

ITIM

Name: Khan Mohammad Waseem
233414

Roll No.

Amazon Web Services (AWS):

AWS stands for Amazon Web Services. It is a comprehensive cloud computing platform provided by Amazon. AWS offers a wide range of cloud services, including computing power, storage, database management, networking, machine learning, artificial intelligence, analytics, and more.

Here are some key components and services offered by AWS:

1. **Elastic Compute Cloud (EC2):** It provides virtual servers in the cloud, allowing users to run applications and services.
2. **Simple Storage Service (S3):** It offers scalable object storage for data backup, archiving, and application hosting.
3. **Relational Database Service (RDS):** It provides managed database services for popular relational database engines such as MySQL, PostgreSQL, Oracle, and SQL Server.
4. **Lambda:** It is a serverless computing service that allows you to run code without provisioning or managing servers.
5. **Amazon DynamoDB:** It is a fully managed NoSQL database service that provides fast and predictable performance at any scale.
6. **Amazon SNS:** It is a fully managed messaging service that enables the distribution of messages to a large number of recipients.
7. **Amazon Redshift:** It is a fully managed data warehousing service that allows you to analyze large volumes of data using SQL queries.
8. **Amazon Glacier:** It is a low-cost storage service for data archiving and long-term backup.
9. **Amazon Elastic Beanstalk:** It is a platform as a service (PaaS) that allows developers to deploy and manage applications easily.
10. **Amazon Machine Learning:** It is a service that provides tools and wizards for creating machine learning models and generating predictions.

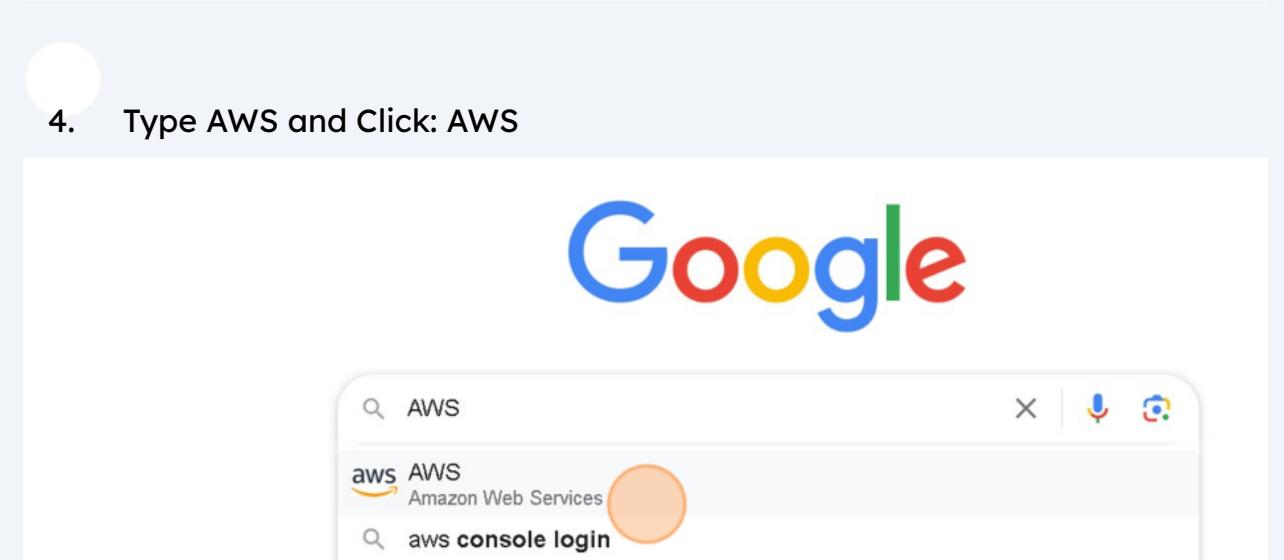
Create an AWS account.

1. Navigate to www.google.com

2. Click on the “Search” field.



4. Type AWS and Click: AWS



5. Click: "Amazon Web Services - AWS Official Site"

The screenshot shows a search results page with the following details:

- Header navigation links: News, Images, Login, Console, Certification, Tutorial, Careers, Educate, Academ.
- Search results count: About 66,30,00,000 results (0.33 seconds).
- Location filter: Results for Nigdi, Pimpri-Chinchwad, Maharashtra · Choose area.
- Sponsored result for Amazon.com:
 - Icon: Blue globe icon.
 - Text: Amazon.com
 - URL: <https://aws.amazon.com/account/sign-up>
 - Link: [Amazon Web Services - AWS Official Site](https://aws.amazon.com)
 - Description: Build, Deploy, and Manage Websites, Apps or Processes On **AWS** Secure, Reliable Network. Sign Up for a Free Account & Experience **AWS** Secure, Reliable, Scalable Services. Secure Solutions. Highly Scalable. Performance At Scale. No Upfront Commitment.
- Section: What is Cloud Computing?
 - Text: Benefits & Types of Cloud Computing Learn How AWS Can Help your Success
- Section: AWS Cloud Compute Service
 - Image: Small circular profile picture.
 - Text: AWS Lightsail: Easy-To-Use Platform For Simple Computing. 1 Year Free.

6. Click: "Create a free account"

The screenshot shows the AWS Free Tier landing page with the following content:

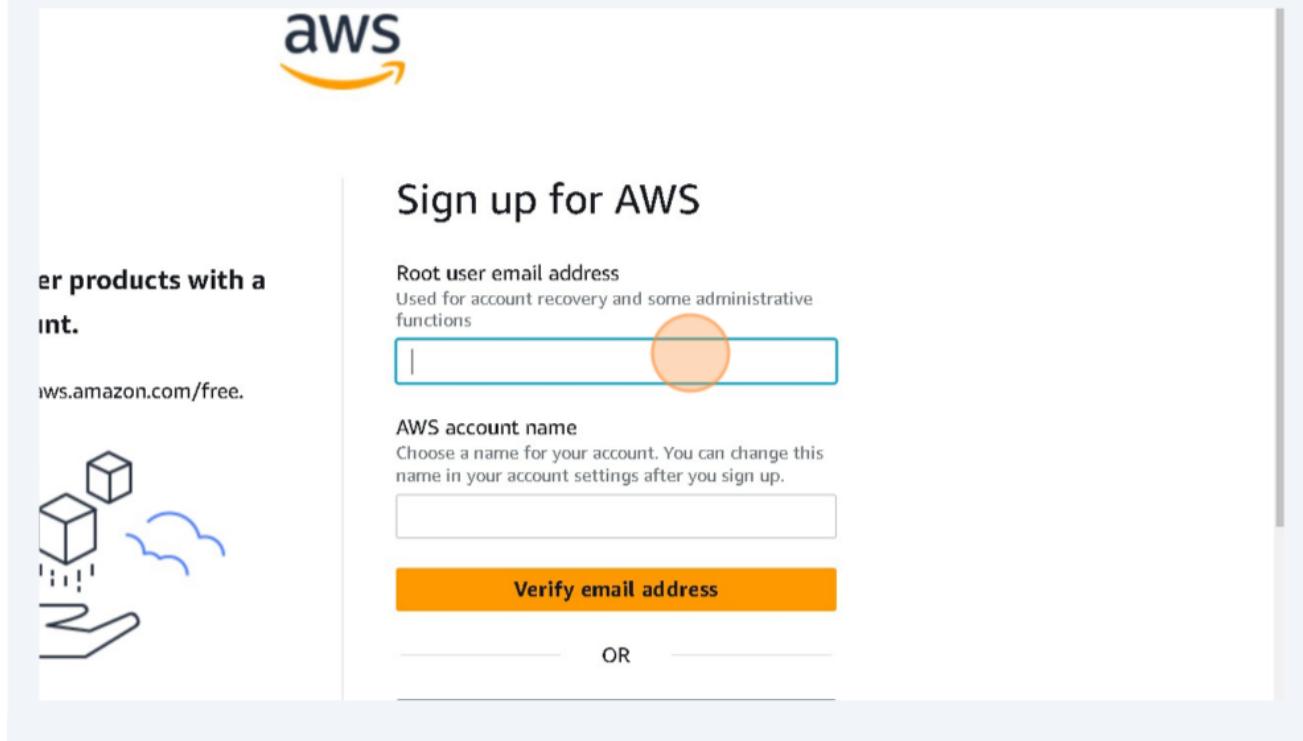
AWS Free Tier

hands-on experience with the AWS platform, products, and services

[Learn more about AWS Free Tier](#) ⓘ

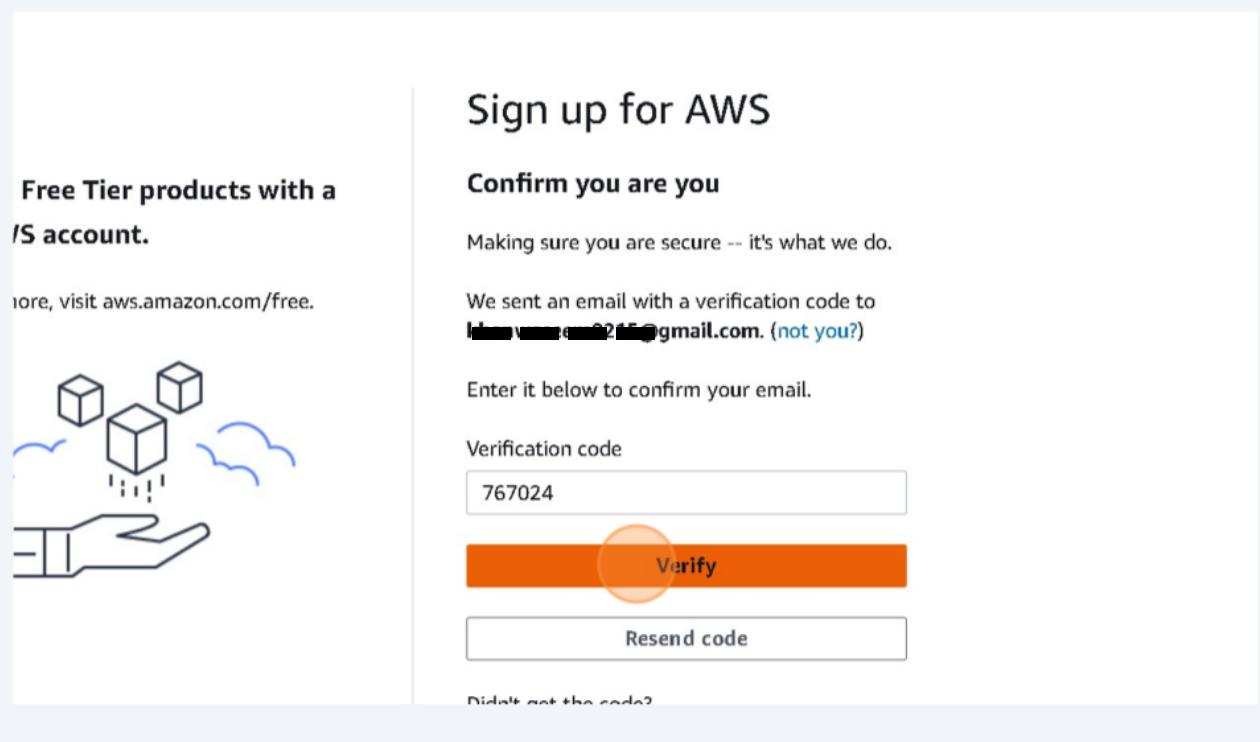
[Create a Free Account](#)

7.
 1. Click the "Root user email address" field and enter your: email address.
 2. Click the "AWS account name" field and enter AWS account name.
 3. Click "Verify email address"



The screenshot shows the AWS Sign Up page. At the top left is the AWS logo. Below it, the page title is "Sign up for AWS". On the left side, there is a sidebar with text about free tier products and a hand icon holding three cubes. The main form area has two input fields: "Root user email address" and "AWS account name". The "Root user email address" field is highlighted with a red circle. Below the fields is a large orange "Verify email address" button. A horizontal line with the word "OR" is centered below the button.

8.
 1. Click the "Verification code" field.
 2. Enter the verification code sent in your email.
 3. Click "Verify"



The screenshot shows the AWS Verify Email page. The title is "Sign up for AWS" and the sub-section is "Confirm you are you". It says "Making sure you are secure -- it's what we do." Below that, it states "We sent an email with a verification code to [REDACTED]@gmail.com. (not you?)". An instruction "Enter it below to confirm your email." is followed by a "Verification code" input field containing "767024", a red "Verify" button, and a "Resend code" button. At the bottom, there is a link "Didn't get the code?"

9. On this page: fill all the credentials.

The screenshot shows the AWS sign-up process. At the top right, the AWS logo is displayed. Below it, the heading "Sign up for AWS" is centered. To the left, there is a sidebar with the heading "Secure verification" and a note: "We will not charge you for usage below AWS Free Tier limits. We may temporarily hold up to \$1 USD (or an equivalent amount in local currency) as a pending transaction for 3-5 days to verify your identity." Below this note is a small decorative icon of a crown. The main form area is titled "Billing Information". It includes fields for "Credit or Debit card number" (with a placeholder box and icons for VISA, MasterCard, and American Express), "Expiration date" (with dropdown menus for Month and Year), and "Cardholder's name" (with a text input field). A small note at the bottom states: "AWS accepts all major credit and debit cards. To learn more about payment options, review our FAQ".

10. Click “Verify and continue (step 3/5)”

The screenshot shows the Amazon Tax Settings Page. It features a sidebar on the left and a main content area. In the main area, under "Billing address", the "Use my contact address" option is selected, showing the address: AKurdi, Thane Bihar 123456, IN. The "Use a new address" option is also available. Below this, a section asks "Do you have a PAN?". It defines PAN as a Permanent Account Number and explains that it is a ten-digit alphanumeric number issued by the Indian Income Tax Department. It states that the number is printed on the front of your PAN card. Two radio buttons are provided: "Yes" (unchecked) and "No" (checked). A note below says: "You can go on the Tax Settings Page on Billing and Cost Management Console to update your PAN information." At the bottom, a large orange button is labeled "Verify and Continue (step 3 of 5)". A note below the button states: "You might be redirected to your bank's website to authorize the verification charge." There is also a decorative blue ribbon graphic at the bottom right.

