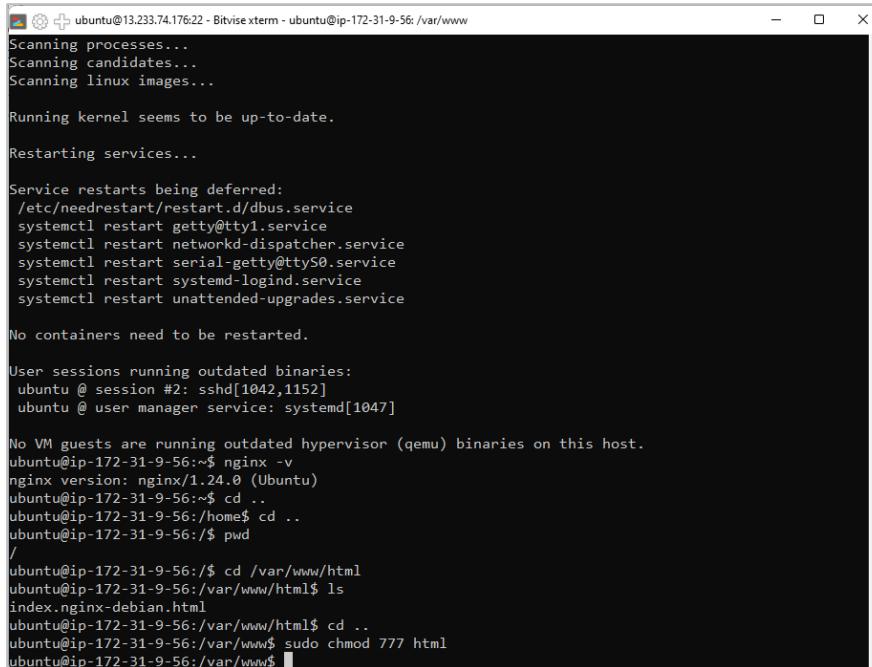
User sessions running outdated binaries:
ubuntu @ session #2: sshd[1042,1152]
ubuntu @ user manager service: systemd[1047]

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-9-56:~$ D
```

11. We need to move our html files into var/www/html directory to host it, but first we need to give write permission to do so. To change file permission enter the following command:

```
cd var/www/
```

```
sudo chmod 777 html
```



```
ubuntu@13.233.74.176:22 - Bitvise XTerm - ubuntu@ip-172-31-9-56:/var/www
Scanning processes...
Scanning candidates...
Scanning linux images...

Running kernel seems to be up-to-date.

Restarting services...

Service restarts being deferred:
/etc/needrestart/restart.d/dbus.service
systemctl restart getty@tty1.service
systemctl restart networkd-dispatcher.service
systemctl restart serial-getty@ttyS0.service
systemctl restart systemd-logind.service
systemctl restart unattended-upgrades.service

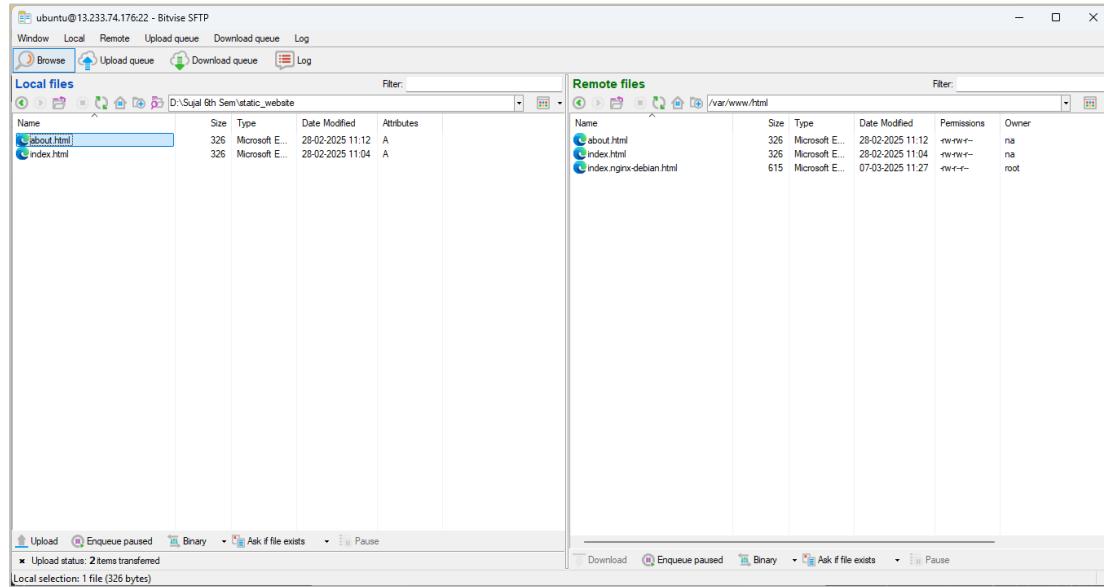
No containers need to be restarted.

User sessions running outdated binaries:
ubuntu @ session #2: sshd[1042,1152]
ubuntu @ user manager service: systemd[1047]