11. Fill OTP and Click “Make Payment”

Merchant : AMAZON	Transaction Amount : ₹ 2.00	SBI Debit Card : 5242xxxx xxxx xx16
-------------------	-----------------------------	-------------------------------------

Authenticate Payment
OTP sent to your mobile number ending 98xx6xxx40

Enter One Time Password (OTP)

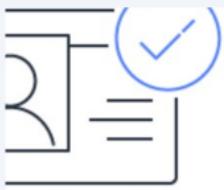
Make Payment [Resend OTP](#)

[Cancel and Go back to merchant](#)

 PCI DSS Certified

Important - SBI Debit Card holder can **Switch-On** and **Switch-Off** Usage (Domestic/International) and Channels (Ecom/POS/ATM/NFC) for his/ her SBI Debit Card through ATM/ Internet Banking/ YONO App/ YONO Lite/ IVR and Branches.

12. Change Country or region code to: India (+91)



Before you can use your AWS account, you must verify your phone number. When you continue, the AWS automated system will contact you with a verification code.

How should we send you the verification code?

Text message (SMS)
 Voice call

Country or region code

▼

Mobile phone number

Security check






13. Provide the “Mobile phone number”

CONTACT YOU WITH A VERIFICATION CODE.

How should we send you the verification code?

Text message (SMS)
 Voice call

Country or region code

India (+91)

Mobile phone number

Type the characters as shown above

- 14.
1. Fill the CAPTCHA
 2. Click “Send SMS (step 4/5)”

India (+91)

Mobile phone number

Please select "Play Audio" button below to continue

A
II C

Please type the numbers that you hear

814692

Send SMS (step 4 of 5)

15. 1. Type Code send on your phone number.
2. Click “continue (step 4/5)”

The screenshot shows the AWS sign-up process at step 4/5. The title "Sign up for AWS" is at the top. Below it, a section titled "Confirm your identity" contains a "Verify code" input field with the value "8970". An orange "Continue (step 4 of 5)" button is below the input field. A note below the button says: "Having trouble? Sometimes it takes up to 10 minutes to retrieve a verification code. If it's been longer than that, [return to the previous page](#) and try again." To the left of the main form, there is a small icon of a circuit board or server components with a checkmark inside a circle. To the right, there is a decorative graphic of a blue wireframe house.

16. Select Academic in field 1. | Select Individual in field 2.
Click “Continue (step 4/5)”

The screenshot shows the AWS sign-up process at step 4/5. The title "Sign up for AWS" is at the top. Below it, a section titled "Confirm your identity" includes a link "(i) Info". Under "Primary purpose of account registration", it says: "Choose one that best applies to you. If your account is tied to a business, select the one that applies to your business." A dropdown menu shows "Academic" selected. Below that, "Ownership type" is set to "Individual". An orange "Continue (step 4 of 5)" button is at the bottom. The same decorative elements as the previous step are present: a circuit board icon with a checkmark, and a blue wireframe house graphic.

17. Click "Complete sign up"

- For account and billing issues only
- Access to Personal Health Dashboard & Trusted Advisor



- 12 (business)-hour response times



- 1-hour response times
- Full set of Trusted Advisor best-practice recommendations



Need Enterprise level support?

From \$15,000 a month you will receive 15-minute response times and concierge-style experience with an assigned Technical Account Manager. [Learn more](#)

Complete sign up

18. Click "Go to the AWS Management Console"



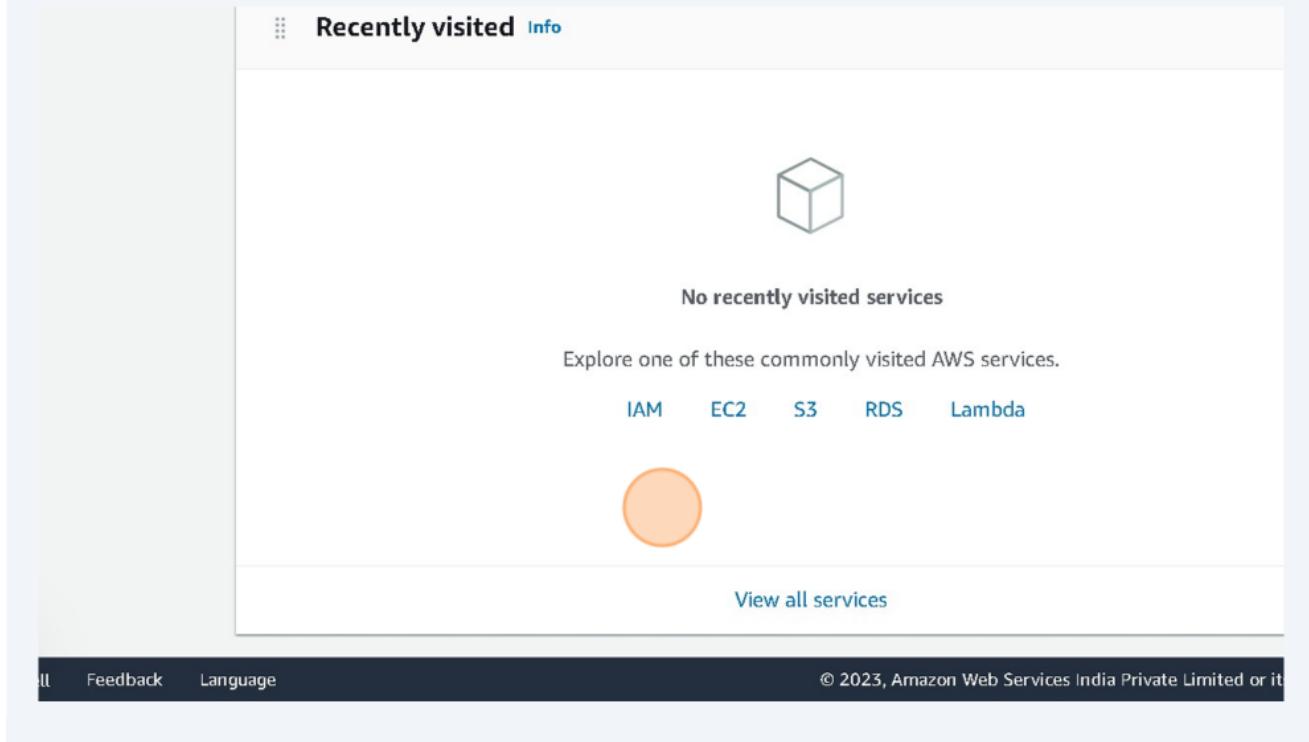
Congratulations

Thank you for signing up for AWS.

We are activating your account, which should only take a few minutes. You will receive an email when this is complete.

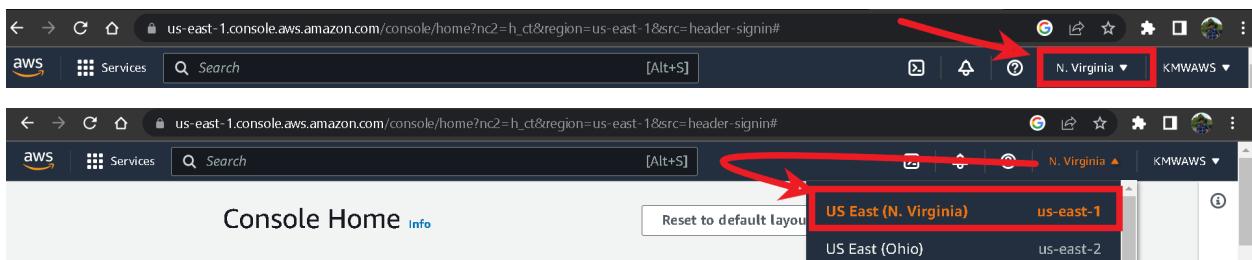
Go to the AWS Management Console

19. AWS account "Sign Up" done successfully.



AWS Account Setup End

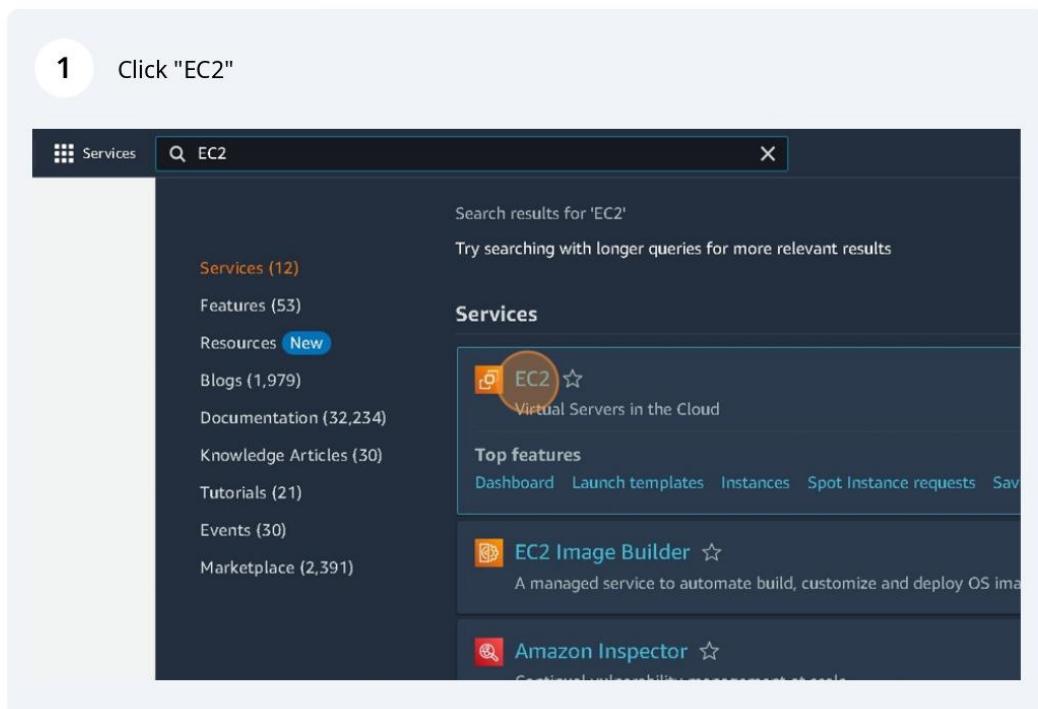
Change Location to : **US East (N. Virginia)**.



Location Change End

Setup EC2 Instance in AWS:

→ AWS → EC2



2 Click "Launch instance"

The screenshot shows the AWS EC2 console. On the left, a sidebar lists various EC2 services: EC2 Global View, Events, Limits, Instances (selected), Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Scheduled Instances, Capacity Reservations, and AMIs. The main content area has a header: "Easily size, configure, and deploy Microsoft SQL Server Always On availability groups on AWS using the AWS Launch Wizard for SQL Server. [Learn more](#)". Below this is a "Launch instance" section with a note: "To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud." It contains two buttons: "Launch instance" (highlighted with a red circle) and "Launch instance from template". A note below says: "Note: Your instances will launch in the US East (N. Virginia) Region". To the right is a "Service health" section showing "Region: US East (N. Virginia)" and "Status: This service is operating normally". At the bottom is a "Scheduled events" section.

3 Click the "Name" field.

The screenshot shows the "Launch an instance" wizard. The first step is "Name and tags". It has a "Name" field containing "e.g. My Web Server" (highlighted with a red circle) and a "Add additional tags" link. The second step is "Application and OS Images (Amazon Machine Image)". It includes a search bar with the placeholder "Search our full catalog including 1000s of application and OS images".

4 Type "Jenkins"

5 Click here.

Application and OS Images (Amazon Machine Image) Info

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Quick Start

Windows	Red Hat	SUSE Linux	Debian
 Microsoft	 Red Hat	 SUSE	 Debian

Amazon Machine Image (AMI)

Amazon Linux 2023 AMI Free tier eligible

Browse more AMIs
Including AMIs from AWS, Marketplace and the Community

Numb
1

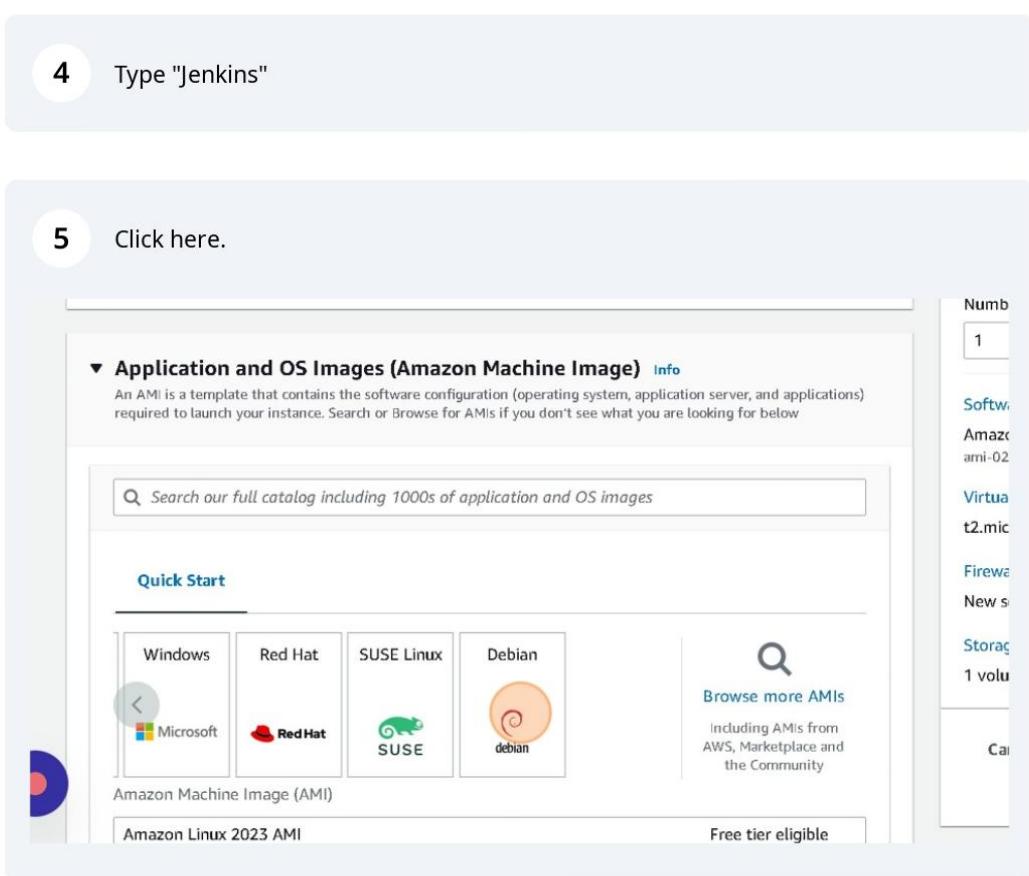
Softw
Amazo
ami-02

Virtua
t2.mic

Firewa
New s

Stora
1 volu

Call



6 Click "Create new key pair"

The screenshot shows the 'Create Function' dialog. On the left, there's a sidebar with 'Compute resources type' and '6 USD per Hour'. The main area has a note about connecting to the instance. A large orange circle highlights the 'Create new key pair' button in the middle of the page. To the right is a 'Summary' section with details like 'Number of instances: 1', 'Software Image (AMI): Debian 11 (20230515-1381)', 'Virtual server type (instance type): t2.micro', 'Firewall (security group): New security group', and 'Storage (volumes): 1 volume(s) - 8 GiB'. At the bottom are 'Cancel' and 'Launch instance' buttons.

7 Give: "Key pair name"

The screenshot shows the 'Create Key Pair' dialog. It includes sections for 'Key pair (login)' (with a note about using it to connect securely), 'Network settings' (with options for VPC, Subnet, and Auto-assign public IP), and 'Key pair type' (with options for RSA and ED25519). The 'Key pair name' field is highlighted with an orange circle. Below it, a note says 'Enter key pair name' and a warning says 'The request must contain the parameter KeyName'. The 'Private key file format' section shows options for '.pem' and '.ppk', with '.ppk' also highlighted with an orange circle. At the bottom are 'Cancel' and 'Create' buttons.

8 Type "jenkins"

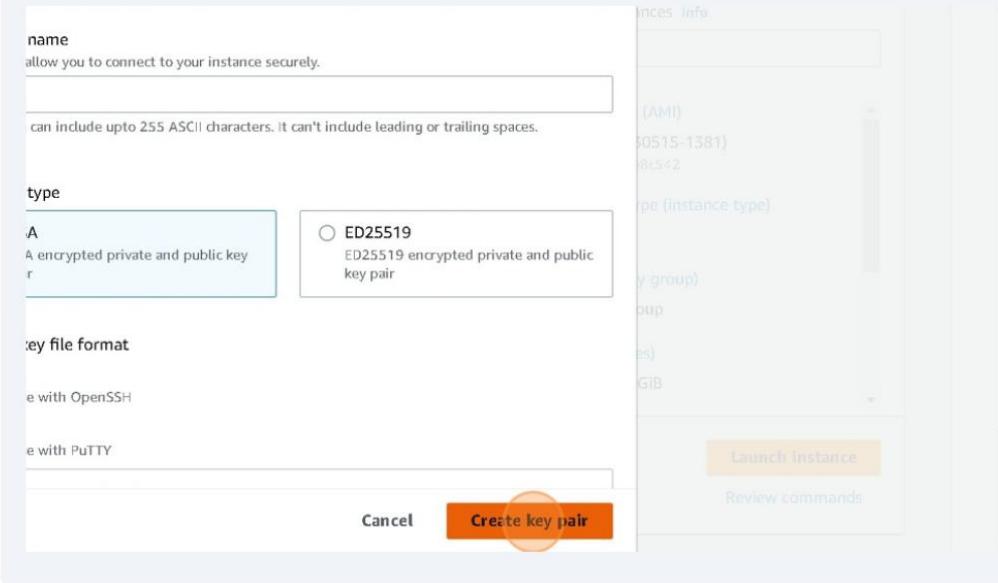
9 Click "RSA"

The screenshot shows the AWS Lambda console interface for creating a new key pair. The 'Key pair name' field is filled with 'jenkins'. In the 'Key pair type' section, the 'RSA' option is selected. Under 'Private key file format', the '.ppk' option is selected. The 'Create' button is visible at the bottom right.

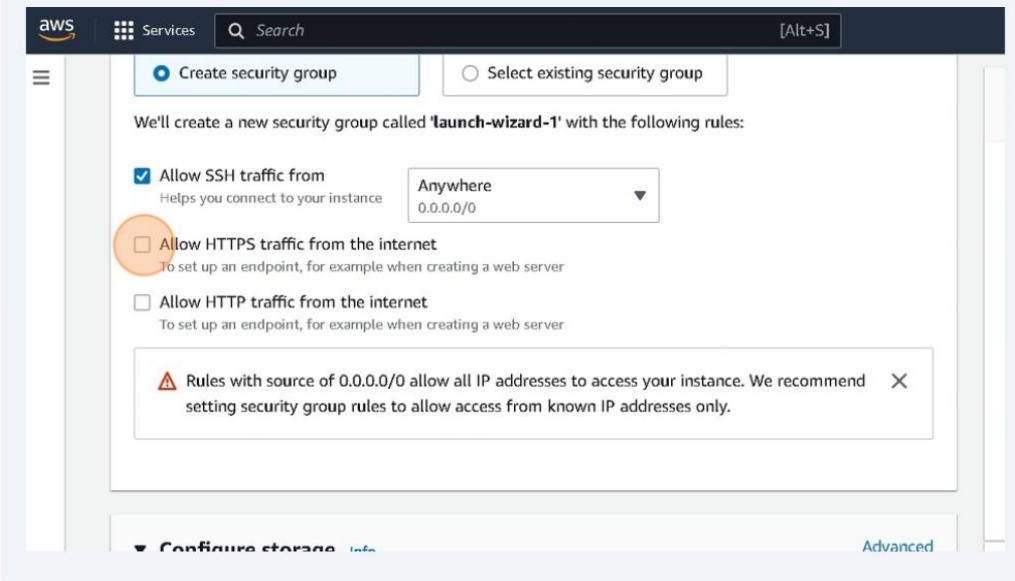
10 Click ".ppk".

The screenshot shows the AWS Lambda console interface for creating a new key pair. The 'Key pair name' field is filled with 'jenkins'. In the 'Key pair type' section, the 'RSA' option is selected. Under 'Private key file format', the '.ppk' option is selected and highlighted with a red circle. The 'Create' button is visible at the bottom right.

11 Click "Create key pair"

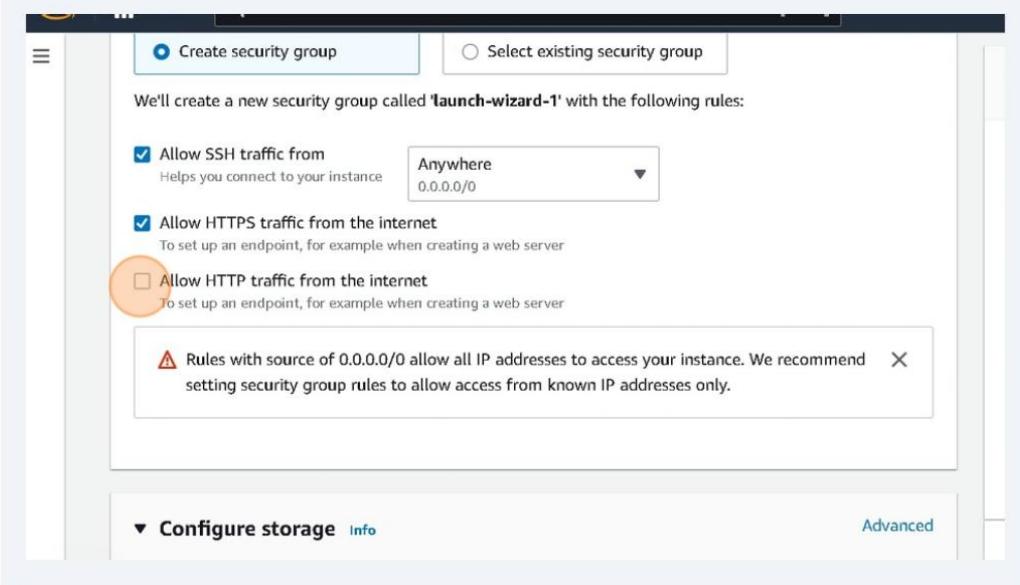


12 Under Networking.



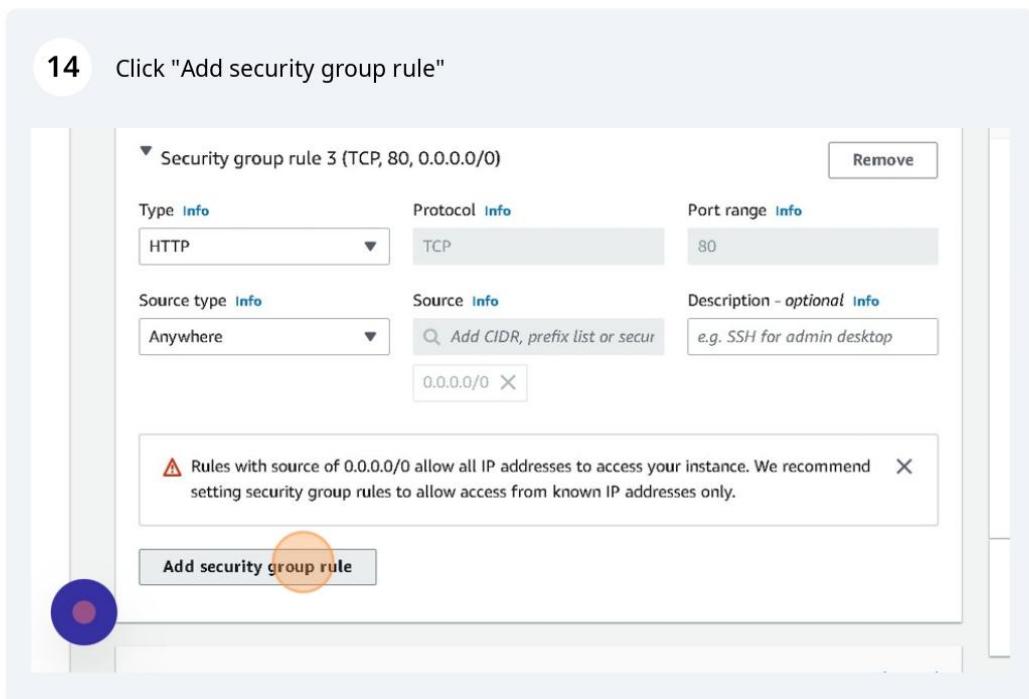
13

1. Click "Allow HTTP traffic from the internet"
2. Click "Allow HTTPS traffic from the internet"



14

- Click "Add security group rule"



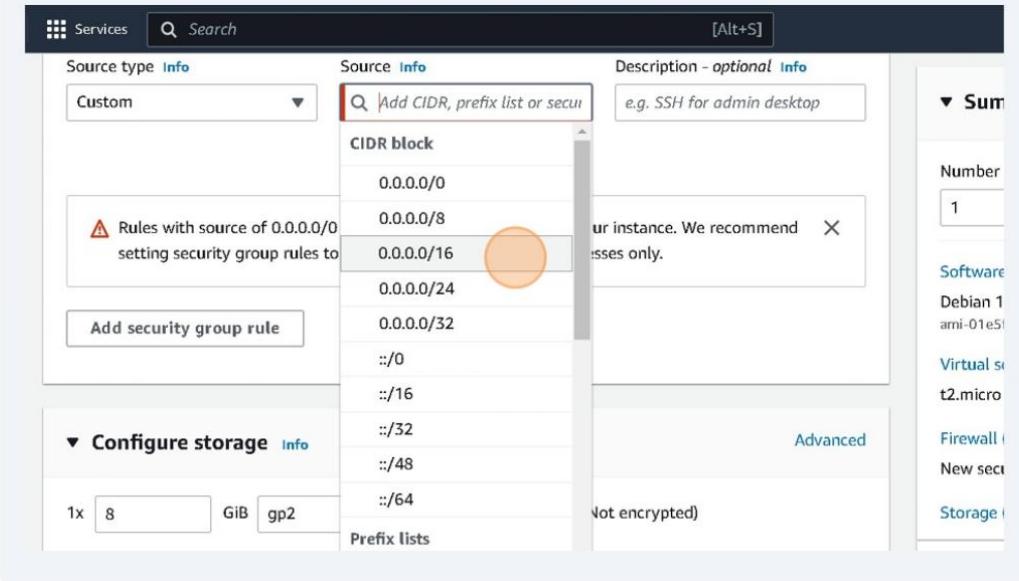
15 Click the "Port range Info" field.

The screenshot shows the AWS CloudFormation Launch Wizard Step 4: Set instance details. On the left, there's a list of rules for launching an Amazon Linux 2 instance. Rule 4 (TCP, 0) has its 'Port range Info' field highlighted with a yellow circle. To the right, there are configuration sections for the instance: 'Number of instances' set to 1, 'Software Image (AMI)' set to Debian 11 (20230515-1381), 'Virtual server type (instance type)' set to t2.micro, 'Firewall (security group)' set to 'New security group', and 'Storage (volumes)'. At the bottom right are 'Cancel' and 'Launch' buttons, with 'Launch' being orange and circled in yellow.