No VM guests are running outdated hypervisor (qemu) binaries on this host.
ubuntu@ip-172-31-9-56:~$ nginx -v
nginx version: nginx/1.24.0 (Ubuntu)
ubuntu@ip-172-31-9-56:~$ cd ..
ubuntu@ip-172-31-9-56:~/home$ cd ..
ubuntu@ip-172-31-9-56:/$ pwd
/
ubuntu@ip-172-31-9-56:/$ cd ./var/www/html
ubuntu@ip-172-31-9-56:~/var/www/html$ ls
index.nginx-debian.html
ubuntu@ip-172-31-9-56:~/var/www/html$ cd ..
ubuntu@ip-172-31-9-56:~/var/www$ sudo chmod 777 html
ubuntu@ip-172-31-9-56:~/var/www$
```

12. Go to SFTP window from the sidebar, under **Local file** open the

folder containing html files and under **Remote file** open /var/www/html. Drag and drop the HTML files from local to remote.



13. Now go back to the AWS window and copy the “**Public IPV4 address**”.

Instance summary for i-0c4e4674909306b65 (Sustained)

Updated less than a minute ago

Instance ID: i-0c4e4674909306b65

IPv6 address: -

Hostname type: IP name: ip-172-31-9-56.ap-south-1.compute.internal

Answer private resource DNS name: IPv4 (A)

Auto-assigned IP address: 13.233.74.176 [Public IP]

IAM Role: -

IMDSv2: Required

Operator: -

Details | Status and alarms | Monitoring | Security | Networking | Storage | Tags

Instance details | Info

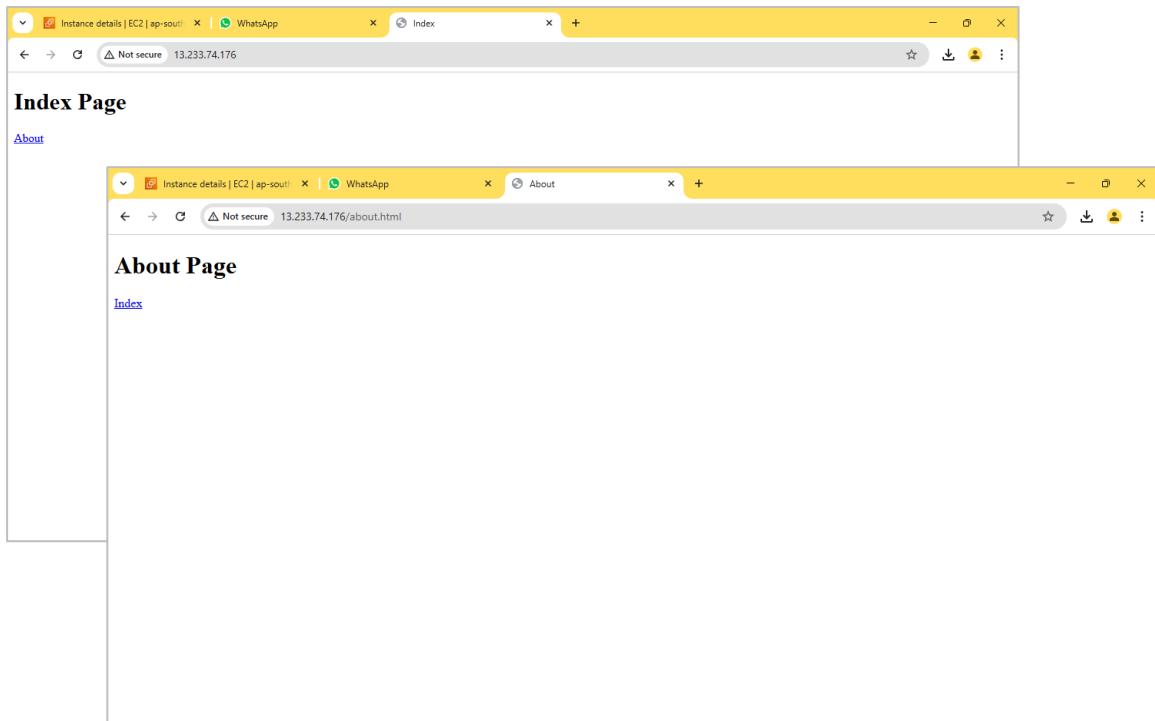
AMI ID: ami-00b6a80f01f03502

Monitoring: disabled

Platform details: Linux/UNIX

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

14. Open the copied address in a new browser window. Our website should be accessible now.



ASSIGNMENT 8:

Problem Definition: Deploy a project from a local machine to GitHub and vice versa.

Instructions:

1. Create a new public Github repository. Click on “**Create repository**” to proceed.

Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository](#).

Required fields are marked with an asterisk ().*

Owner *  **Repository name ***

 **SujalGupta52** 

/

✔ SDITO_Lab is available.

Great repository names are short and memorable. Need inspiration? How about [scaling-eureka](#) ?

Description (optional)

 **Public**
 Anyone on the internet can see this repository. You choose who can commit.

 **Private**
 You choose who can see and commit to this repository.

Initialize this repository with:

Add a README file
 This is where you can write a long description for your project. [Learn more about READMEs](#).

Add .gitignore

 Choose which files not to track from a list of templates. [Learn more about ignoring files](#).

Choose a license

 A license tells others what they can and can't do with your code. [Learn more about licenses](#).

ⓘ You are creating a public repository in your personal account.

[Create repository](#)

2. Go to **Settings** then **Developer settings**. Now click on “**Personal access tokens**” dropdown and then click on “**Tokens(classic)**”.

The screenshot shows the GitHub user profile sidebar on the left and the "Personal access tokens" section of the "Developer settings" page on the right.

User Profile Sidebar:

- SujalGupta52
- Sujal Gupta
- Set status
- Your profile
- Your repositories
- Your Copilot
- Your projects
- Your stars
- Your gists
- Your organizations
- Your enterprises
- Your sponsors
- Try Enterprise (Free)
- Feature preview
- Settings
- GitHub Website
- GitHub Docs
- GitHub Support
- GitHub Community
- Sign out

Personal access tokens Section:

- GitHub Apps
- OAuth Apps
- Personal access tokens** (highlighted with a blue border)
- Fine-grained tokens
- Tokens (classic)

Text Overlay: And then “Generate new”

3. Now go to [Developer settings](#) and then “**Generate new**

The screenshot shows the GitHub Developer Settings interface. On the left sidebar, there are three main items: GitHub Apps, OAuth Apps, and Personal access tokens. The Personal access tokens item is currently selected, indicated by a blue border around its icon and the text "Personal access tokens". Below this, two sub-options are listed: "Fine-grained tokens" and "Tokens (classic)".

The main content area is titled "Personal access tokens (classic)". It contains a button labeled "Generate new token" with a dropdown arrow. Below this, a section titled "Tokens you have generated" lists several tokens:

- mytoken — admin:enterprise
- admin:gpg_key, admin:org

Each token entry has a "Delete" link at the end.

4. Set expiration date for as long as possible and select all scopes. Then click on “**Generate token**”.

New personal access token (classic)

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Note
mytoken

What's this token for?

Expiration

The token will expire on the selected date

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes](#).

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input type="checkbox"/> <code>repo:status</code>	Access commit status
<input type="checkbox"/> <code>repo:deployment</code>	Access deployment status
<input type="checkbox"/> <code>public_repo</code>	Access public repositories
<input type="checkbox"/> <code>repo:invite</code>	Access repository invitations
<input type="checkbox"/> <code>security_events</code>	Read and write security events
<input checked="" type="checkbox"/> workflow	Update GitHub Action workflows
<input checked="" type="checkbox"/> write:packages	Upload packages to GitHub Package Registry
<input type="checkbox"/> <code>read:packages</code>	Download packages from GitHub Package Registry
<input checked="" type="checkbox"/> delete:packages	Delete packages from GitHub Package Registry
<input checked="" type="checkbox"/> admin:org	Full control of orgs and teams, read and write org projects
<input type="checkbox"/> <code>write:org</code>	Read and write org and team membership, read and write org projects
<input type="checkbox"/> <code>read:org</code>	Read org and team membership, read org projects
<input type="checkbox"/> <code>manage_runners:org</code>	Manage org runners and runner groups
<input checked="" type="checkbox"/> <code>admin:public_key</code>	Full control of user public keys
<input type="checkbox"/> <code>write:public_key</code>	Write user public keys
<input type="checkbox"/> <code>read:public_key</code>	Read user public keys
<input checked="" type="checkbox"/> <code>admin:repo_hook</code>	Full control of repository hooks
<input type="checkbox"/> <code>write:repo_hook</code>	Write repository hooks

<input checked="" type="checkbox"/> admin:enterprise	Full control of enterprises
<input type="checkbox"/> <code>manage_runners:enterprise</code>	Manage enterprise runners and runner groups
<input type="checkbox"/> <code>manage_billing:enterprise</code>	Read and write enterprise billing data
<input type="checkbox"/> <code>read:enterprise</code>	Read enterprise profile data
<input type="checkbox"/> <code>scim:enterprise</code>	Provisioning of users and groups via SCIM
<input checked="" type="checkbox"/> audit_log	Full control of audit log
<input type="checkbox"/> <code>read:audit_log</code>	Read access of audit log
<input checked="" type="checkbox"/> codespace	Full control of codespaces
<input type="checkbox"/> <code>codespace:secrets</code>	Ability to create, read, update, and delete codespace secrets
<input checked="" type="checkbox"/> copilot	Full control of GitHub Copilot settings and seat assignments
<input type="checkbox"/> <code>manage_billing:copilot</code>	Edit Copilot Business seat assignments
<input checked="" type="checkbox"/> write:network_configurations	Write org hosted compute network configurations
<input type="checkbox"/> <code>read:network_configurations</code>	Read org hosted compute network configurations
<input checked="" type="checkbox"/> project	Full control of projects
<input type="checkbox"/> <code>read:project</code>	Read access of projects
<input checked="" type="checkbox"/> admin:gpg_key	Full control of public user GPG keys
<input type="checkbox"/> <code>write:gpg_key</code>	Write public user GPG keys
<input type="checkbox"/> <code>read:gpg_key</code>	Read public user GPG keys
<input checked="" type="checkbox"/> admin:ssh_signing_key	Full control of public user SSH signing keys
<input type="checkbox"/> <code>write:ssh_signing_key</code>	Write public user SSH signing keys
<input type="checkbox"/> <code>read:ssh_signing_key</code>	Read public user SSH signing keys

5. After generating the token, click on copy icon and copy the token somewhere secure.

Personal access tokens (classic)

[Generate new token ▾](#)

Tokens you have generated that can be used to access the [GitHub API](#).

mytoken — `admin:enterprise, admin:gpg_key, admin:org, admin:org_hook, admin:public_key, admin:repo_hook, admin:ssh_signing_key, audit_log, codespace, copilot, delete:packages, delete_repo, gist, notifications, project, repo, user, workflow, write:discussion, write:network_configurations, write:packages`

Expires on **Tue, May 20 2025**.

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

6. Now go to **Control Panel**, click **User's account**, then go to **Credential Manager** and click on **Window Credentials**. From **Generic Credentials** remove any existing GitHub account credentials.

Screenshot 1: Control Panel User Accounts

- User Accounts
- Appearance and Personalization
- Clock and Region
- Ease of Access

Screenshot 2: Credential Manager

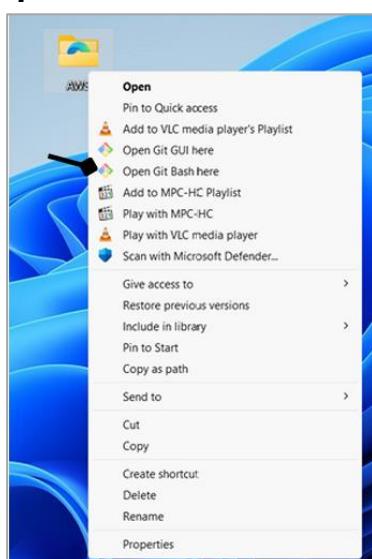
- User Accounts
- Change account type
- Remove user accounts
- Credential Manager
- Manage Web Credentials | Manage Windows Credentials

Screenshot 3: Manage your credentials

View and delete your saved logon information for websites, connected applications and networks.

Category	Action
Web Credentials	Back up Credentials Restore Credentials
Windows Credentials	Add a Windows credential
Certificate-Based Credentials	Add a certificate-based credential
Generic Credentials	Add a generic credential
com.logi.ghub/shared	Modified: 08-12-2022
Nearby_115450349601444920667	Modified: 03-03-2025
Oracle mysql-secret-store-windows-credential password	Modified: 24-06-2024
teamsliv/teams	Modified: 01-03-2023
teamsKey/teams	Modified: 01-03-2023
UnityHub/combinedTokens	Modified: 24-09-2024

7. Go to the folder containing your project files. Right Click, and select “**Open Git Bash Here**” from the pop-up menu.

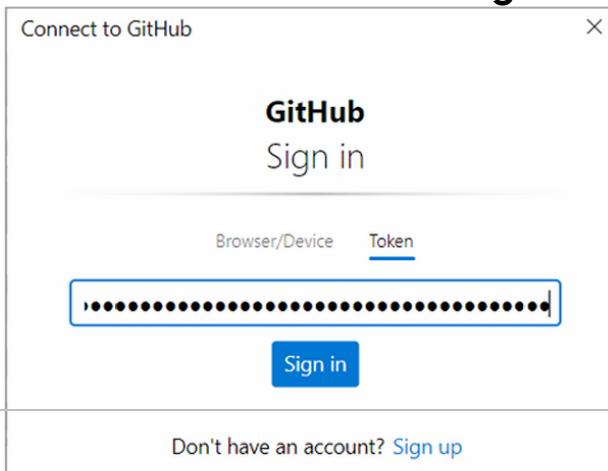


8. Run the following commands:

```
git init  
git add .  
git status  
git config --global user.email <user email address>  
git config --global user.name <username>  
git commit -m "<message>"  
git push -u origin master
```

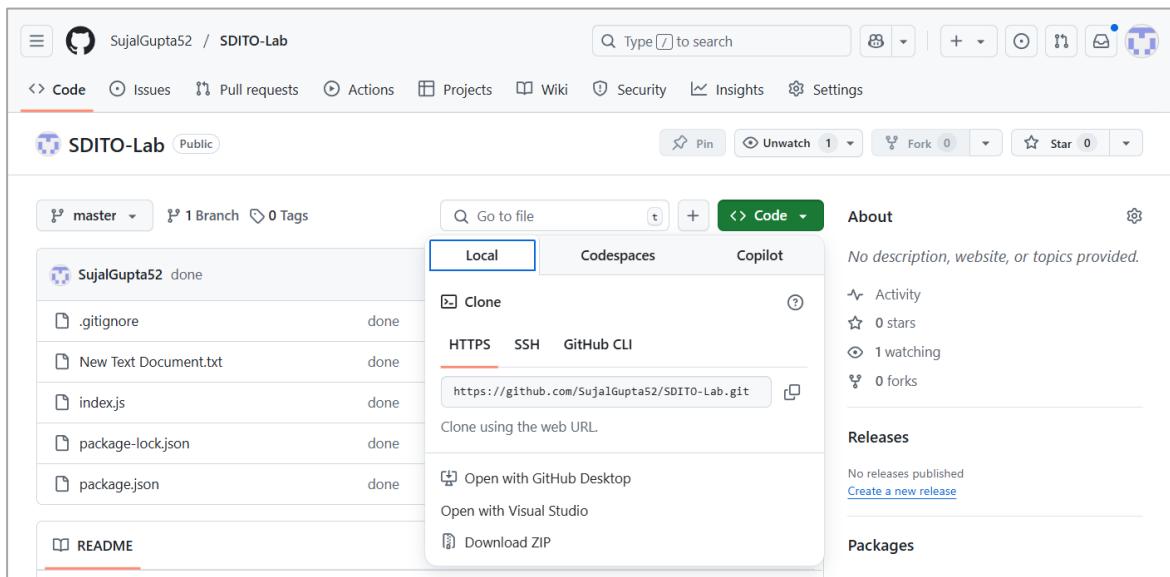
```
Meghana@DESKTOP-FBKGRS MINGW64 ~/Desktop/AWS  
$ git init  
Initialized empty Git repository in C:/Users/Meghana/Desktop/AWS/.git/  
Meghana@DESKTOP-FBKGRS MINGW64 ~/Desktop/AWS (master)  
$ dir  
index.html index1.html index2.html  
Meghana@DESKTOP-FBKGRS MINGW64 ~/Desktop/AWS (master)  
$ git add .  
warning: in the working copy of 'index.html', LF will be replaced by CRLF the next time Git touches it  
warning: in the working copy of 'index1.html', LF will be replaced by CRLF the next time Git touches it  
warning: in the working copy of 'index2.html', LF will be replaced by CRLF the next time Git touches it  
Meghana@DESKTOP-FBKGRS MINGW64 ~/Desktop/AWS (master)  
$ git status  
On branch master  
No commits yet  
Changes to be committed:  
  (use "git rm --cached <file>..." to unstage)  
    new file: index.html  
    new file: index1.html  
    new file: index2.html  
Meghana@DESKTOP-FBKGRS MINGW64 ~/Desktop/AWS (master)  
$ git config --global user.email meghana.choudhary1504@gmail.com  
Meghana@DESKTOP-FBKGRS MINGW64 ~/Desktop/AWS (master)  
$ git config --global user.name meghana-choudhary  
Meghana@DESKTOP-FBKGRS MINGW64 ~/Desktop/AWS (master)  
$ git commit -m "Done"  
[master (root-commit) 3d1dc75] Done  
 3 files changed, 40 insertions(+)  
  create mode 100644 index.html  
  create mode 100644 index1.html  
  create mode 100644 index2.html  
Meghana@DESKTOP-FBKGRS MINGW64 ~/Desktop/AWS (master)  
$ git remote add origin https://github.com/meghana-choudhary/MyAWS.git  
Meghana@DESKTOP-FBKGRS MINGW64 ~/Desktop/AWS (master)  
$ git push -u origin master  
Enumerating objects: 5, done.  
Counting objects: 100% (5/5), done.  
Delta compression using up to 4 threads  
Compressing objects: 100% (5/5), done.  
Writing objects: 100% (5/5) 583 bytes | 582.00 KiB/s, done.  
Total 5 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)  
remote: Resolving deltas: 100% (1/1) done.  
To https://github.com/meghana-choudhary/MyAWS.git  
 * [new branch] master -> master  
branch 'master' set up to track 'origin/master'.  
Meghana@DESKTOP-FBKGRS MINGW64 ~/Desktop/AWS (master)  
$ |
```

9. After the last command, a new “**Connect to GitHub**” window should open. Go to “**Token**” tab and paste the previously generated token into the text field. Click on “**Sign in**” button.



Don't have an account? [Sign up](#)

10. Open the repository in **Github**. Our project files should now be visible. Now click on “**Code**” button and copy the HTTPS URL.

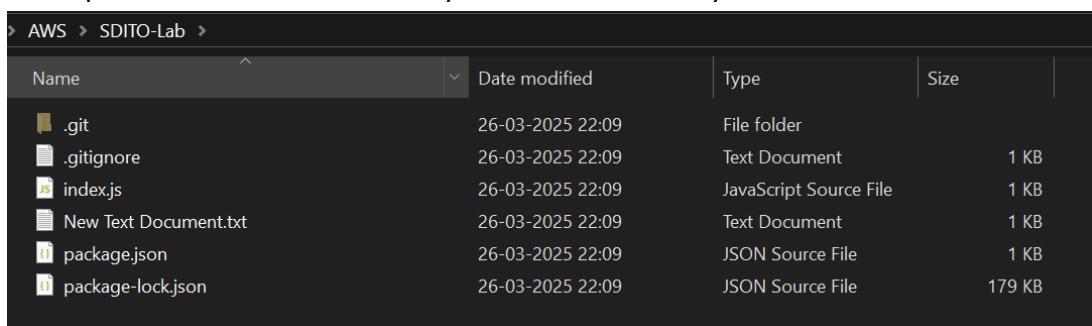


The screenshot shows a GitHub repository page for 'SDITO-Lab'. The 'Code' button is highlighted in red. A dropdown menu is open, showing options for 'Local', 'Codespaces', and 'Copilot'. Under 'Clone', the 'HTTPS' option is selected, displaying the URL 'https://github.com/SujalGupta52/SDITO-Lab.git'. Other options include 'SSH' and 'GitHub CLI'. Below the URL, there are links for 'Clone using the web URL.', 'Open with GitHub Desktop', 'Open with Visual Studio', and 'Download ZIP'. The repository has 1 branch ('master') and 0 tags. The 'About' section indicates no description, website, or topics provided. It shows 1 watching and 0 forks. There are sections for 'Releases' (no releases published) and 'Packages'.

11. Now make a new folder and open the newly created folder in git bash terminal. Run the following command to clone our repo:
`git clone <Repository address>`