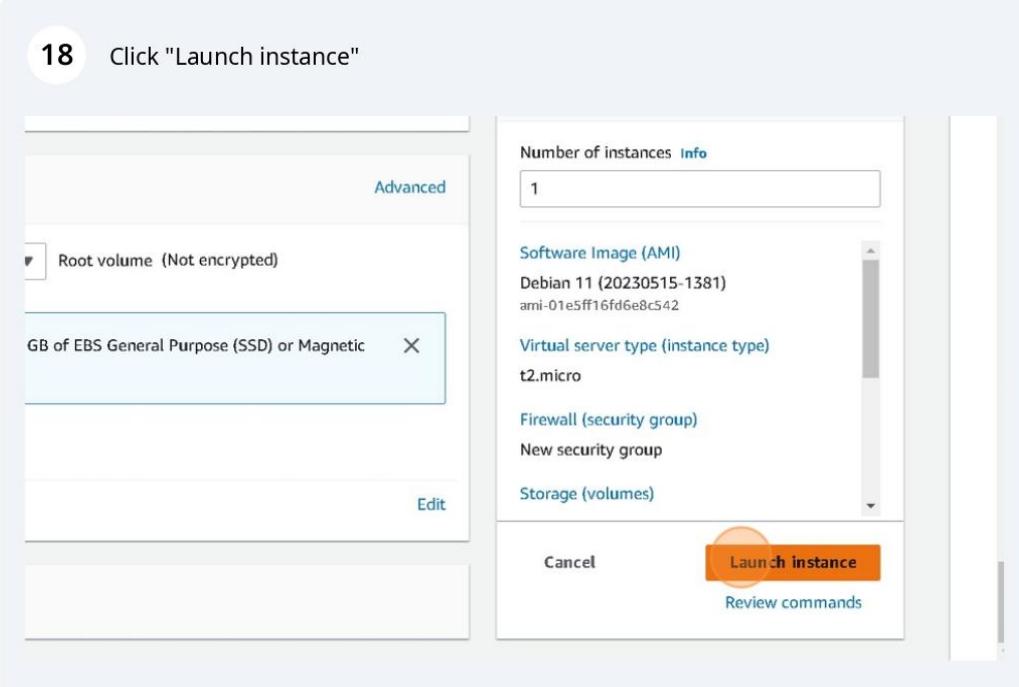
16 Type "8080"

The screenshot shows the AWS CloudFormation Launch Wizard Step 4: Set instance details. Rule 4 (TCP, 0) now has its 'Port range Info' field set to '8080' and is highlighted with a yellow circle. The configuration on the right remains the same as in the previous step: 1 instance of Debian 11 (20230515-1381) on t2.micro, using 'New security group', and the 'Launch' button is circled in yellow.

17 Click "0.0.0.0/0"



18 Click "Launch instance"



19 Click "Instances"

The screenshot shows the AWS EC2 Instances launch success page. At the top, there's a navigation bar with the AWS logo, 'Services' dropdown, a search bar containing 'Search' with a keyboard shortcut '[Alt+S]', and a 'Success' message: 'Successfully initiated launch of instance (i-0b214898a53d659bf)'. Below this, a 'Launch log' link is visible. A 'Next Steps' section follows, featuring a search bar with placeholder text 'What would you like to do next with this instance, for example "create alarm" or "create backup"' and three buttons: 'Create billing and free', 'Connect to your instance', and 'Connect an RDS'.

20 Click "i-0b214898a53d659bf"

The screenshot shows the AWS EC2 Instances list page. The title bar includes 'Services', a search bar, and a keyboard shortcut '[Alt+S]'. Below the title, it says 'Instances (1) Info'. A search bar with placeholder text 'Find instance by attribute or tag (case-sensitive)' is present. A table lists one instance: 'Jenkins' with Instance ID 'i-0b214898a53d659bf', Instance state 'Pending', and Instance type 't2.micro'. The 'Instance ID' column is highlighted with an orange circle. A 'Select an instance' button is at the bottom of the list.

21 EC2 Instance Setup Successful

- Instances → Your Instance → Copy Public IP
- Open PuTTY → Paste IP → Connections → SSH → Auth
- Credentials → Private Key → Save Settings
- Login → Username: **admin** [Login Successful]

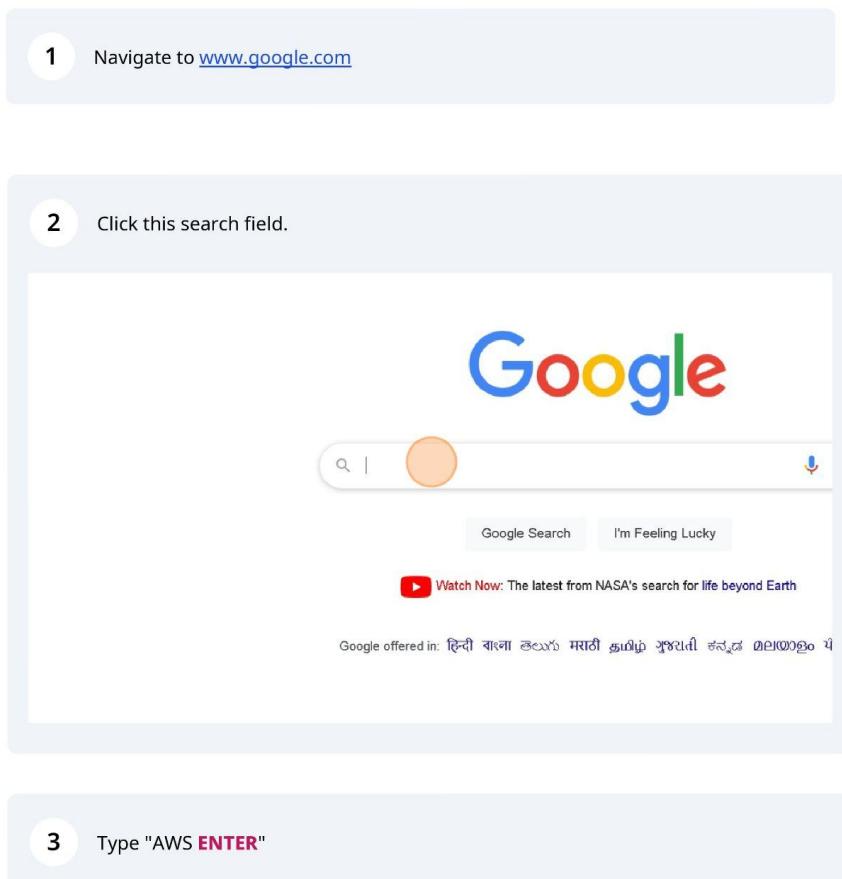
```
→ apt get upgrade  
→ apt get update  
→ apt get-install apache2  
→ sudo systemctl restart apache2  
→ sudo systemctl status apache2
```

→ Verify On Browser Using the IP Address

→ Now Again Go to : AWS → Instance → Instance State: Stop Instance

AWS EC2 End

How to Create an S3 Bucket in AWS:



- 4 Click "Amazon Web Services - AWS Official Site"

The screenshot shows a Google search results page for the query "AWS". At the top, there's a navigation bar with links for News, Images, Login, Console, Certification, Tutorial, Careers, Educate, and EC2. Below the search bar, it says "About 65,90,00,000 results (0.37 seconds)". A "Sponsored" result from Amazon.com is displayed, showing the URL <https://aws.amazon.com/account/sign-up>. The title "Amazon Web Services - AWS Official Site" is underlined. Below the link, there's a brief description: "Build, Deploy, and Manage Websites, Apps or Processes On AWS Secure, Reliable Network. Sign Up for a Free Account & Experience AWS Secure, Reliable, Scalable Services. Easily Manage Clusters. Highly Scalable. No Upfront Commitment." Further down, there are sections for "What is Cloud Computing?", "Amazon Web Services FAQs", and "Learn How AWS Can Help Lower Costs & Innovate Faster. Free For 1 Year."

- 5 Click "Sign In to the Console"

The screenshot shows the official AWS homepage. At the top, there's a dark header with links for Contact Us, Support, English, My Account, and a prominent orange "Sign In to the Console" button. Below the header, there's a navigation bar with links for Learn, Partner Network, AWS Marketplace, Customer Enablement, Events, Explore More, and a search icon. Underneath the navigation, there are links for How to Create an Account, Featured Offers for Business, FAQs, and Terms and Conditions. The main content area features a large banner with the text "AWS Free Tier" and a subtext "-on experience with the AWS platform, products, and services". Below the banner, there's a link to "Learn more about AWS Free Tier".

6 Click "S3"

The screenshot shows the AWS 'Recently visited' dashboard. At the top, there is a header with three dots, the text 'Recently visited', and an 'Info' link. Below this is a large, empty white area with a small 3D cube icon in the center. Below the icon, the text 'No recently visited services' is displayed. Underneath this, there is a section titled 'Explore one of these commonly visited AWS services.' with five buttons: IAM, EC2, S3 (which is highlighted with a yellow circle), RDS, and Lambda. At the bottom of the main content area is a 'View all services' button. The footer of the page includes links for 'Language', '© 2023, Amazon Web Services India Private Limited or its affiliates.', and 'Previous step'.

7 Click "Create bucket"

The screenshot shows the 'Create a bucket' wizard. The title 'Create a bucket' is at the top. Below it, a text block explains that every object in S3 is stored in a bucket and that to upload files and folders to S3, a bucket needs to be created where the objects will be stored. A prominent orange 'Create bucket' button is centered below this text. To the left of the main content, there is a dark sidebar with the text 'eve any amount nywhere' and 'offers industry-leading scalability, data availability,'. On the right side, there is a 'Pricing' section with the text 'With S3, there are no minimum fees. You only pay for what you use. Prices are based on the location of'. The top of the screen shows the AWS navigation bar with options like 'File', 'Edit', 'View', 'Global', and 'KMWAWS'.

- 8 Click the "Bucket name" field.

Amazon S3 > Buckets > Create bucket

Create bucket Info

Buckets are containers for data stored in S3. Learn more 

General configuration

Bucket name 

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

AWS Region 

Copy settings from existing bucket - *optional*
Only the bucket settings in the following configuration are copied.

- 9 Type "my_bucket"

10 Click "EU (Stockholm) eu-north-1"

rs for data stored in S3. Learn more 

General configuration

Bucket name 

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

AWS Region 

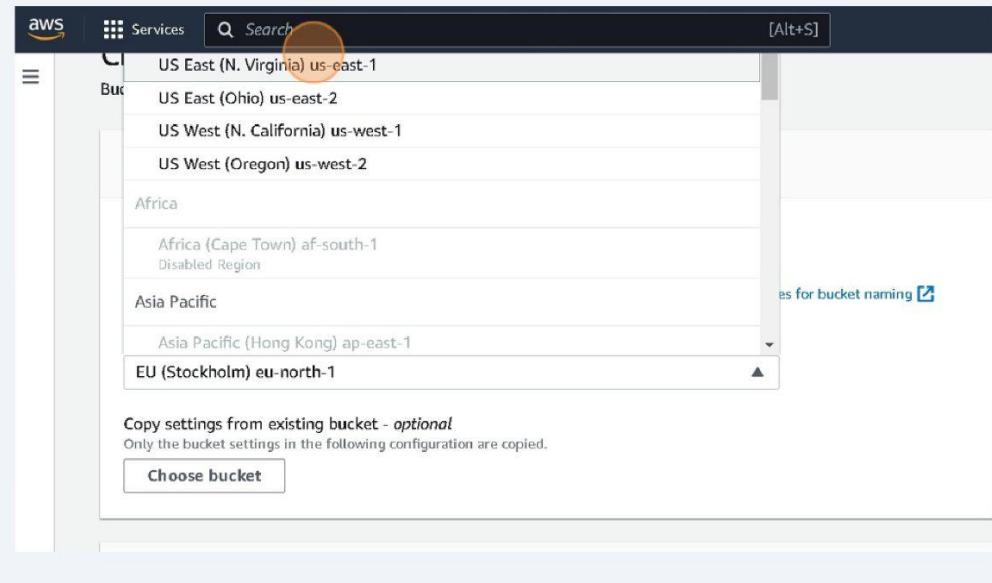
Copy settings from existing bucket - *optional*
Only the bucket settings in the following configuration are copied.

Region info

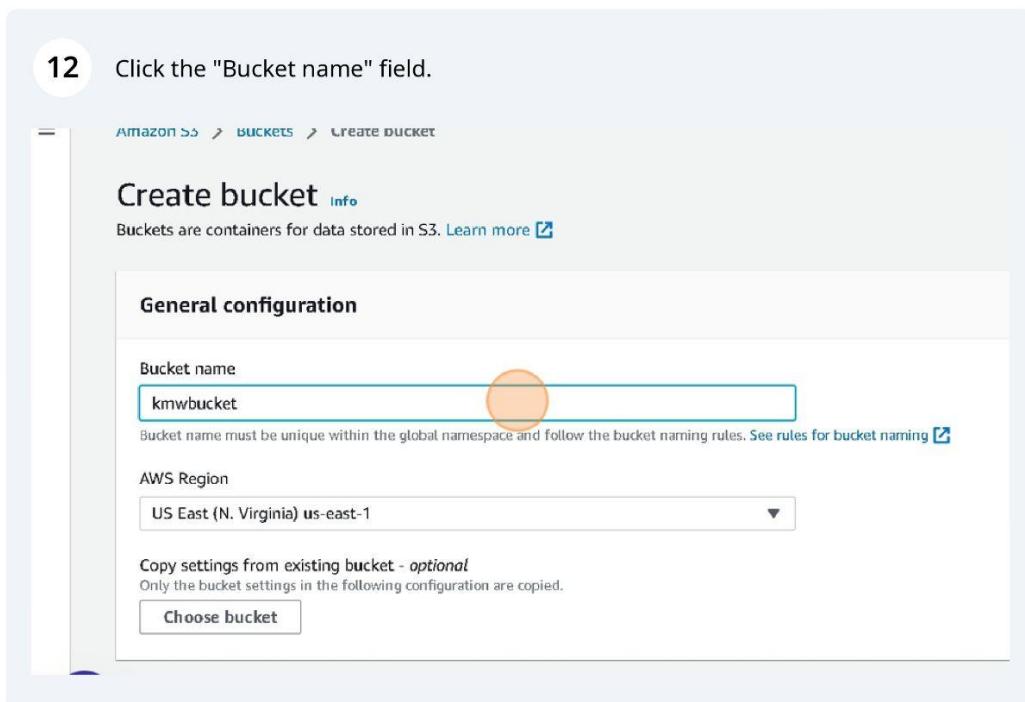
Region: eu-north-1
Language: English (US)