```
PS C:\Users\sujal\OneDrive\Desktop\AWS> git clone git@github.com:SujalGupta52/SDITO-Lab.git
Cloning into 'SDITO-Lab'...
remote: Enumerating objects: 7, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 7 (delta 0), reused 7 (delta 0), pack-reused 0 (from 0)
Receiving objects: 100% (7/7), 48.22 KiB | 137.00 KiB/s, done.
```

12. Our repo files is successfully cloned locally.



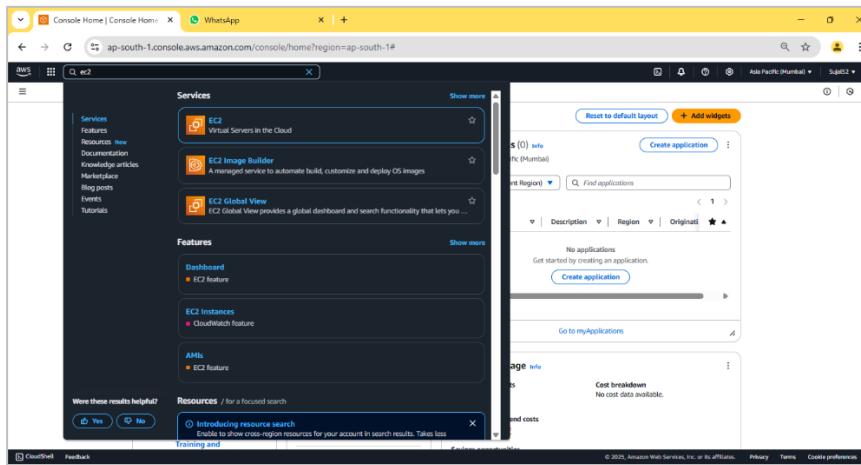
Name	Date modified	Type	Size
.git	26-03-2025 22:09	File folder	
.gitignore	26-03-2025 22:09	Text Document	1 KB
index.js	26-03-2025 22:09	JavaScript Source File	1 KB
New Text Document.txt	26-03-2025 22:09	Text Document	1 KB
package.json	26-03-2025 22:09	JSON Source File	1 KB
package-lock.json	26-03-2025 22:09	JSON Source File	179 KB

ASSIGNMENT 9:

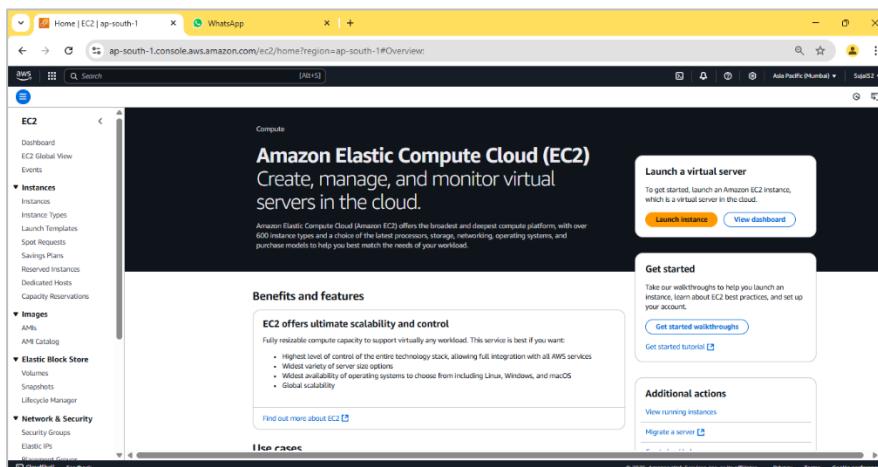
Problem Definition: Deploy a project from GitHub to EC2.

Instructions:

1. Access the **AWS console**, search for **EC2**, and select the top option from the search results.



2. Click on “**Launch instance**”.



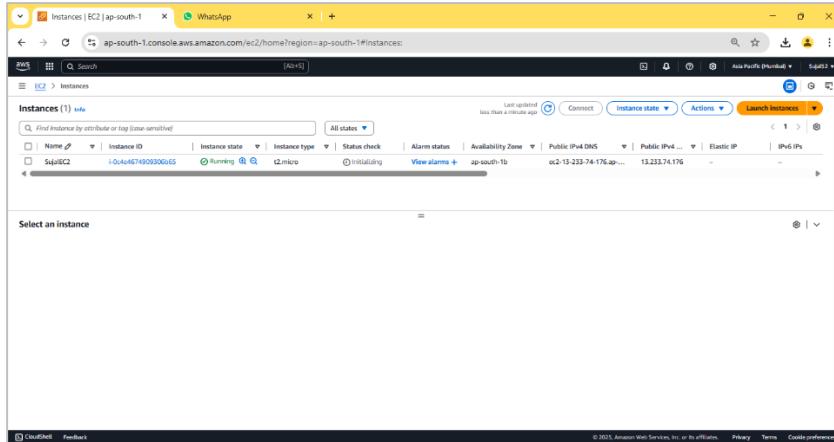
3. Enter instance name and select **Ubuntu** under “**Application and OS Images**”. Under “**Key Pair (login)**” click on “**Create new key pair**”, enter key pair name then click on “**Create key pair**” button, a *.pem file will be downloaded. Then click on “**Launch instance**” button

The screenshot shows the AWS EC2 'Launch an instance' wizard. In the 'Application and OS Images (Amazon Machine Image)' section, 'Ubuntu' is selected from the list. A tooltip for the 'Create key pair' step states: 'Free tier: In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free tier AMIs. This includes up to 100 hours per month of t2.micro usage, 30 GB of EBS storage, 2 million I/Os, 1 GiB of snapshots, and 100 GiB of bandwidth to the internet.' Below this, the 'Launch instance' button is visible.

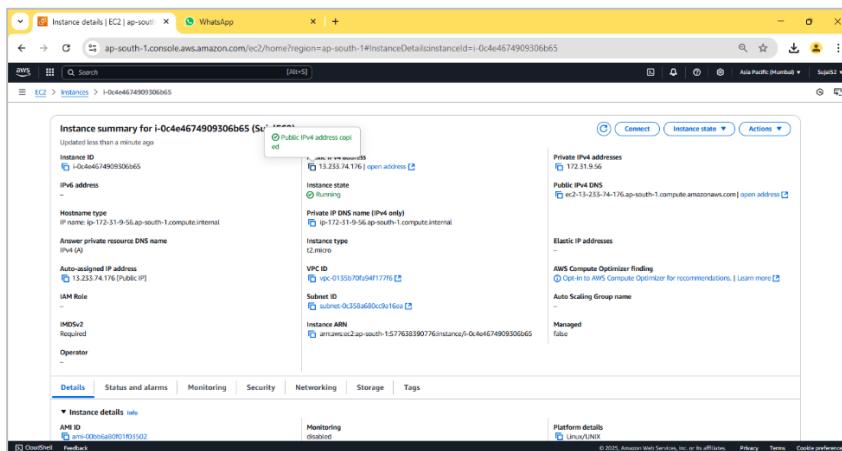
4. Our instance is created successfully.

The screenshot shows the AWS EC2 'Launch an instance' wizard. A green success message at the top reads: 'Success Successfully initiated launch of instance i-0c4a674099306565'. Below this, a 'Next Steps' section lists several options: 'Create billing and free tier usage alerts', 'Connect to your instance', 'Connect an RDS database', 'Create EBS snapshot policy', 'Manage detailed monitoring', 'Create Load Balancer', 'Create AWS budget', and 'Manage CloudWatch alarms'. Each option has a corresponding 'Learn more' link.

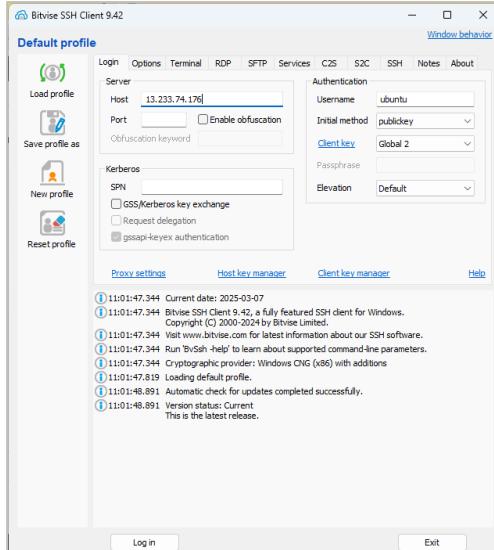
5. Go to “**Instance**” from the sidebar, then click on **Instance ID** of our newly created instance.



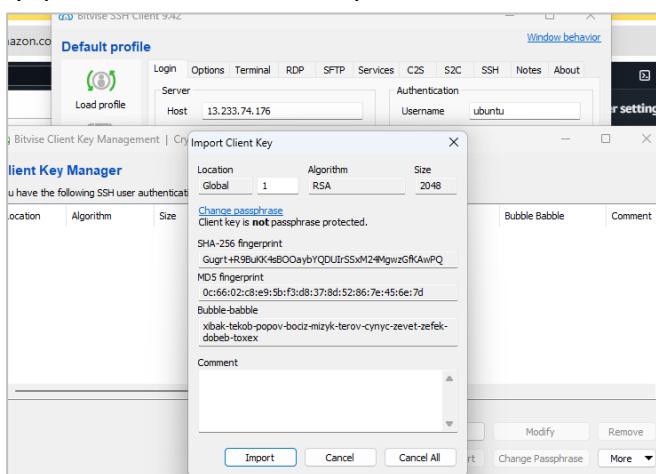
6. Copy “**Public IPV4 address**”.



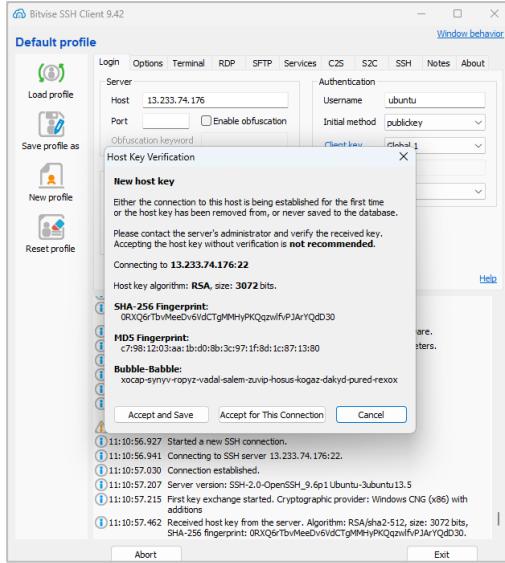
7. Open **Bitvise SSH Client**, paste the previously copied address in host field.



8. Click “**Client key manager**”, click on “**Import**” and select our previously downloaded key pair file. Click on “**Import**” button. Our key pair should be imported successfully.



9. In “**Bitvise SSH Client**”, under “**Authentication**” give the username as “**ubuntu**”, select “**Global 1**” under “**Client Key**” then click on “**Log in**” & “**Accept and save**”.



10. Click on “**New terminal console**” from the sidebar. In the terminal enter the following commands to clone our repo and run our Node server:

```
sudo apt-get update
```

```
sudo apt-get upgrade
```

```
sudo apt-get install nginx
```

```
curl -sL https://deb.nodesource.com/setup_18.x | sudo -E bash -
```

```
sudo apt install nodejs
```

```
git clone <Github repo link>
```

```
cd <Github repo name>
```

```
npm install
```

```
node index.js
```

```

ubuntu@ip-172-31-13-131:~$ ls
Aritra_01
ubuntu@ip-172-31-13-131:~$ cd Aritra_01
ubuntu@ip-172-31-13-131:~/Aritra_01$ ls
index.js  package-lock.json  package.json
ubuntu@ip-172-31-13-131:~/Aritra_01$ npm install
npm warn deprecated uid@3.4.0: Please upgrade to version 7 or higher. Older versions may use Math.random() in certain circumstances, which is known to be problematic. See https://v8.dev/blog/math-random for details.

added 258 packages, and audited 259 packages in 8s

18 packages are looking for funding
  run `npm fund` for details

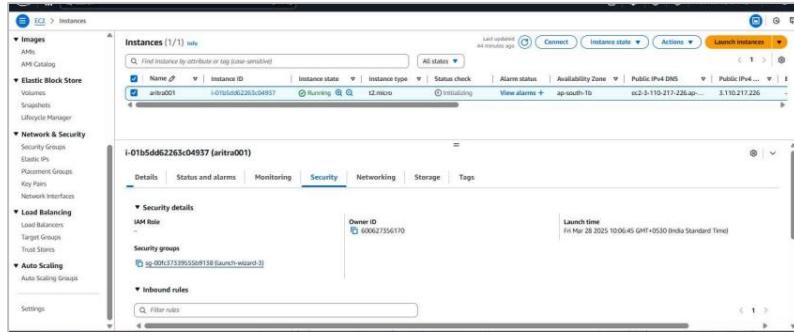
23 vulnerabilities (3 low, 2 moderate, 16 high, 2 critical)

To address all issues, run:
  npm audit fix

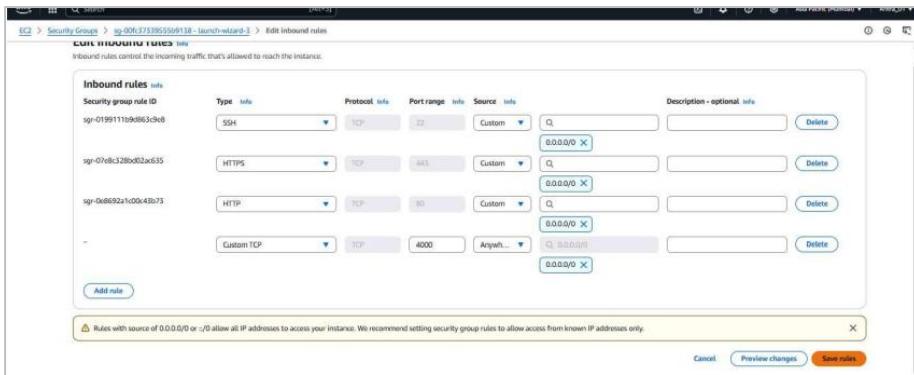
Run `npm audit` for details.
npm notice
npm notice New major version of npm available! 10.8.2 -> 11.2.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v11.2.0
npm notice To update run: npm install -g npm@11.2.0
npm notice
ubuntu@ip-172-31-13-131:~/Aritra_01$ npm -v
10.8.2
ubuntu@ip-172-31-13-131:~/Aritra_01$ node index.js
Started server

```

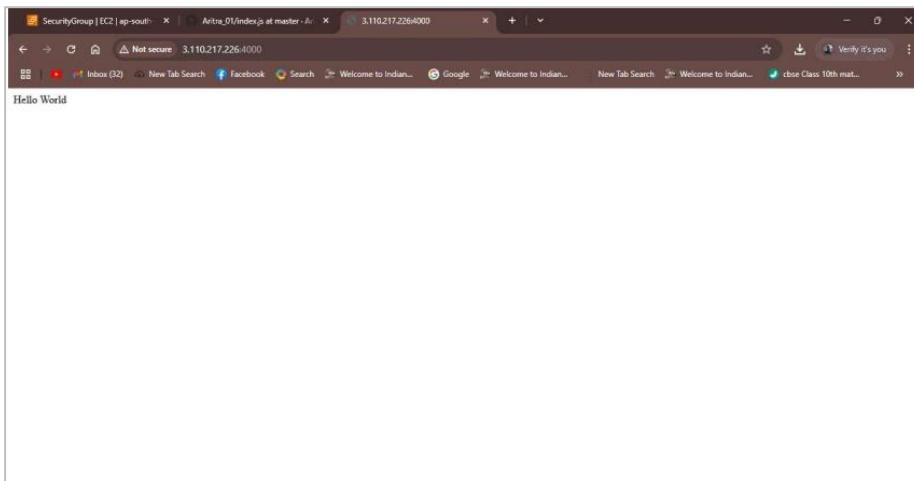
11. Go to “**Instance Id**”, and click on our instance. Then go to ‘Security’ and click on link under ‘Security groups’ to get the edit option.



12. Then in “**Inbound rules**”, click on “**Edit inbound rules**”. Click on “**Add rule**” and set the “**Port range**” to “**4000**”, in “**Source**” set “**0.0.0.0/0**” & click on “**Save rules**”. A confirmation is shown stating our rules are saved.



13. Now copy the “**Public IPv4 address**” & paste it on a new tab. Add “**:4000**” at the end and then press enter. The project has been successfully deployed from GitHub to EC2.

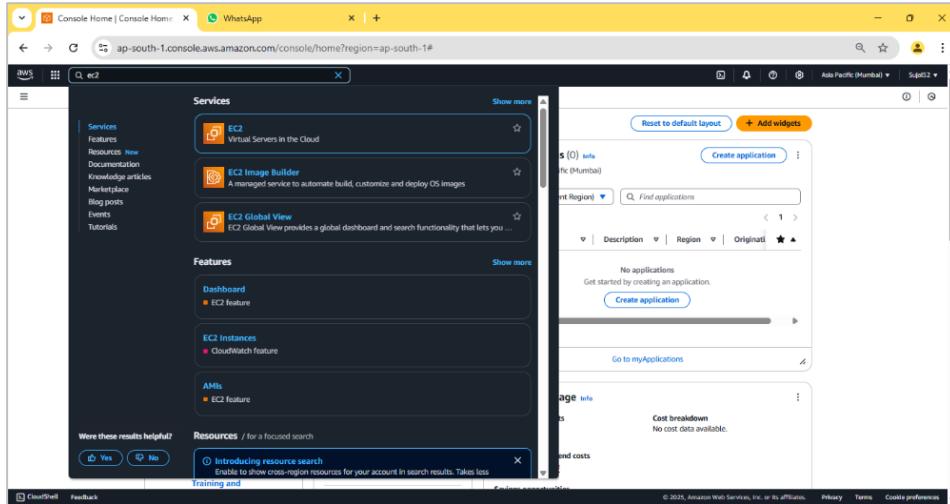


ASSIGNMENT 10:

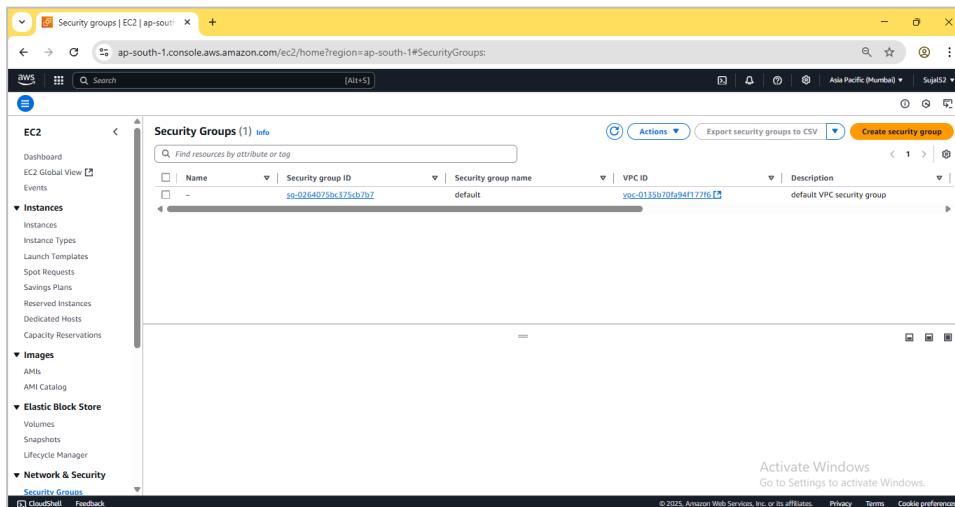
Problem Definition: Deploy a project from GitHub to EC2 by creating a new security group and user data.

Instructions:

1. Access the **AWS console**, search for **EC2**, and select the top option from the search results.



2. Select **Security Groups** from the side bar and click on “**Create security group**”.



3. Add name and description. Under “**Inbound Rules**”, add the “**Add rule**” to add the following rules: SSH, HTTP, HTTPS and Custom TCP with port 4000. Select 0.0.0.0/0 under source for all rules. Finally click

on “**Create security group**”. Pop up is shown stating our rules are created successfully.

The screenshots illustrate the steps to create a security group:

- Step 1: Basic Details**
The first screenshot shows the "Create security group" page. Under "Basic details", the "Security group name" is set to "MySecurityGroup". A note states: "A security group acts as a virtual firewall for your instance to control inbound and outbound traffic. To create a new security group, complete the fields below." The "Description" field contains "Security Group". The "VPC" dropdown is set to "vpc-0135b70fa94f177f6".
- Step 2: Inbound Rules**
The second screenshot shows the "Inbound rules" section. It lists four default rules: "Custom TCP" (port 4000), "SSH" (port 22), "HTTP" (port 80), and "HTTPS" (port 443). Each rule has a "Delete" button. A note at the bottom says: "Rules with source of 0.0.0.0/0 or ::/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only."
- Step 3: Outbound Rules**
The third screenshot shows the "Outbound rules" section. It lists one rule: "All traffic" (Protocol: All, Port range: All, Destination: Custom, Destination IP: 0.0.0.0/0). A note at the bottom says: "Rules with destination of 0.0.0.0/0 or ::/0 allow your instances to send traffic to any IPv4 or IPv6 address. We recommend setting security group rules to be more restrictive and to only allow traffic to specific known IP addresses."

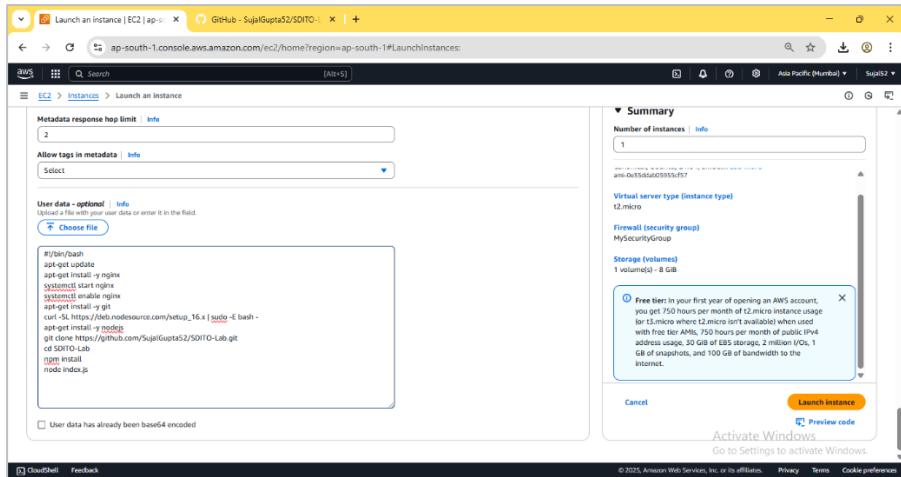
4. Go to **Instances** from the sidebar and click on “**Launch Instance**”. Enter instance name, select Ubuntu as operating system, select an existing key pair or create a new one. Under “**Network**

settings", click on "**Select existing security group**" and select our previously created security group.

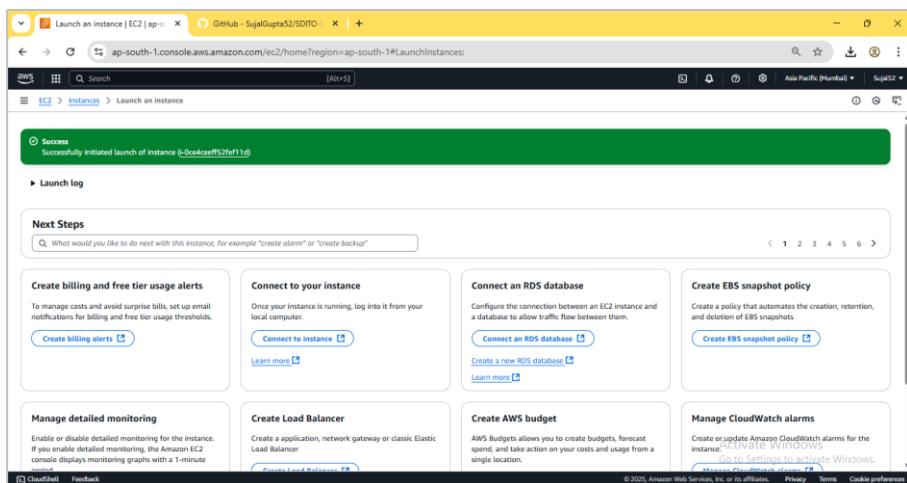
The image consists of three vertically stacked screenshots of the AWS EC2 "Launch an instance" wizard. Each screenshot shows a different step in the process:

- Screenshot 1 (Top):** Shows the initial "Launch an instance" screen where the user selects the instance type (t2.micro), operating system (Ubuntu), and storage (1 volume(s) - 8 GB). A tooltip for the "Free tier" is visible, stating: "In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free-tier AMIs, 750 hours per month of public IPv4 address usage, 30 GB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet."