© 2023, Amazon Web Services India Private Limited or its affiliates. Privacy

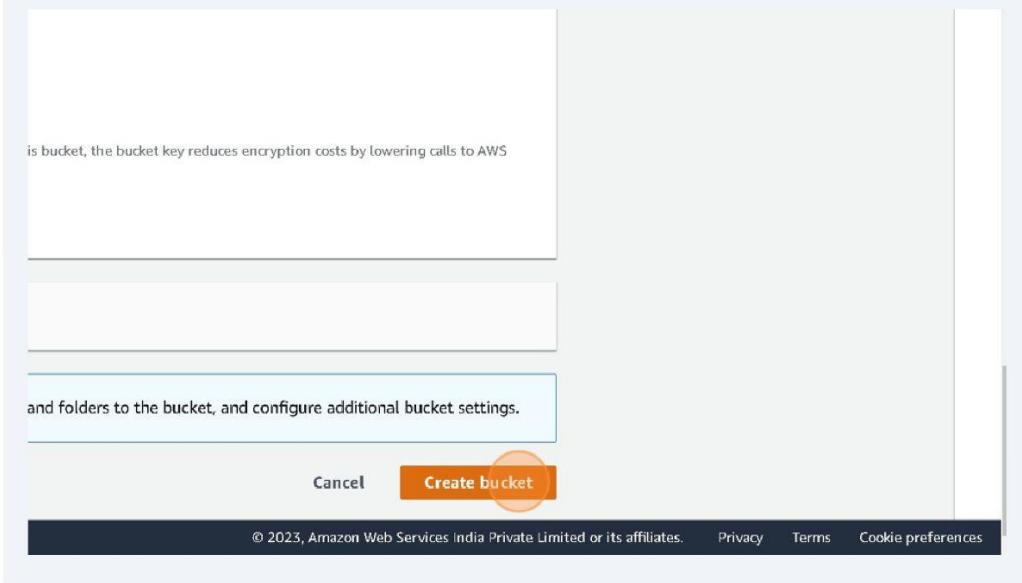
11 Click "US East (N. Virginia) us-east-1"



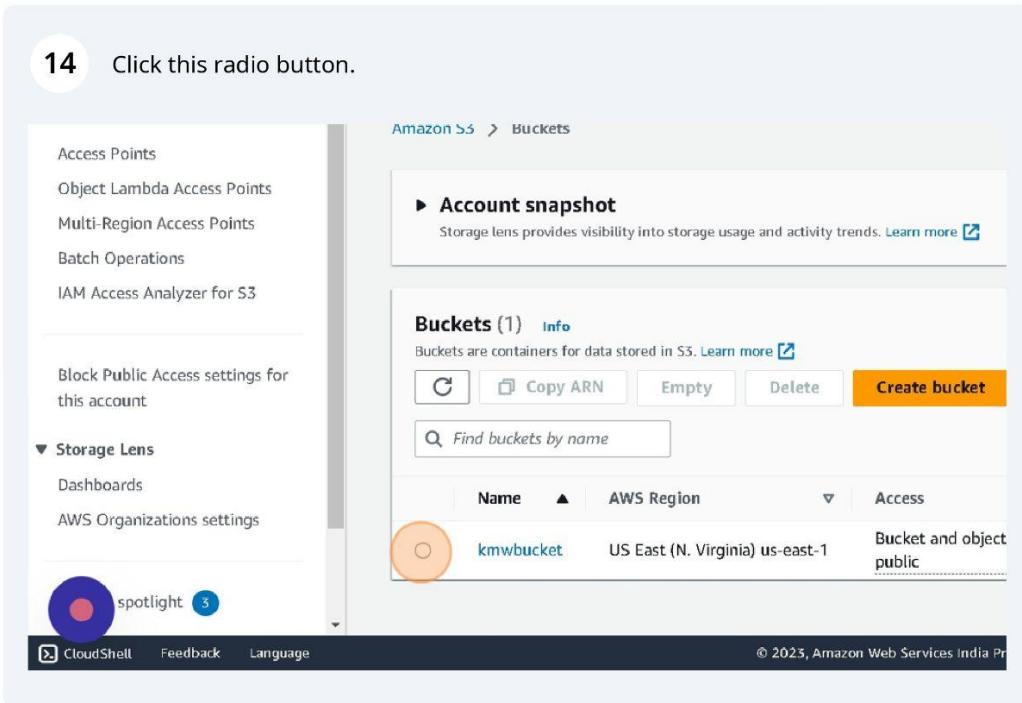
12 Click the "Bucket name" field.



13 Click "Create bucket"



14 Click this radio button.



15 Click "kmwbucket"

The screenshot shows the AWS S3 Buckets page. On the left, there's a sidebar with links like Access Points, Object Lambda Access Points, Multi-Region Access Points, Batch Operations, IAM Access Analyzer for S3, Block Public Access settings for this account, Storage Lens (with Dashboards and AWS Organizations settings), and CloudShell. The main area has a heading 'Account snapshot' with a link to 'Storage lens provides visibility into storage usage and activity trends'. Below it is a section titled 'Buckets (1) [Info](#)' with a note that 'Buckets are containers for data stored in S3.' It shows one bucket: 'kmwbucket' (highlighted with a red circle), located in 'US East (N. Virginia)' ('us-east-1'), with 'Bucket and objects public' access. There are buttons for Copy ARN, Empty, Delete, and Create bucket.

16 Click "Properties"

The screenshot shows the AWS S3 Objects Properties tab. The top navigation bar includes Services, Search, and a [Alt+S] key shortcut. The left sidebar is identical to the previous screenshot. The main area has tabs for Objects (highlighted with a red circle), Properties, Permissions, Metrics, Management, and Actions. Under the Objects tab, it says 'Objects (0)' and provides a note about granting permissions. It has buttons for Create folder, Upload, and a search bar for 'Find objects by prefix'. A table at the bottom lists columns for Name, Type, Last modified, and Size, with a note 'No objects' and 'You don't have any objects in this bucket'.

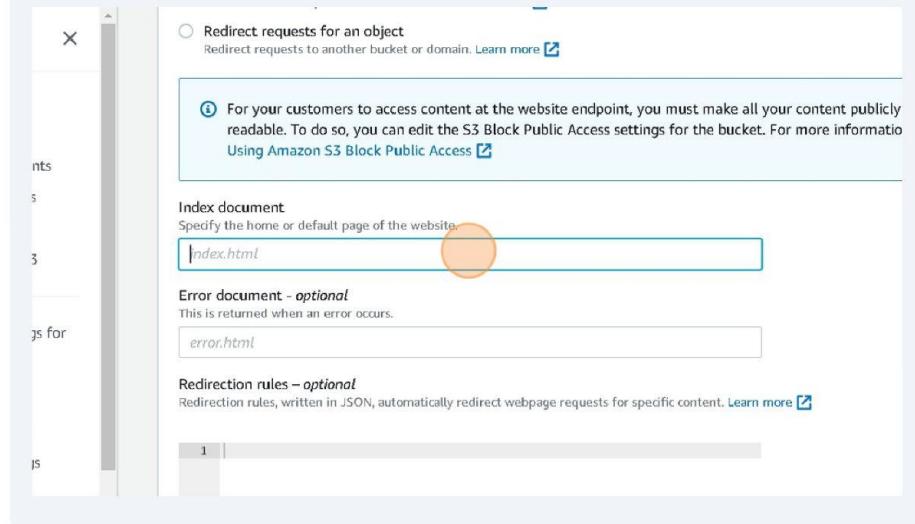
17 Click "Edit"

The screenshot shows the 'Static website hosting' section of an AWS S3 bucket's properties. It includes fields for 'Index document' (index.html) and 'Error document' (error.html). Below these, there is a note about requester pays access and an 'Edit' button. Further down, there is another 'Edit' button next to the 'Static website hosting' status. At the bottom of the page, there is a footer bar with copyright information and links for Privacy, Terms, and Cookie preferences.

18 Click the "Enable" field.

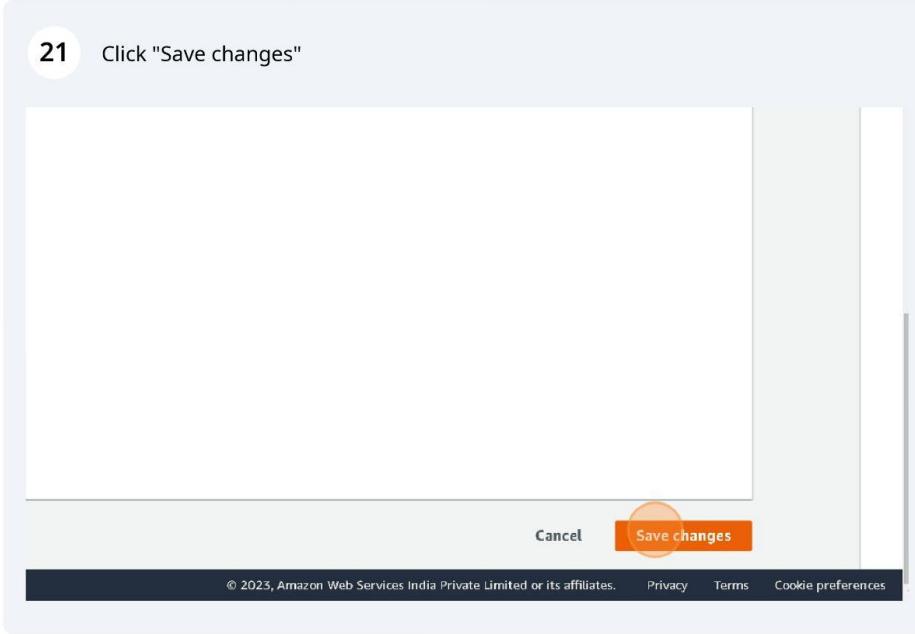
The screenshot shows the 'Edit static website hosting' dialog. On the left is a sidebar with navigation links like Buckets, Access Points, Object Lambda Access Points, Multi-Region Access Points, Batch Operations, IAM Access Analyzer for S3, and Block Public Access settings for this account. Below that is a 'Storage Lens' section with Dashboards and AWS Organizations settings. At the bottom is a 'spotlight' section. The main right panel displays the 'Static website hosting' configuration. It has a heading, a note about using the bucket for hosting, and a section where the 'Static website hosting' status is set to 'Disable'. There is also an 'Info' link and a note about using this bucket to host a website or redirect requests.

- 19 Click the "Index document" field.

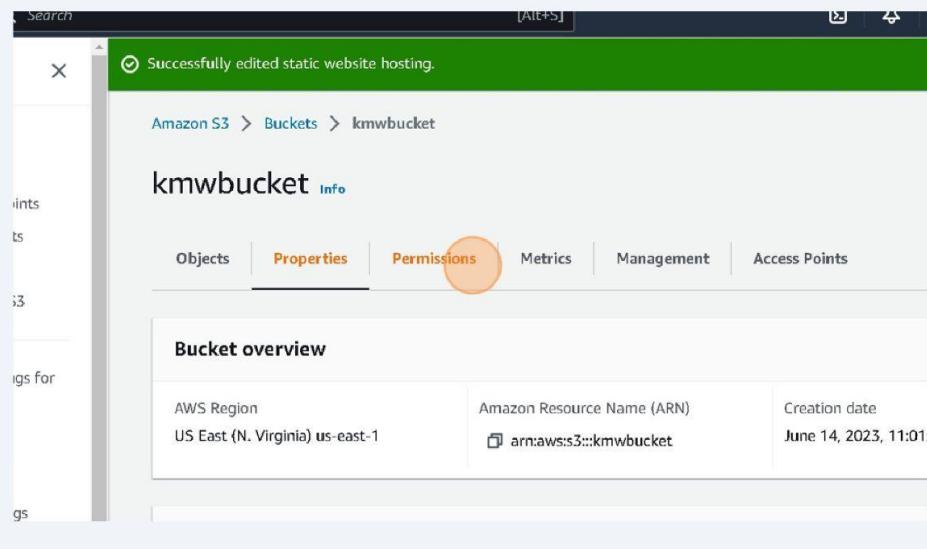


- 20 Type "index.html TAB"

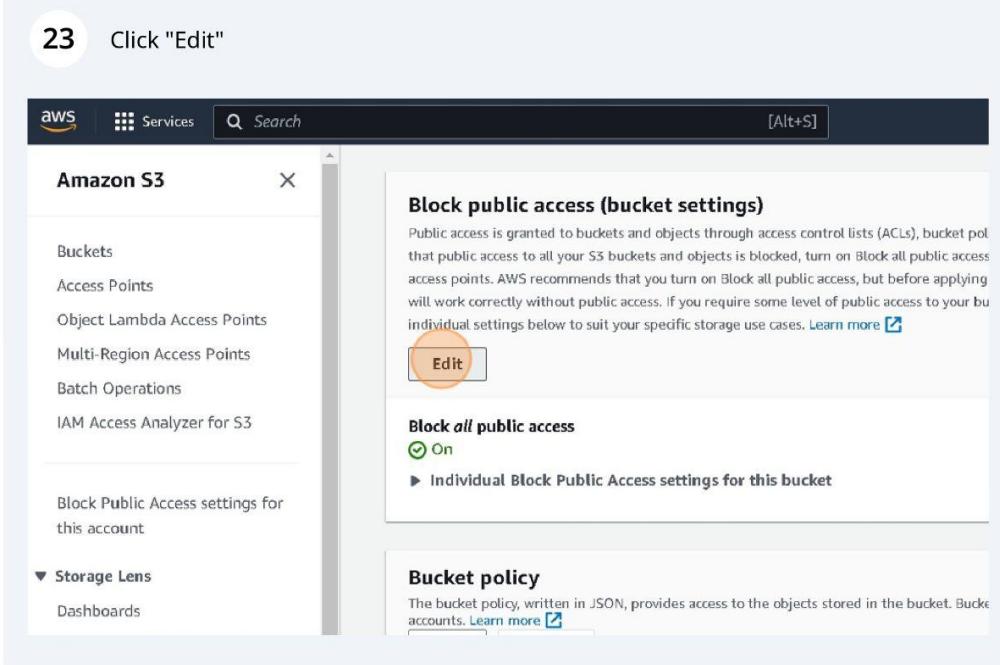
- 21 Click "Save changes"



22 Click "Permissions"



23 Click "Edit"



24

Click "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...

The screenshot shows the AWS S3 console with the 'Block Public Access settings for this account' section selected. A callout bubble highlights the 'Block all public access' checkbox, which is checked. Below it, four other settings are listed, each with a descriptive text and a checked checkbox.

ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access to the bucket and its access points. AWS recommends that you turn on Block all public access, but be sure that your applications will work correctly without public access. If you require some level of access to objects within, you can customize the individual settings below to suit your specific storage needs.

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.

- 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**
Ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

25

Click "Save changes"

The screenshot shows a confirmation dialog box with the title 'public access'. It contains four sections: 'public access', 'public access to buckets and objects granted through new access control lists (ACLs)', 'public access to buckets and objects granted through any access control lists (ACLs)', and 'public access to buckets and objects granted through new public bucket or access point policies'. Each section has a descriptive text and a checked checkbox. At the bottom right of the dialog is a large orange button labeled 'Save changes'.

public access
This setting on is the same as turning on all four settings below. Each of the following settings are independent of one another.

public access to buckets and objects granted through new access control lists (ACLs)
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.

public access to buckets and objects granted through any access control lists (ACLs)
Ignore all ACLs that grant public access to buckets and objects.

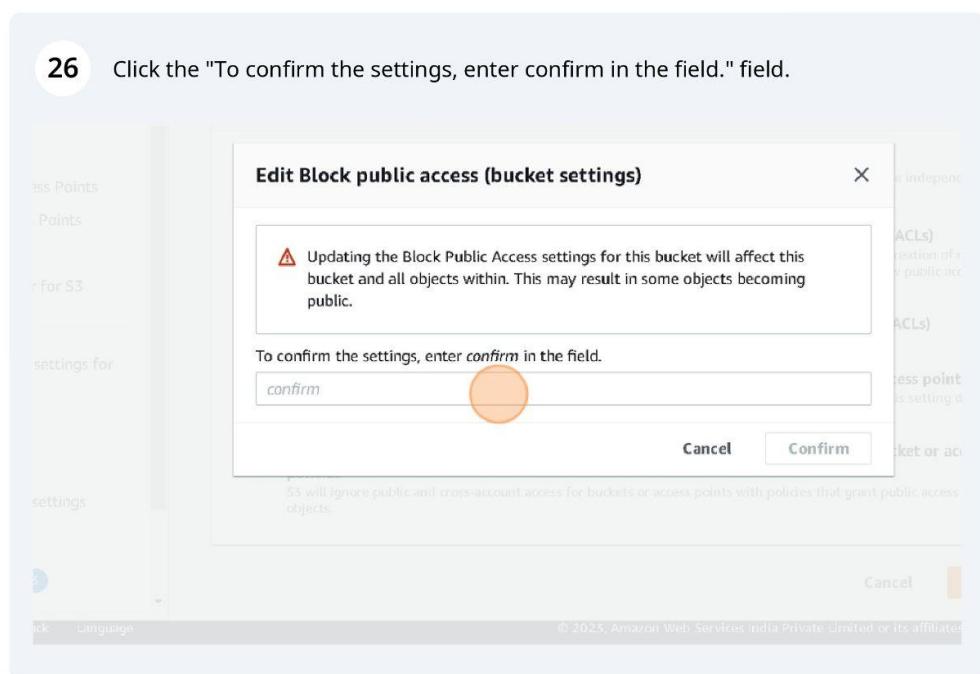
public access to buckets and objects granted through new public bucket or access point policies
Block new bucket and access point policies that grant public access to buckets and objects. This setting doesn't change any existing policies that allow public access to S3 resources.

public and cross-account access to buckets and objects through any public bucket or access point policies
Ignore public and cross-account access for buckets or access points with policies that grant public access to buckets and objects.

Cancel Save changes

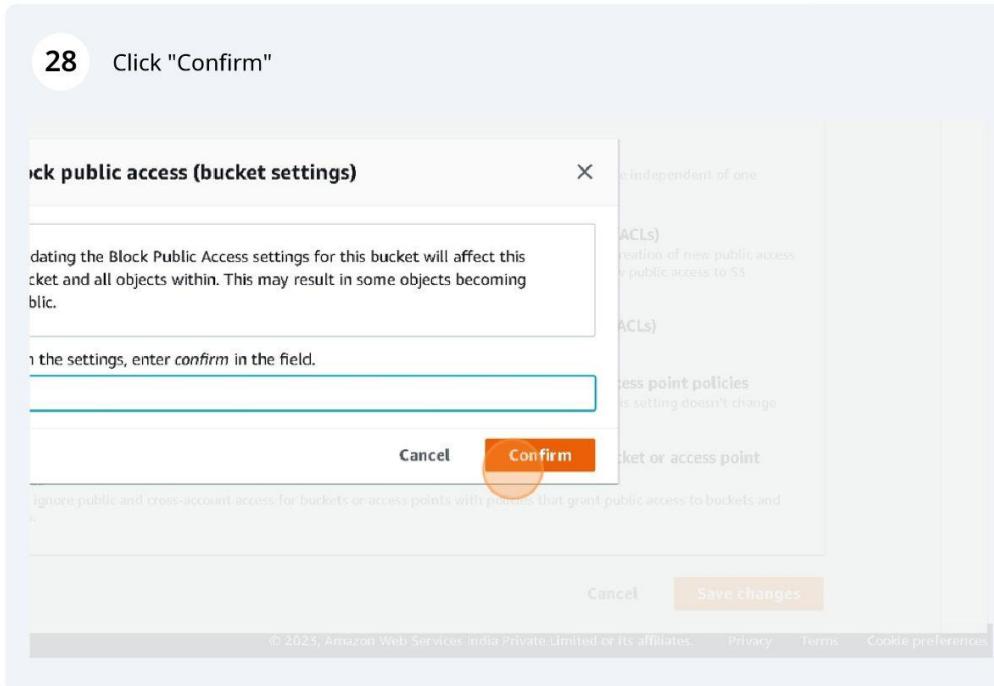
© 2023, Amazon Web Services India Private Limited or its affiliates. Privacy Terms Cookie preferences

26 Click the "To confirm the settings, enter confirm in the field." field.



27 Type "confirm"

28 Click "Confirm"



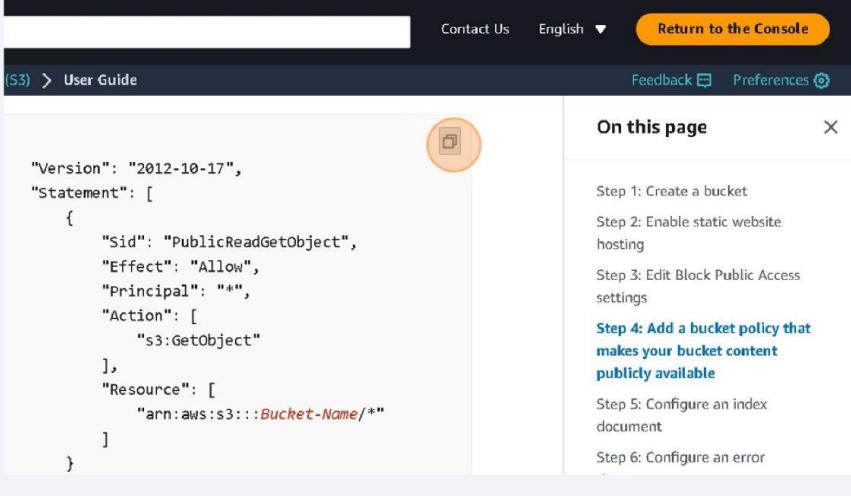
29 Click "Edit"

The screenshot shows the Amazon S3 console. On the left, there's a sidebar with options like Buckets, Access Points, Object Lambda Access Points, Multi-Region Access Points, Batch Operations, IAM Access Analyzer for S3, and Storage Lens. Under Storage Lens, there are links for Dashboards and AWS Organizations settings. The main area displays 'Block Public Access settings for this account'. It shows that the setting is currently 'Off' and provides a link to 'Individual Block Public Access settings for this bucket'. Below this, there's a section titled 'Bucket policy' with a note about JSON bucket policies and a link to learn more. An 'Edit' button is highlighted with an orange circle.

30 Click "Tutorial: Configuring a static website on Amazon S3"

The screenshot shows a Google search results page for the query "tutorial configuring a static website on amazon s3". The first result is a link to a tutorial on Amazon's documentation site: [Tutorial: Configuring a static website on Amazon S3](https://docs.aws.amazon.com/latest/userguide/HowToStaticWebsites.html). Below the search results, there's a "People also ask" section with two collapsed questions: "How do I deploy a static website using S3?" and "Can S3 be used for static web sites?".

31 Click here.



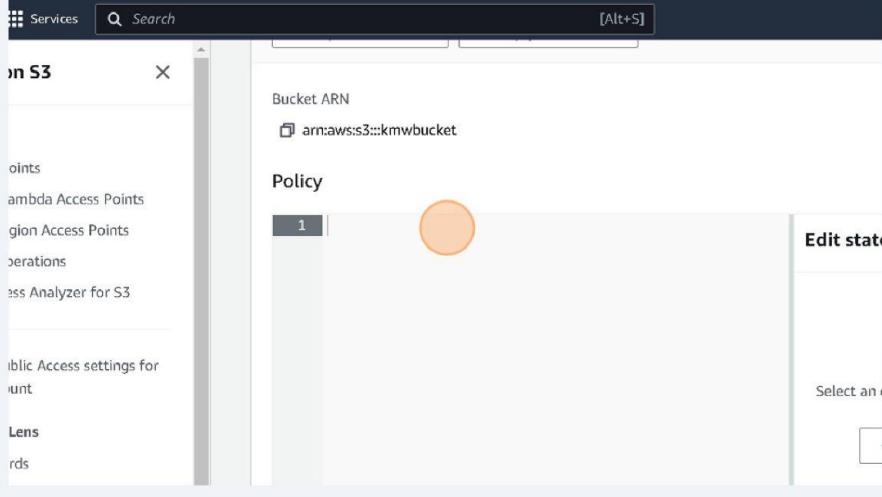
The screenshot shows a portion of the AWS S3 User Guide. On the left, there is a code editor containing a JSON policy document:

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

A large orange circle highlights the "Resource" field in the JSON code. On the right side of the screen, there is a sidebar titled "On this page" with several links:

- Step 1: Create a bucket
- Step 2: Enable static website hosting
- Step 3: Edit Block Public Access settings
- Step 4: Add a bucket policy that makes your bucket content publicly available** (This link is highlighted with an orange circle)
- Step 5: Configure an index document
- Step 6: Configure an error

32 Click here.



The screenshot shows the AWS S3 console for a bucket named "kmwbucket". The left sidebar lists various S3 features like Points, Lambda Access Points, Region Access Points, Operations, and the S3 Analytic for S3. The main panel displays the bucket's ARN and its policy:

Bucket ARN
arn:aws:s3:::kmwbucket

Policy

1 |

Edit state

Select an item

A large orange circle highlights the "1" icon in the Policy section.

33 Press **CTRL + V**

34

Replace: Bucket-Name with "youbucket" name.
Eg: "arn : aws : s3 :: my-bucket/*"

The screenshot shows the AWS IAM Policy editor interface. On the left, there's a sidebar with navigation links like 'Points', 'or S3', 'Settings for', and 'Settings'. The main area is titled 'Policy' and contains a JSON code editor. The code defines a policy with one statement that grants public read access to objects in a bucket named 'Bucket-Name'. A specific line in the code, which contains the placeholder 'Bucket-Name', is highlighted with a yellow oval. To the right of the code editor is a panel titled 'Edit statement' with a sub-section 'Select a stat'. It includes a dropdown menu with options like 'Select an existing statement' and 'add a new stat', and a button '+ Add new s'. At the bottom of the editor, there's a 'Language' dropdown set to 'JSON' and a copyright notice: '© 2023, Amazon Web Services India Private Limited or its affiliates.'

```
1 {  
2     "Version": "2012-10-17",  
3     "Statement": [  
4         {  
5             "Sid": "PublicReadGetObject",  
6             "Effect": "Allow",  
7             "Principal": "*",  
8             "Action": [  
9                 "s3:GetObject"  
10            ],  
11             "Resource": [  
12                 "arn:aws:s3:::Bucket-Name/*"  
13            ]  
14        }  
15    ]  
16}
```

35

Click "Save changes"

The screenshot shows the 'Save changes' step in the AWS IAM Policy editor. The interface includes a sidebar with various AWS services like ASC, Access Analyzer, Account, Activate, Alexa for Business, and Amplify. Below the sidebar, there are sections for 'new statement', 'Add a resource', and 'Add a condition (optional)'. At the bottom, there are status indicators for errors, warnings, and suggestions, along with a 'Preview external access' link. The 'Save changes' button is highlighted with a yellow oval. The footer of the page includes a copyright notice and links for 'Cancel', 'Save changes', 'Privacy', 'Terms', and 'Cookie preferences'.

ASC
Access Analyzer
Account
Activate
Alexa for Business
Amplify

new statement

2. Add a resource

3. Add a condition (optional)

Errors: 0 Errors: 0 Warnings: 0 Suggestions: 0 Preview external access

Cancel

© 2023, Amazon Web Services India Private Limited or its affiliates. Privacy Terms Cookie preferences

36 Click "Objects"

The screenshot shows the AWS S3 console interface. The left sidebar lists 'Buckets', 'Access Points', 'Object Lambda Access Points', 'Multi-Region Access Points', 'Batch Operations', and 'IAM Access Analyzer for S3'. Below this is a section for 'Block Public Access settings for this account'. Under 'Storage Lens', there are 'Dashboards' and 'AWS CloudTrail integration'. The main content area shows the 'kmwbucket' bucket details. The 'Permissions' tab is selected, indicated by an orange circle. A green banner at the top right says 'Successfully edited bucket policy.' The URL in the address bar is 'Amazon S3 > Buckets > kmwbucket'.

37 Click "Upload"

The screenshot shows the 'Objects (0)' page in the AWS S3 console. At the top, there are buttons for 'Create folder' and 'Upload', with 'Upload' being highlighted by an orange circle. Below is a search bar with placeholder text 'Find objects by prefix'. A navigation bar shows page 1 of 1. The main table header includes columns for 'Name', 'Type', 'Last modified', 'Size', and 'Storage class'. A message at the bottom states 'No objects' and 'You don't have any objects in this bucket.' The 'Upload' button is located at the bottom center of the page. The footer contains copyright information and links to 'Privacy', 'Terms', and 'Cookie'.

38 Click "Add files"

want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon

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

)
le will be uploaded.

Remove

Add files

Add folder

< 1 >

▲ Folder

▼

Type

▼

Size

▼

No files or folders

You have not chosen any files or folders to upload.