- Screenshot 2 (Middle):** Shows the "Create key pair" step. The user has chosen "RSA" and named it "mykey123". A tooltip for the "Free tier" is visible, stating: "In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free-tier AMIs, 750 hours per month of public IPv4 address usage, 30 GB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet."
- Screenshot 3 (Bottom):** Shows the "Network settings" step. Under "Firewall (security groups)", the user selects "Select existing security group" and chooses "MySecurityGroup". A tooltip for the "Free tier" is visible, stating: "In your first year of opening an AWS account, you get 750 hours per month of t2.micro instance usage (or t3.micro where t2.micro isn't available) when used with free-tier AMIs, 750 hours per month of public IPv4 address usage, 30 GB of EBS storage, 2 million I/Os, 1 GB of snapshots, and 100 GB of bandwidth to the internet."

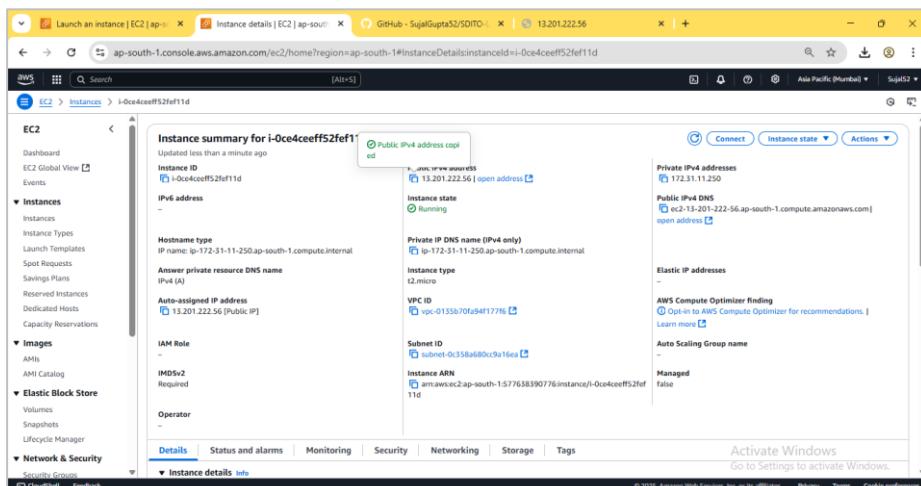
5. Expand "**Advanced details**", scroll down to "**User data**" and put the following bash script. Then click on "**Launch Instance**".



6. Our instance is created successfully

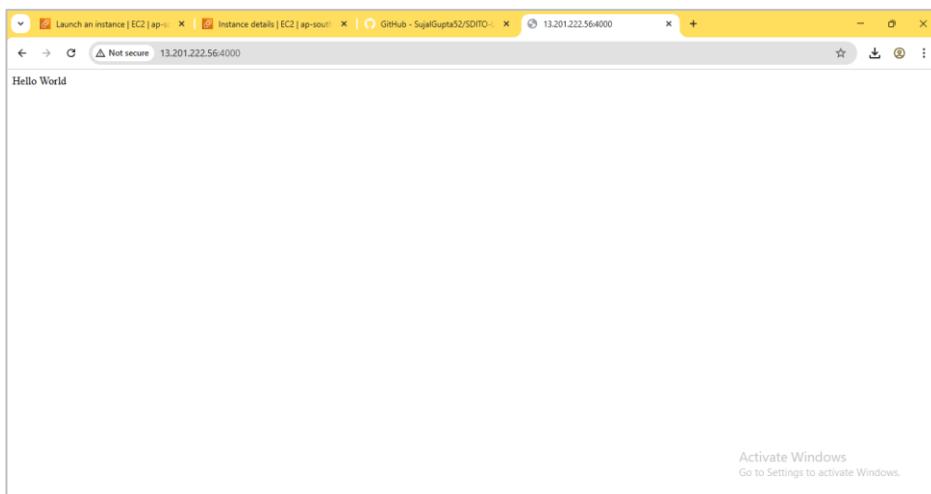


7. Go to **Instance**, click on “**Instance ID**” of our newly created instance. Copy the **IPV4 address**.



8. Paste the previously copied address into a new browser tab and suffix “:4000” to indicate our port. Our NodeJS server is successfully

deployed.

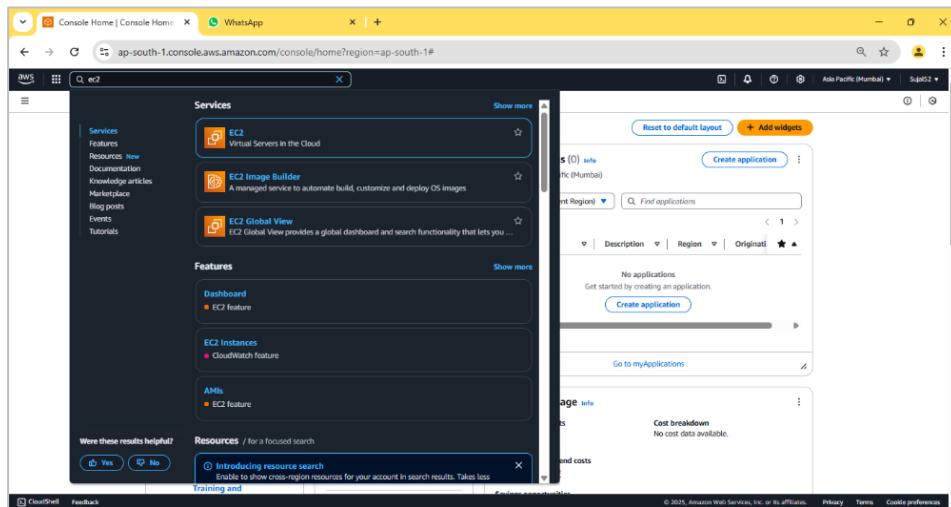


ASSIGNMENT 11:

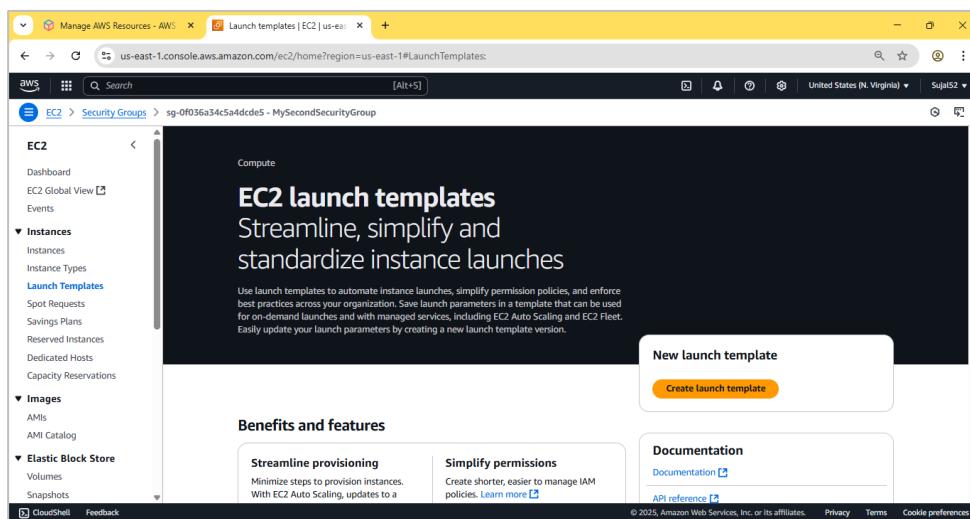
Problem Definition: Build scaling plans in AWS that balance load on different EC2 instances.

Instructions:

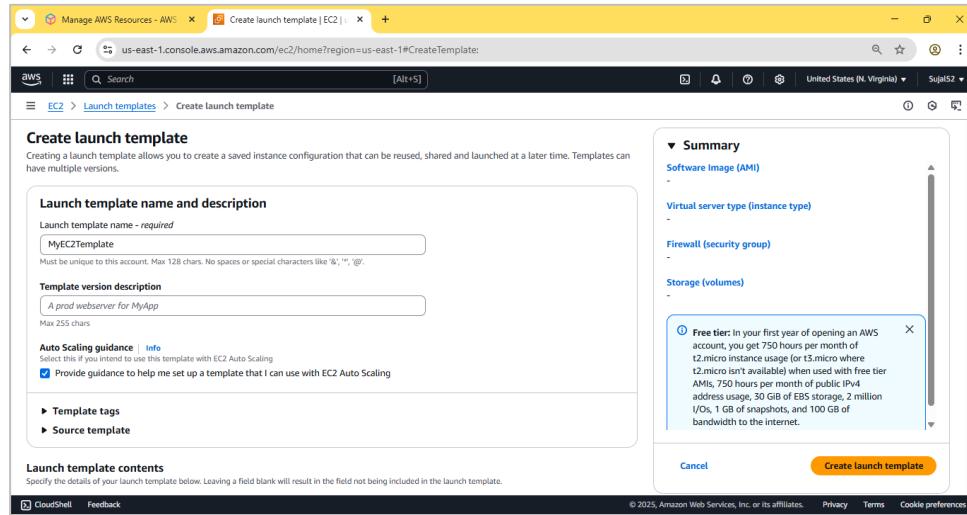
1. Access the **AWS console**, search for **EC2**, and select the top option from the search results.



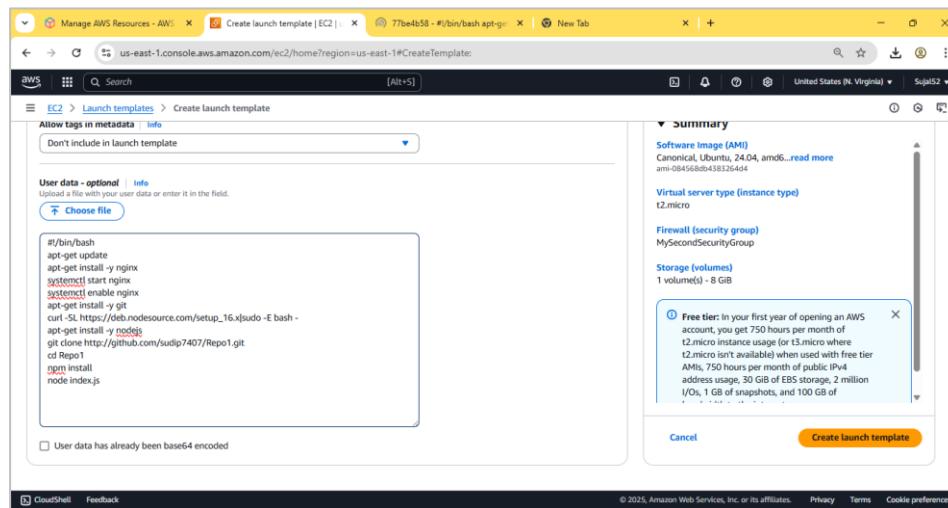
2. Click on “**Launch Templates**” from the sidebar then click on “**Create launch template**”.



3. Enter name and description. Check the box under “**Auto Scaling guidance**”. Select **Ubuntu** as OS Image, instance type as **t2.micro**. Create keypair and select an existing security group where port 4000 is open for incoming traffic.



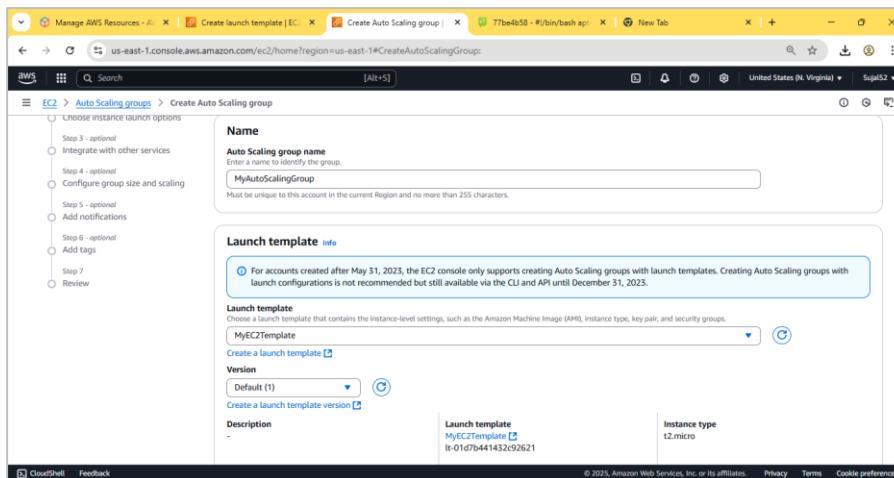
Under “**Advance details**”, scroll down to “**User data**” and add the following bash script and then click on create instance



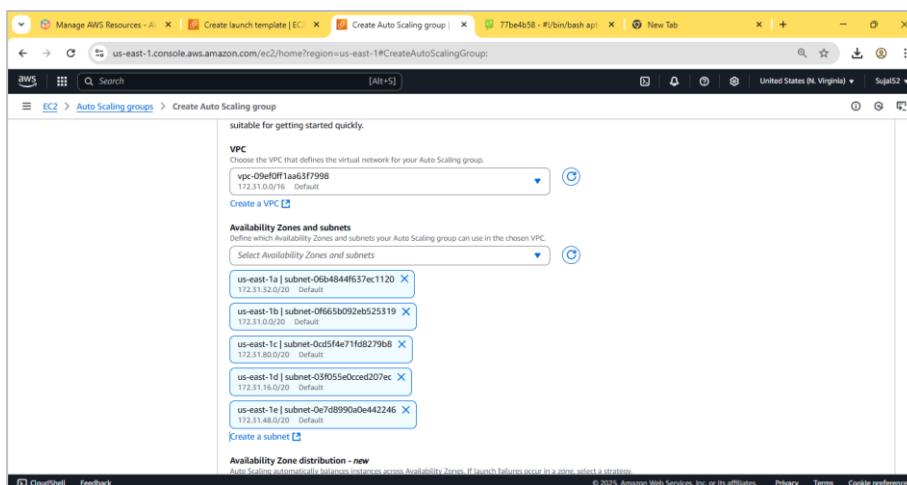
Our launch template is created successfully.



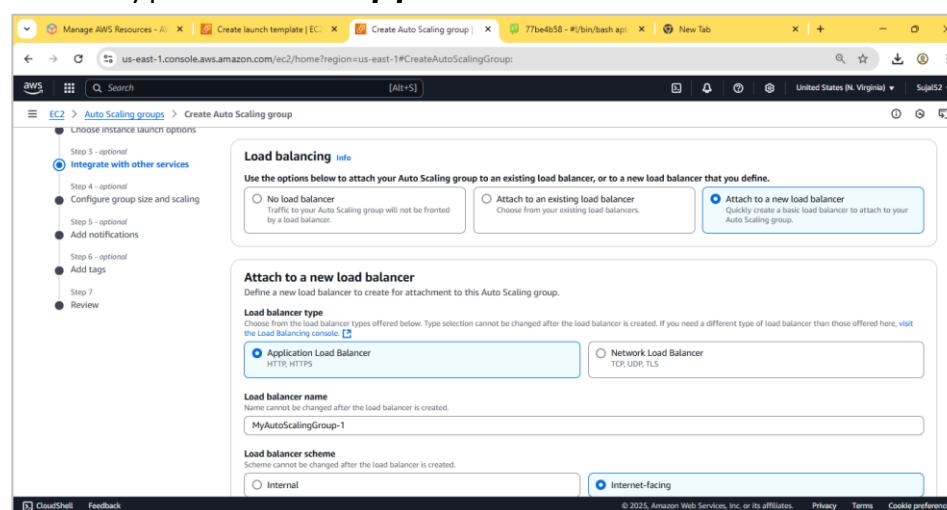
4. From sidebar, scroll down and select “**Auto Scaling Groups**”. Enter name and select our previously created launch template from the dropdown. Then click on next.



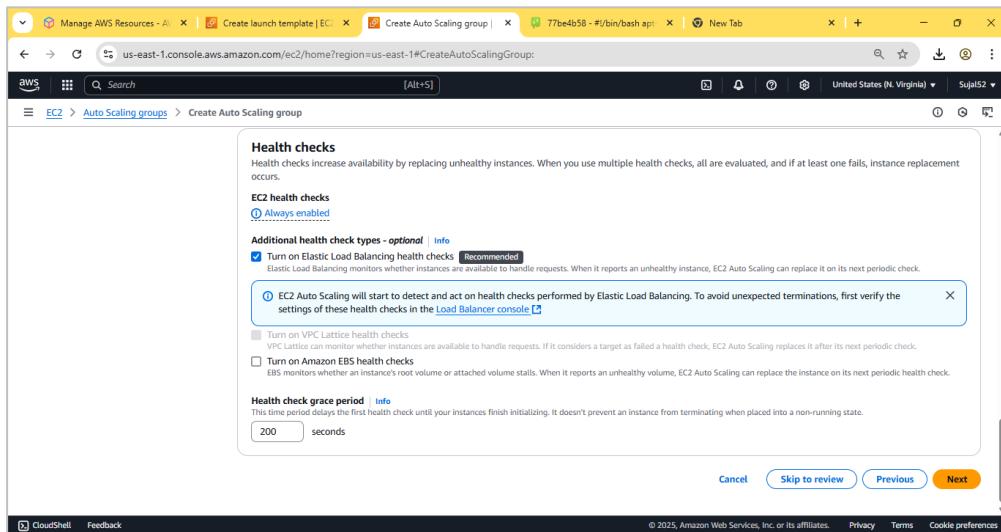
Choose all “**Availability Zones**” available then click on next.



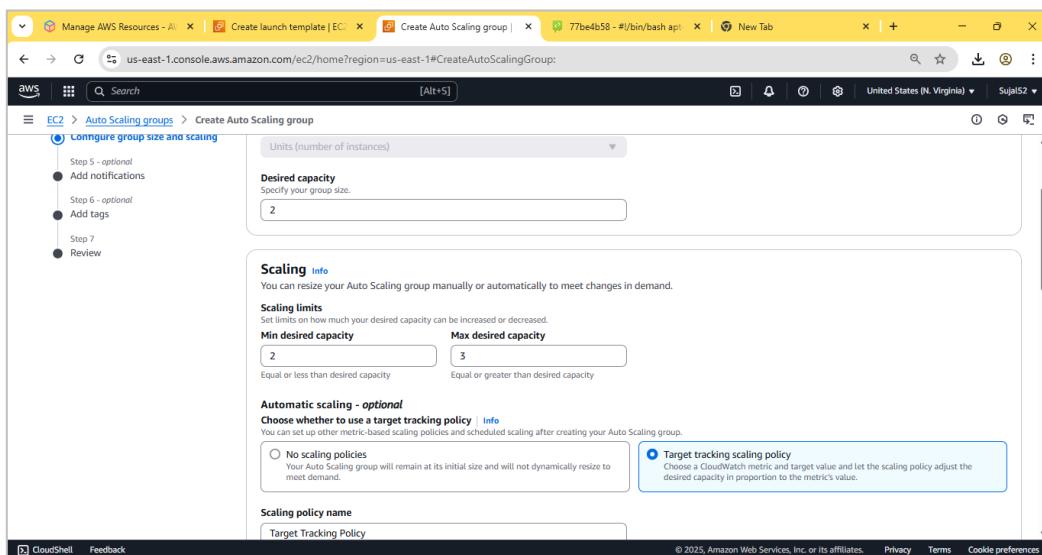
Under “**Load balancing**”, select “**Attach a new load balancer**”, under type select “**Application load balancer**” and “**Internet-facing**”



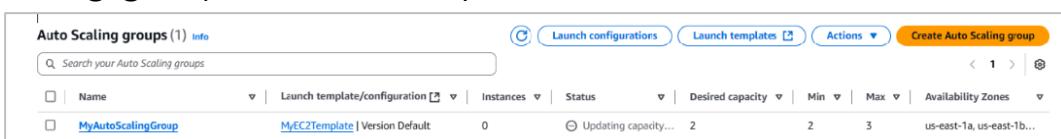
Under “**Health checks**” tick the box on “**Turn on Elastic Load Balancing Health Check**” and enter the “**Health check grace period**” as 240 seconds. Click on next.



In **Group size** set **Desired capacity** as 2. In **Scaling** set **Min desired capacity** as 2 and **Max desired capacity** as 3. In **Automatic scaling** choose **target scaling policy** and set the **instance warmup** as 240 second. Keeping everything else as default, click on Next.



Skip last two steps and click on **Create Auto Scaling group**. Our auto scaling group is successfully created.



5. Now go to **Instances** from the sidebar. We can see two instances are created. Copy the **IPv4 address** of any of the two instance and log into **Bitwise SSH Client** using key pair.

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IP
	i-05a8ca1c17e0d6bed	Running	t2.micro	Initializing	View alarms	us-east-1a	ec2-54-8
	i-08f2e75400ee1b3da	Running	t2.micro	Initializing	View alarms	us-east-1b	ec2-3-23

Instance ID	Public IPv4 address	Private IPv4 addresses
i-05a8ca1c17e0d6bed	54.89.81.166	172.31.35.150

6. Open terminal console and enter the following commands:

```
vim file.sh
```

Now paste the following code:

```
#!/bin/bash

while(true)

do

echo "Inside Loop"
```

done

Now make the script executable and run it enter the following commands:

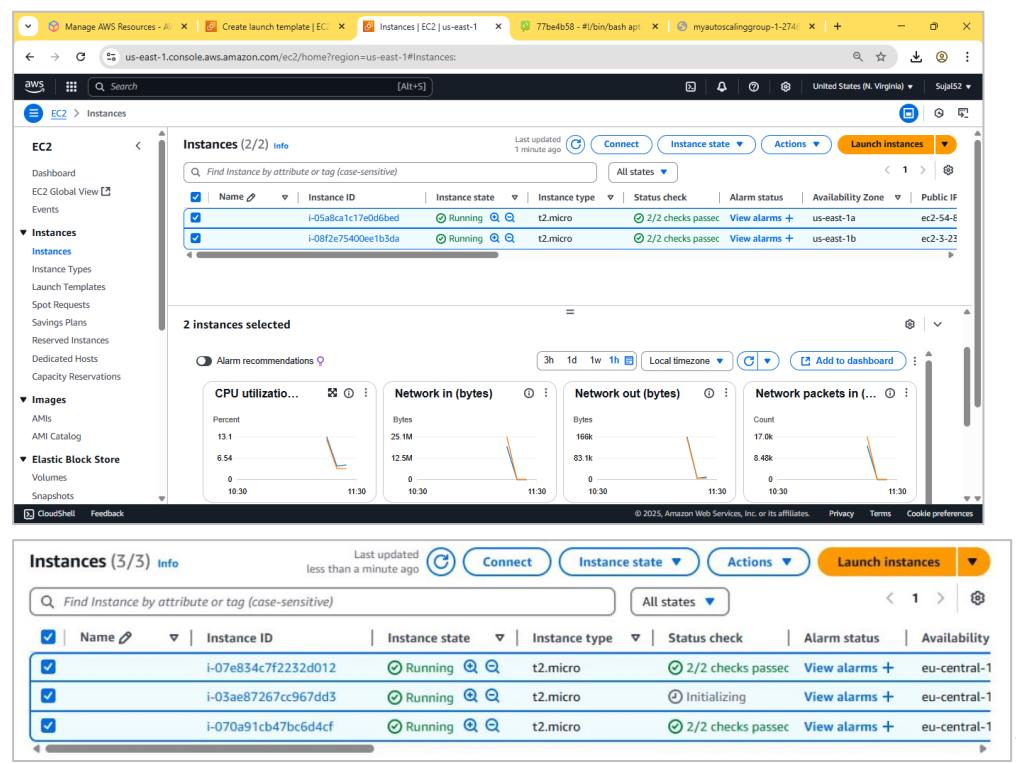
```
sudo chmod 777 file.sh
```

sh file.sh

```
|ubuntu@ip-172-31-35-150:~$ sudo nano file.sh  
ubuntu@ip-172-31-35-150:~$ sudo chmod +x file.sh  
ubuntu@ip-172-31-35-150:~$
```

An infinite loop will start to run, stressing the cpu of the instance.

7. Select the two instances to see their utilization graph. After a while, when it surpasses 50% threshold, we can see that auto scaler spins up another instance to balance the load.

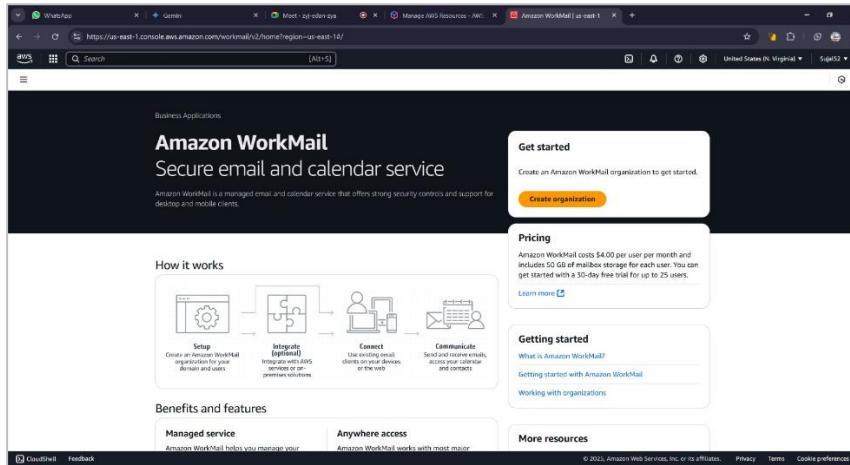


ASSIGNMENT 13:

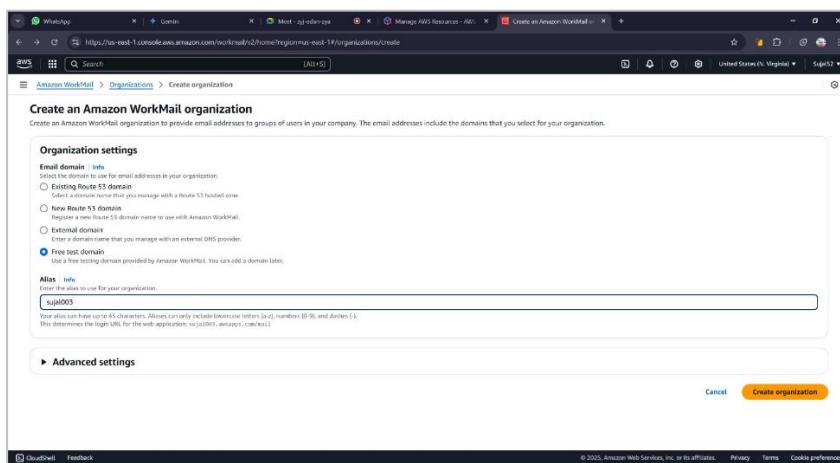
Problem Definition: Create a workmail for your organization.

Instructions:

1. Access the **AWS console**, search for **Workmail**, and select the top option from the search results. Click on **Create Organization**.



2. Select **Free Test Domain** under **Organization settings**. Enter an alias then click on **Create Organization**.



Our organization is created successfully.

The screenshot shows the 'Organizations' page in the Amazon WorkMail console. A green banner at the top states 'Your organization is now ready to use' with a link to 'Go to the Organization details page for next steps'. Below the banner, there is a table with one row for 'sujit003'. The table includes columns for 'Organization ID' (m-474409114d44c0b6fd23d45c7d9d8d8), 'Default domain' (sujit003.awsapps.com), and 'State' (Active). There are buttons for 'Delete', 'Create organizations', and a search bar.

3. Click on Organisation name. Go to **Users** and click on **Add User** button.

The screenshot shows the 'Users' page for the organization 'sujit003'. The left sidebar shows navigation options like 'Organizations', 'Users' (which is selected), 'Groups', 'Resources', and 'Identity Center'. The main area displays a table with no users listed, with columns for 'Username', 'Display name', 'Primary email address', and 'State'. Buttons for 'Delete', 'Disable', 'Enable', 'Reset password', and 'Add user' are visible at the top right.

4. Fill username, display name and password. Then click on **Add User**.
A green banner will be displayed indicating our user is created successfully.

The screenshot shows the 'Add a user' form. In the 'Email address' section, 'ad2003' is entered in the 'Primary email address' field, and '@topg.awsapps.com' is selected from the dropdown. The 'Show in global address list' checkbox is checked. In the 'Password setup' section, both 'Password' and 'Repeat password' fields are filled with 'Topg@123'. At the bottom, there are 'Cancel' and 'Add user' buttons.

5. Go to Organisation Details page again and click on the **Amazon WorkMail web application**.

The screenshot shows the 'Organization details' section of the Amazon WorkMail console. It displays organization information such as Organization ID (m-4744fb91b16b84ac0f60423465z79d88), State (Active), ARN (arn:aws:workmail:us-west-2:577638290796:organization/m-4744fb91b16b84ac0f60423465z79d88), Date created (April 19, 2025 at 11:27 (UTC+5:30)), Directory ID (d-92678bd425), and Default domain (sujal003.awsapps.com). Below this, there's a 'User setup guide' section with links for 'Amazon WorkMail web application' and 'WorkMail documentation for setting up email clients'. The left sidebar shows navigation options like Organizations, Users, Groups, Resources, Identity Center, Domains, Organization settings, Access control rules, Retention policies, Impersonation roles, Monitoring, and Logging settings.

6. Enter our previously created username and password. The inbox should open indicating our workmail is created successfully.

The screenshot shows the Amazon WorkMail login page. It features a logo with a green envelope icon and the text 'amazon WorkMail'. A message says 'Please log in with your sujal003 credentials.' Below it is a form with fields for 'Username (not email address)' containing 'SujalS2', a 'Remember username' checkbox (unchecked), and a 'Password' field with masked input. A 'Sign In' button is at the bottom. Below the form, small text reads: 'By continuing, you agree to the AWS Customer Agreement or other agreements for AWS services, and the Privacy Notice. This site uses essential cookies. See our Cookie Notice for more information.'

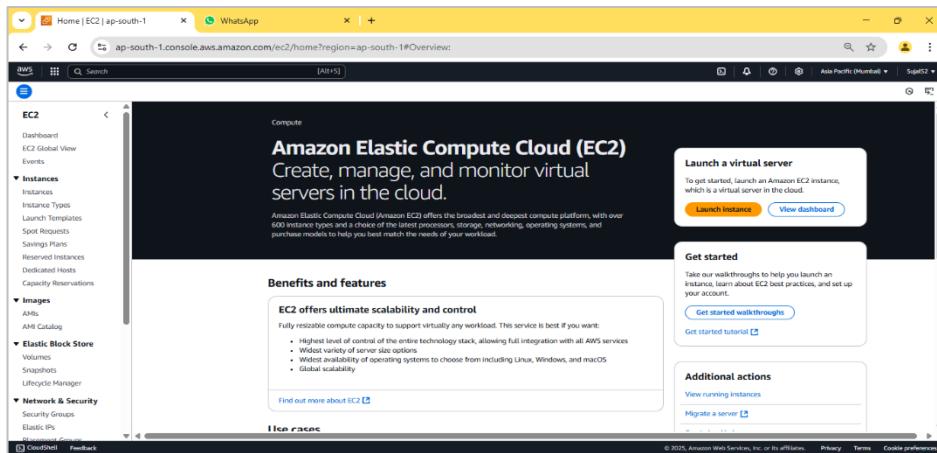
The screenshot shows the Amazon WorkMail inbox interface. The left sidebar has a dark theme with categories: 'My Mail' (Inbox, Junk Email, Outbox, Drafts, Sent Items, Deleted Items, RSS Feeds), 'Open other inbox', and a 'Search globally' bar. The main area is titled 'Inbox' and shows a message: 'There are no items to show in this view.' The top right shows the user name 'Sujal Gupta'.

ASSIGNMENT 14:

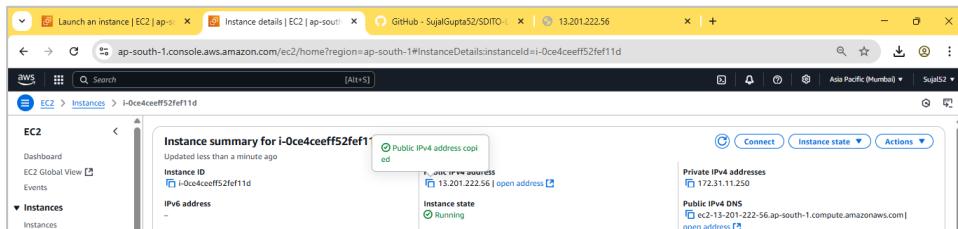
Problem Definition: Create an Elastic IP for an Instance.

Instructions:

1. Access the **AWS console**, search for **EC2**, and select the top option from the search results. Then go to **Instances** from the sidebar.



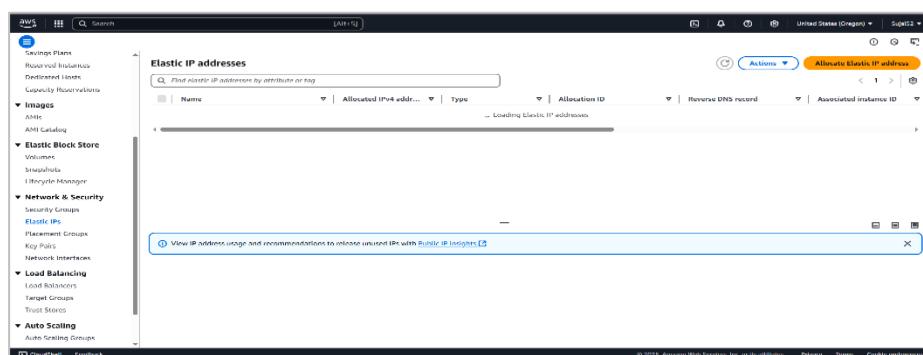
2. Launch a EC2 instance and copy its Public IPv4 address for future comparison.



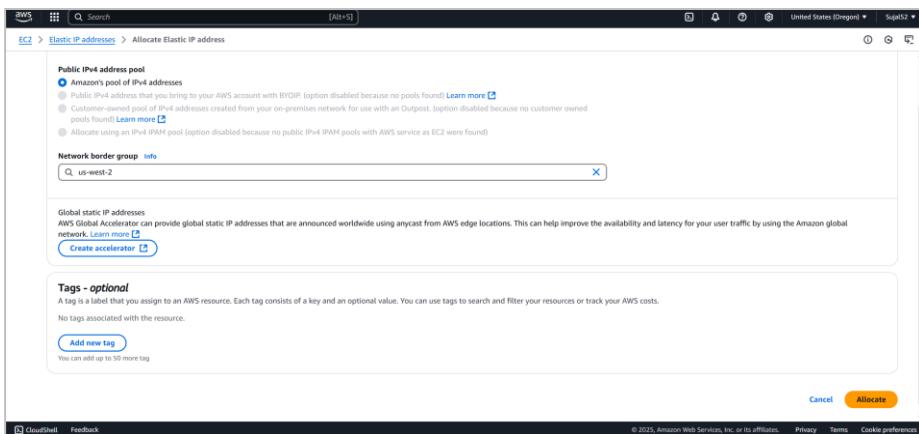
3. Now restart the instance and compare its Public IPv4 address with the one we previously saved. We can see it changes every time a instance is launched.



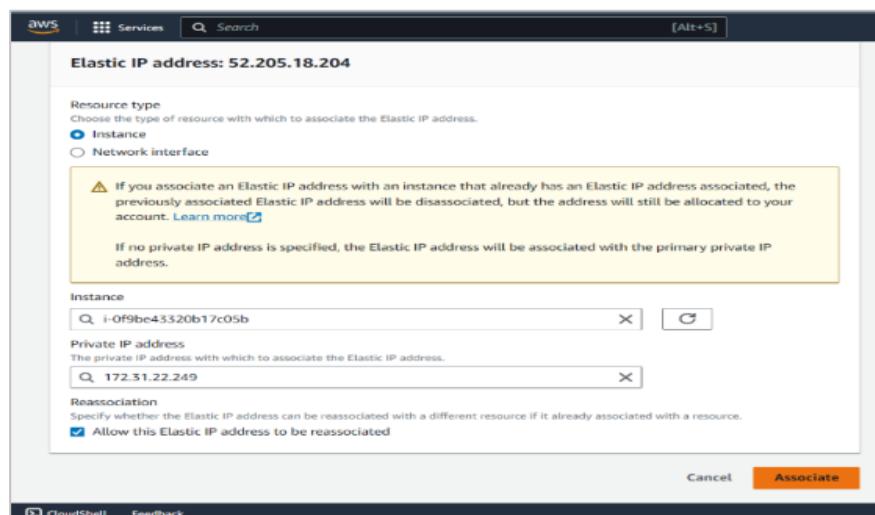
4. From the **EC2 Dashboard**, go to **Elastic IPs** and click on **Allocate Elastic IP**



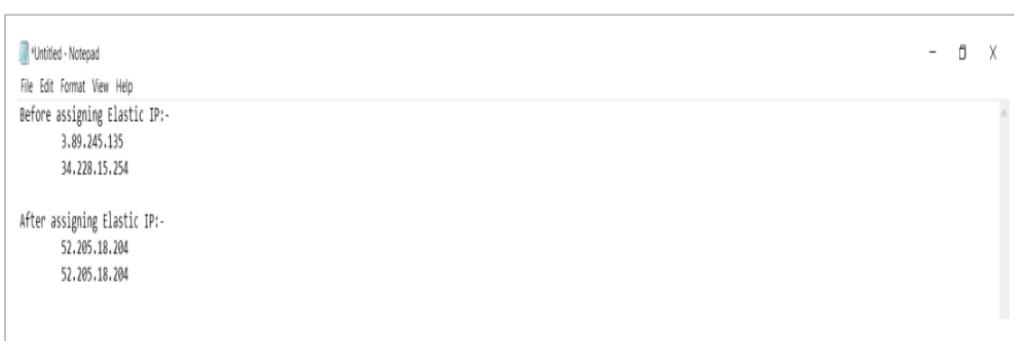
5. Select **Amazon's pool of IPv4 Addresses** and click on **Allocate**. A green banner will be shown indicating our Elastic IP has been created.



6. Click on **Associate Elastic IP address**. Select resource type as **Instance**, select name of our instance, its private IP address and then click on **Associate**.



7. Restart our instance again and we can see our Elastic IP is assigned and does not change on restart.

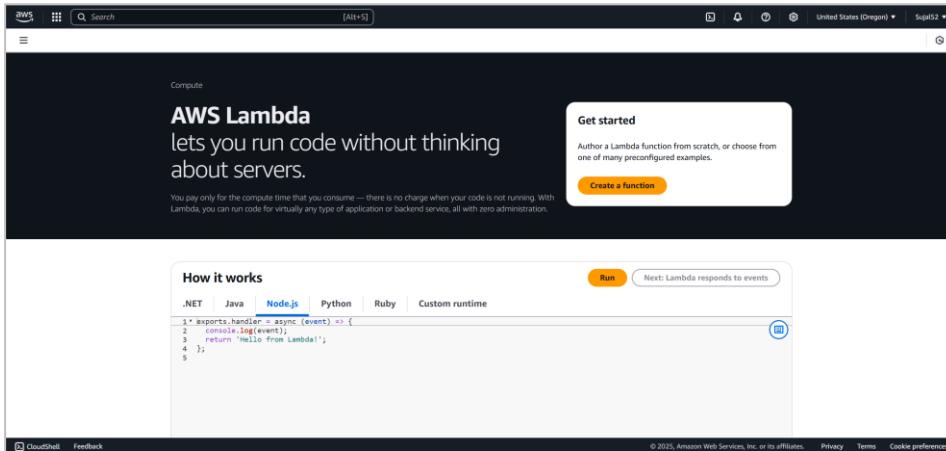


ASSIGNMENT 15:

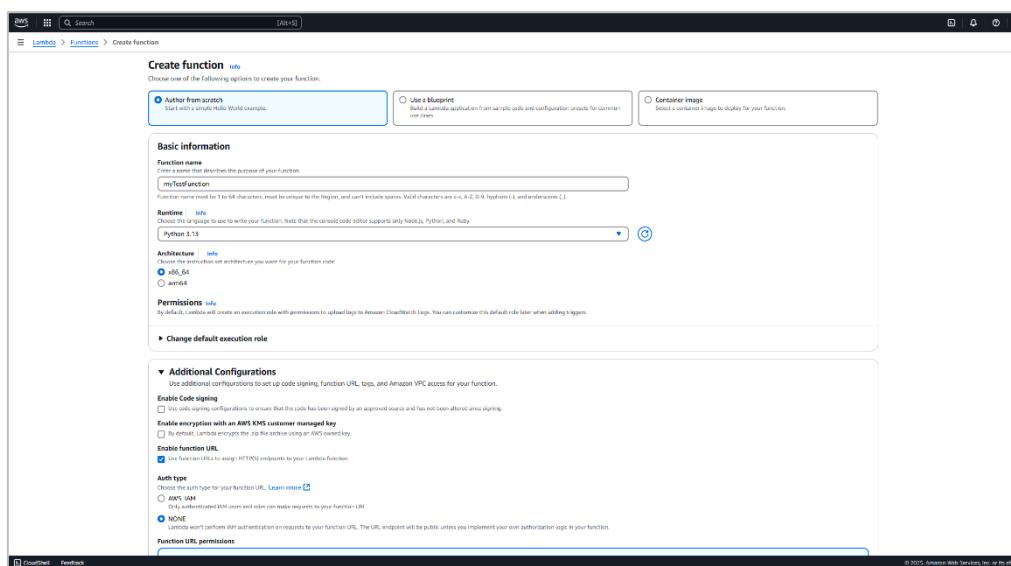
Problem Definition: Create a serverless computing service

Instructions:

1. Access the **AWS console**, search for **Lambda**, and select the top option from the search results.



2. Click on **Create a function**. Enter name and select **Python** as **Runtime**. Under **Advanced Configuration**, check **Enable function URL** and select **Auth type** as **None**. Click on **Create Function**.



3. Our function is created successfully. Now copy the function URL and open it in a new tab.

The screenshot shows the AWS Lambda Functions interface. A green success message at the top states: "Successfully created the function myTestFunction. You can now change its code and configuration. To invoke your function with a test event, choose 'Test'." Below this, the "myTestFunction" card is displayed, showing a diagram icon, the name "myTestFunction", and a "Layers (0)" section. A "Copy ARN" button is visible. The "Code" tab is selected in the navigation bar. Under the "Code source" section, there is a text input field containing the URL: "https://cy6i7ae6ownrhymquikawwmg0aheeo.lambda-url.us-west-2.on.aws". The browser's address bar also displays this URL. The "Function URL" section shows the full URL: "https://cy6i7ae6ownrhymquikawwmg0aheeo.lambda-url.us-west-2.on.aws/". On the right side, there are "Info" and "Tutorials" tabs, and a "Create a simple web app" tutorial is shown.