39 Click "Upload"

text/html

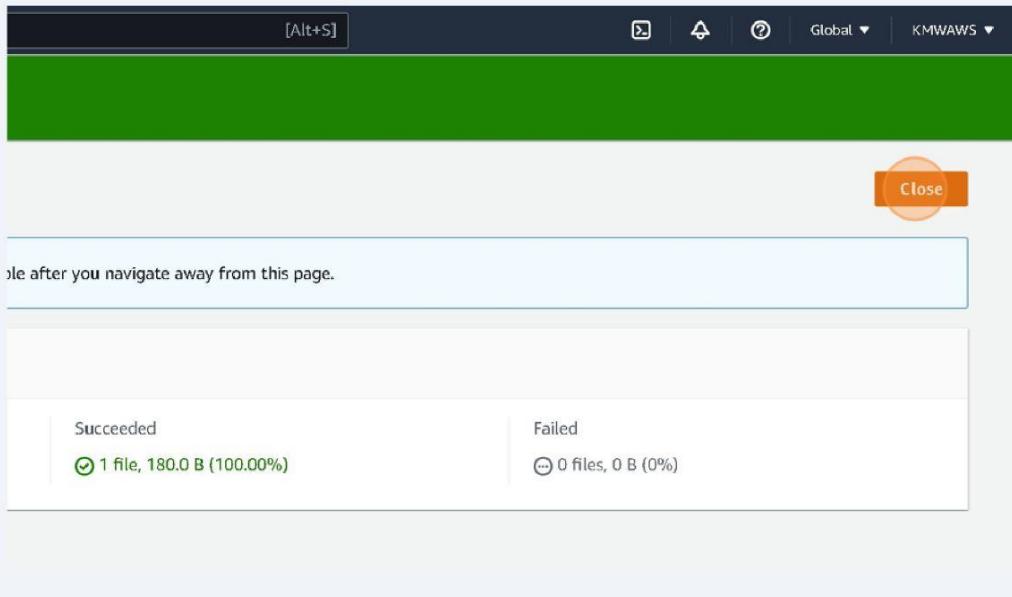
180.0 B

in the specified destination.

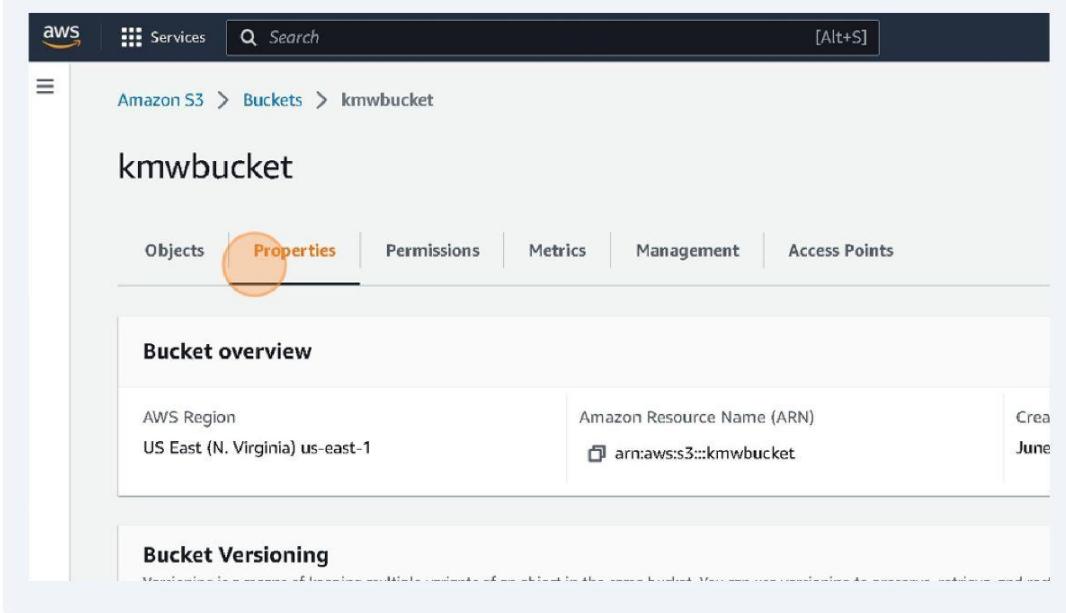
Cancel

Upload

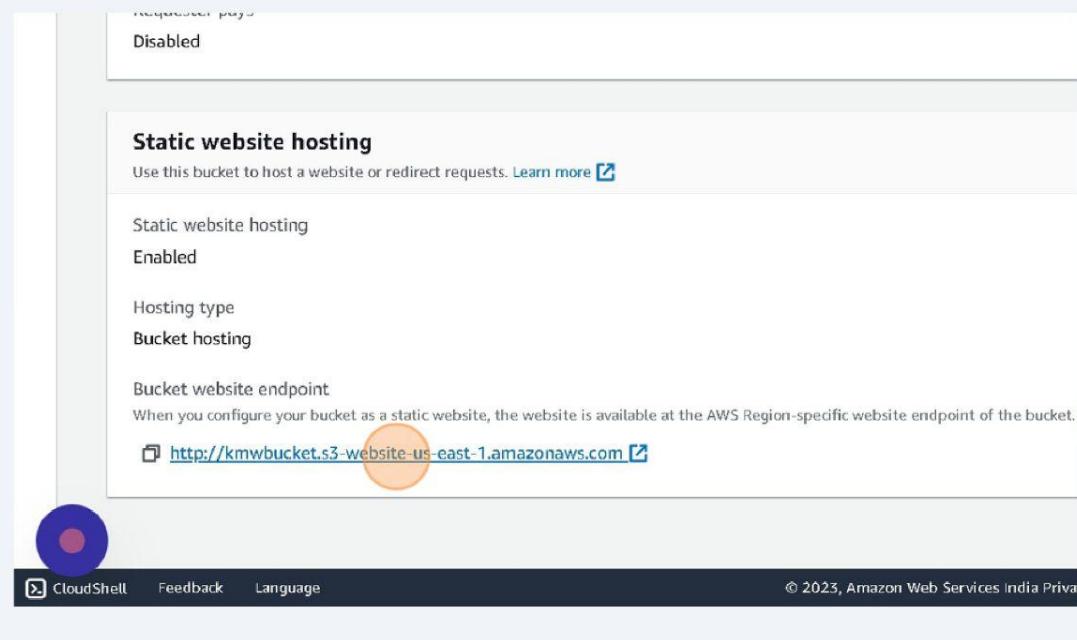
40 Click "Close"



41 Click "Properties"



42 Click "http://kmwbucket.s3-website-us-east-1.amazonaws.com"



43 AWS S3 Setup Successful.

Delete The Bucket:

Amazon S3 → Choose Bucket: **iacsd.com** → Click On: Empty

→ Type : (*permanently delete*) → Click on : Empty → Click on : Delete

→ Type: *bucket_name* [Eg. **iacsd-web**] → Click On: Delete

----- AWS S3 End -----

Git:

- Take a fresh “debian Machine”
- Configure the machine
- Open PuTTY
- Login from the PuTTY

```
sudo su -  
apt-get update  
apt-get install git  
mkdir techgalena  
cd techgalena  
git init  
ls -a  
nano index.html
```

```
GNU nano 3.2          index.html          Modified ^  
  
<!DOCTYPE html>  
<html>  
<head>  
<title>TechGalena</title>  
</head>  
<body>  
<h1>Welcome to My TechGalena</h1>  
<p>This is a single page HTML document.</p>  
<p>Feel free to customize it as per your needs.</p>  
<ul>  
<li>List item 1</li>  
<li>List item 2</li>  
<li>List item 3</li>  
</ul>  
<p>Thank you for visiting!</p>  
</body>  
</html>
```

```
git status
```

```
On branch master  
No commits yet  
Untracked files:  
(use "git add <file>..." to include in what will be committed)  
index.html  
nothing added to commit but untracked files present (use "git add" to track)
```

```
git add index.html  
git status  
git commit -m "This is my first commit"  
git config --global user.email "example@gmail.com"  
git config --global user.name "username" #username = repository name  
git commit -m "This is my first commit"
```

```
git log  
git push origin master  
git remote add origin "git@github.com:kmwaseem15/techgalena.git"  
git push origin master  
ssh-keygen  
cat /root/.ssh/id_rsa.pub
```

Paste this key to github repository: Settings → Deploy Keys → Click on: Add deploy key → Give key a Name Under : Title → Paste the key Under: Key → Click On: Allow Write → Click On: Add key.

The screenshot shows two stacked GitHub repository pages. The top page is for the repository 'kmwaseem15/techgalena' (Public). The bottom page is for the same repository. Both pages have the 'Settings' tab selected. In the top page's 'Deploy keys' section, there is one existing key named 'Master Branch Key'. The 'Key' field contains a long SSH public key. In the bottom page's 'Deploy keys / Add new' section, a new key is being added with the title 'Master Branch Key' and the same long SSH public key in the 'Key' field. A note below the key field states: 'Begins with 'ssh-rsa'', 'ecdsa-sha2-nistp256'', 'ecdsa-sha2-nistp384'', 'ecdsa-sha2-nistp521'', 'ssh-ed25519'', 'sk-ecdsa-sha2-nistp256@openssh.com'', or 'sk-ssh-ed25519@openssh.com''. There is also a checked checkbox for 'Allow write access'.

```
git config pull.rebase true  
git pull origin master  
git push origin master
```

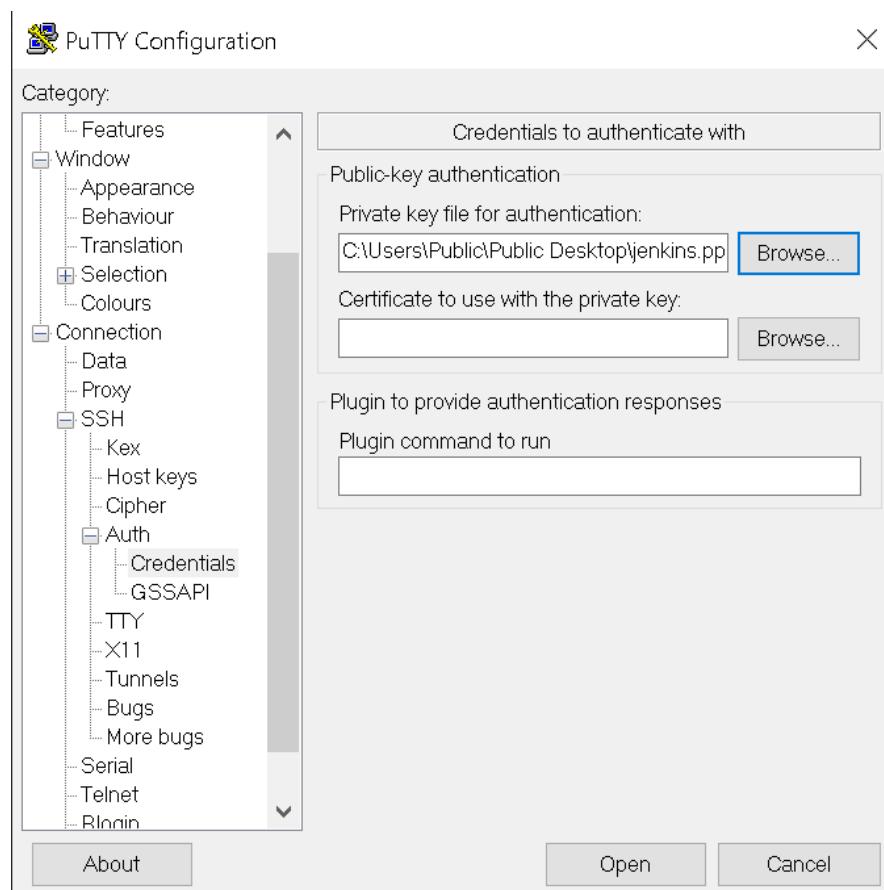
Jenkins Installation On: AWS (Deb 10)

- AWS
- Region: US East (N. Virginia)
- EC2
- Launch Instance
- Name : Jenkins
- OS: Debian
- Create Key Pair
- Networking: Auto Assign Public IP Enable
 - Allow SSH Traffic from: Anywhere
 - Allow HTTP Traffic From Internet
 - Allow HTTPS Traffic From Internet
- Click: Launch Instance

→ Open PuTTY - Enter IP Address

→ Connections → SSH Auth → Credentials → Private Key File For Auth

→ Save Settings → Open



→ Login → Username: **admin** [Login Successful]

```
admin@ip-172-31-0-150: ~
Using username "admin".
Authenticating with public key "jenkins"
Linux ip-172-31-0-150 5.10.0-23-cloud-amd64 #1 SMP Debian 5.10.179-1 (2023-05-12)
) x86_64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/*copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
admin@ip-172-31-0-150:~$
```

```
apt-get update -y
apt-get upgrade -y
apt-get install gnupg -y
apt-get install default-jre -y
curl -fsSL https://pkg.jenkins.io/debian/jenkins.io-2023.key | sudo
tee \
    /usr/share/keyrings/jenkins-keyring.asc > /dev/null
echo deb [signed-by=/usr/share/keyrings/jenkins-keyring.asc] \
    https://pkg.jenkins.io/debian binary/ | sudo tee \
    /etc/apt/sources.list.d/jenkins.list > /dev/null
apt update -y
apt-get install jenkins -y
sudo usermod -a -G root jenkins
systemctl status jenkins
```

- Go to AWS → EC2 → Select Your Instance → Scroll Below : Security
→ Allow Inbound Traffic on port : 8080

The screenshot shows the AWS EC2 Instances page. On the left, there's a sidebar with navigation links like EC2 Dashboard, EC2 Global View, Events, Limits, Instances, Instances, Instance Types, Launch Templates, and Spot Requests. The main area displays a table of instances. One instance is selected, and its details are shown on the right. Under the 'Security Groups' section, it shows 'sg-04cd96a06213b176'. Below this, the 'Inbound rules' section is expanded, showing a table with four rows of rules:

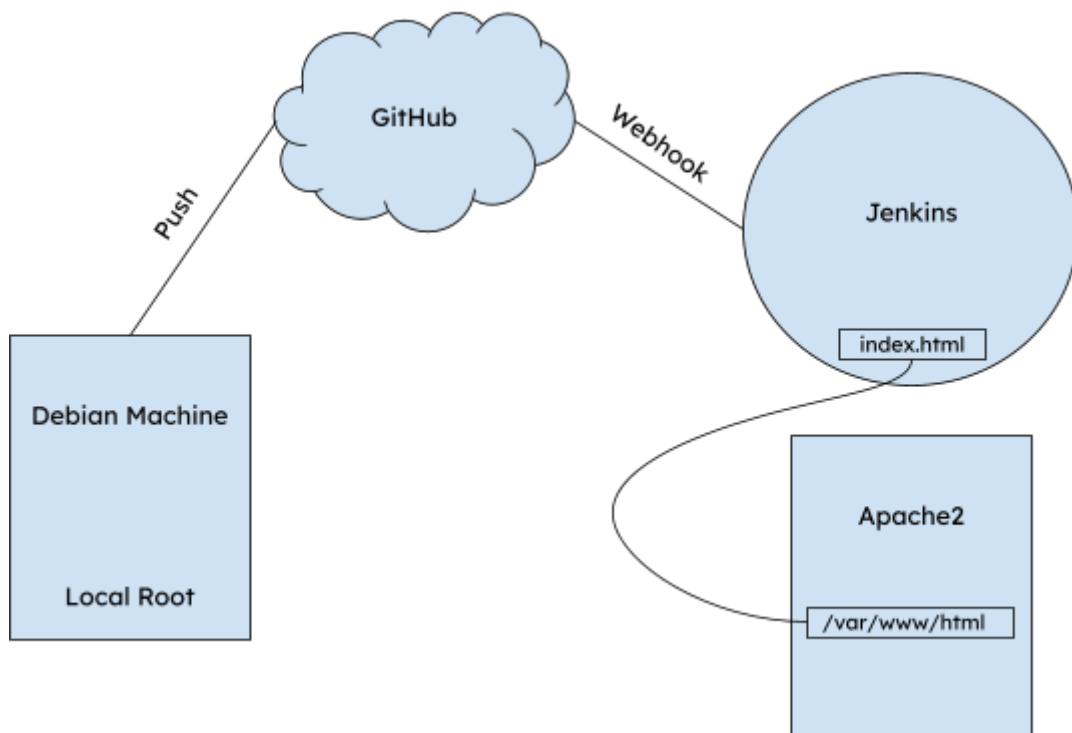
Name	Security group rule ID	Port range	Protocol	Source
-	sgr-04cd96a06213b176	22	TCP	0.0.0.0/0
-	sgr-099ddec8f07bade4f	80	TCP	0.0.0.0/0
-	sgr-0a1f41020f15e99b5	8080	TCP	0.0.0.0/16
-	sgr-04d2115f2e4709d46	443	TCP	0.0.0.0/0

- Open Jenkins Public IP in Browser: 3.238.69.100:8080
- Copy Red Path
`/var/lib/jenkins/secrets/initialAdminPassword`
- Cat : the path in deb 10

```
admin@ip-172-31-2-20:~$ sudo cat /var/lib/jenkins/secrets/initialAdminPassword
6801195dab6a4dc1a0b08e61fcc17c04
```

- Copy cat content
- Paste in : Admin Password Filed
- Install: Suggested Plugins
- Create First Admin User
 - Save and Continue
- Start USing jenkins
- Logout
- Stop EC2 Instance
- Verify Stoppage on instances: EC2 → Dashboard → Instances
→ Instance State → No Instance Should be there.

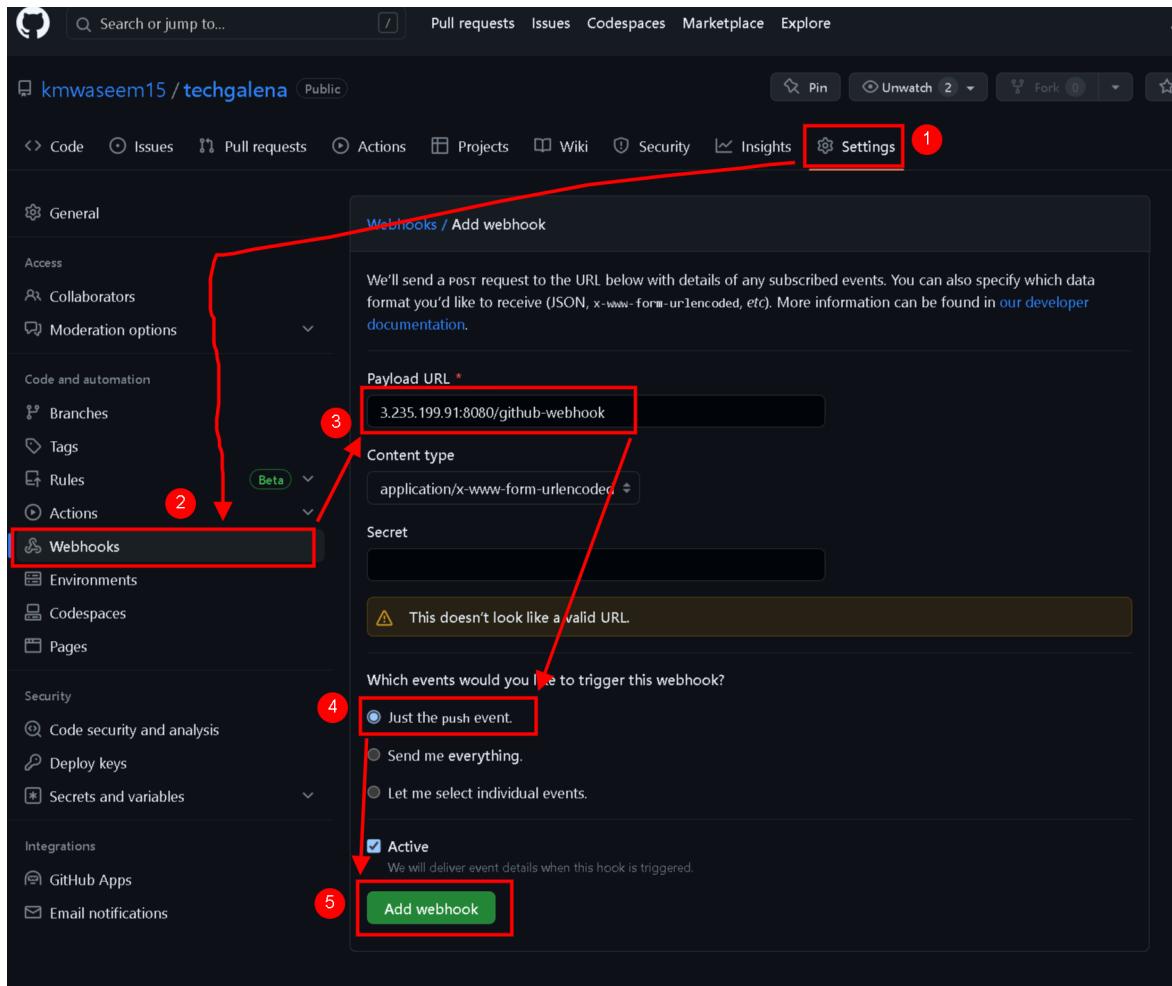
Jenkins Working With Github/Debian Machine For CI/CD :



Go to: Github → Your Repository → Settings → Webhook

→ Click on: Add webhook → Edit Payload URL: jenkins ip:808/github-webhook

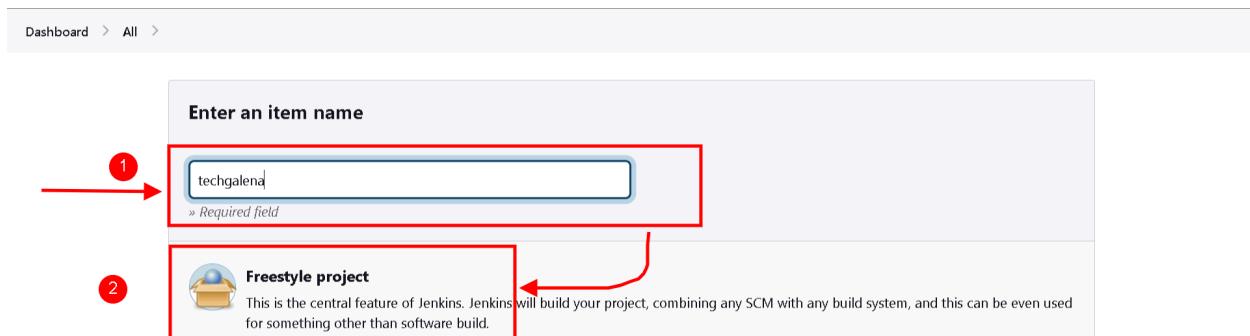
→ Scroll Below & Click on: Add webhook



Now go to: Jenkins Web Page → New Item



Enter : item name → Click on: Freestyle project → Click: OK



Enter: Description

The screenshot shows the Jenkins 'General' configuration page. On the left, there's a sidebar with 'Configure' and several tabs: 'General' (selected), 'Source Code Management', 'Build Triggers', and 'Build Environment'. The main area has a 'Description' section with a text input containing the text 'This is my first Jenkin project as a student'. A 'Enabled' toggle switch is at the top right, set to 'Enabled'.

Check: Github Project Enter Github Repository Link

The screenshot shows the Jenkins 'GitHub project' configuration page. It includes a 'Project url' input field with the value 'https://github.com/kmwaseem15/techgalena/'. There are tabs for 'General', 'Source Code Management' (selected), and 'Build Triggers'.

Install: git on Jenkins Machine Running on AWS

```
admin@ip-172-31-2-20:~$ sudo apt-get install git
```

Branch Name: */master

The screenshot shows the Jenkins 'GitHub project' configuration page. In the 'Branches to build' section, there's a 'Branch Specifier' input field containing the value '*/master'. Other tabs visible include 'General' and 'Source Code Management'.

Check: GitHub hook triggers | Check: Add timestamp to the Console Output

The screenshot shows the Jenkins 'Build Triggers' and 'Build Environment' configuration pages. Under 'Build Triggers', the 'GitHub hook trigger for GITScm polling' checkbox is selected. Under 'Build Environment', the 'Add timestamps to the Console Output' checkbox is selected. Other tabs like 'General', 'Source Code Management', 'Build Steps', and 'Post-build Actions' are also visible.

Go to: Build Steps → Expand Below Tab → Check: Execute Shell
→ Type Something

Configure

The screenshot shows the Jenkins 'Configure' screen for a project. On the left, there is a sidebar with icons for General, Source Code Management, Build Triggers, Build Environment, Build Steps (which is selected and highlighted in grey), and Post-build Actions. The main area is titled 'Build Steps' and contains a single step named 'Execute shell'. The 'Command' field contains the command 'echo "This command will be generated"'. There is also a link to 'See the list of available environment variables'.

Post build actions: → Expand Add post-build action: Select: Send e-mail Notification → Type your email

Configure

The screenshot shows the Jenkins 'Configure' screen for a project. On the left, there is a sidebar with icons for General, Source Code Management, Build Triggers, Build Environment, Build Steps, and Post-build Actions (which is selected and highlighted in grey). The main area is titled 'Post-build Actions' and contains a single action named 'E-mail Notification'. The 'Recipients' field contains the email address 'example@gmail.com'. There are two checkboxes: one checked for 'Send e-mail for every unstable build' and one unchecked for 'Send separate e-mails to individuals who broke the build'. At the bottom, there is a button labeled 'Add post-build action'.

Click on: Save

Save

Apply

CI/CD in Jenkins

Go to: Jenkins Machine running on AWS
Change “jenkins” user’s password

```
admin@ip-172-31-2-20:~$ sudo passwd jenkins
```

Switch to “jenkins” user

```
admin@ip-172-31-2-20:~$ sudo su jenkins
```

Generate “key-pair” for jenkins user

```
jenkins@ip-172-31-2-20:/home/admin$ ssh-keygen
```

Cat the generated “key-pair”

```
jenkins@ip-172-31-2-20:~$ cat /var/lib/jenkins/.ssh/id_rsa.pub
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQgQDjISQZYxRiAIvZzRGThJiSVCaE0Cj7lm708O7DDwMCN8o8WRXumnSf7wTFyJV
L+Cv3uFj1HmxLd8uS4rKh06aB9Se6n/BWBvPgPwqINafTKngEm52YGg1531Vyd2pDlk0v0z1DLqFq9USwyqCDPZC4ZTHalQ1Xmv
Xrzua3DKqU1J/7FVZ1i5e09ZfFEcoePV0S7ISqlpol8Ah06btDAFGJnBY71nIbz4x7h1G+BXu/1JNAr2Fm5UKVqcY/7goJdR52M
mHWx3vMqr2U/pV4s8VWKAoc1HtrJfy85GVingZTOFd/DNUwdv/1xynQG7cUKB5EsruNjR3zYONmDu03+Ndt9J0CV+U0F5fkDhk
ZgPt9dNVtBz1mu/kw2WHWt/9x7Ct0/kbmtf7Ce9vb4LZAOrhwy2B8k1B1jB4m85BjLrRm3Nd3pqJcaaqqk0lp5+1YuYSaVgJPy+m
y3XJTBsyU/vJNFbkJwL2ZCq5XRKRkTirFigDLyy1H2Wohs3/NMK7d4GM= jenkins@ip-172-31-2-20
```

Now copy this public key to: Web server's :

“/home/admin/.ssh/authorized_keys” path

```
GNU nano 5.4                               /home/admin/.ssh/authorized_keys
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQgH9422DFiUtg7ZSCtkLEuG8cWS6zcaBoNGdpd4N+iEJlIhek53SyFdXBew>
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQgQDjISQZYxRiAIvZzRGThJiSVCaE0Cj7lm708O7DDwMCN8o8WRXumnSf7wTFyJ>
```

Change permission of /var/www/ in web Server machine.

```
admin@ip-172-31-15-56:~$ sudo chmod -R 7777 /var/www/
```

Go to jenkins Your Project Configure :

Add : scp index.html admin@172.31.15.56:/var/www/html/



Click on: Save

Save

Apply

Now Come to: Jenkins Machine running on AWS (git)

Go to your git directory: Eg. techgalena

```
admin@ip-172-31-2-20:~$ cd techgalena/
```

Make a new: index.html file

```
admin@ip-172-31-2-20:~/techgalena$ sudo nano index.html
```

```
GNU nano 5.4                                         index.html *
<!DOCTYPE html>
<html>
<head>
<title>My Website</title>
<style>
```

```
admin@ip-172-31-2-20:~/techgalena$ git add index.html
```

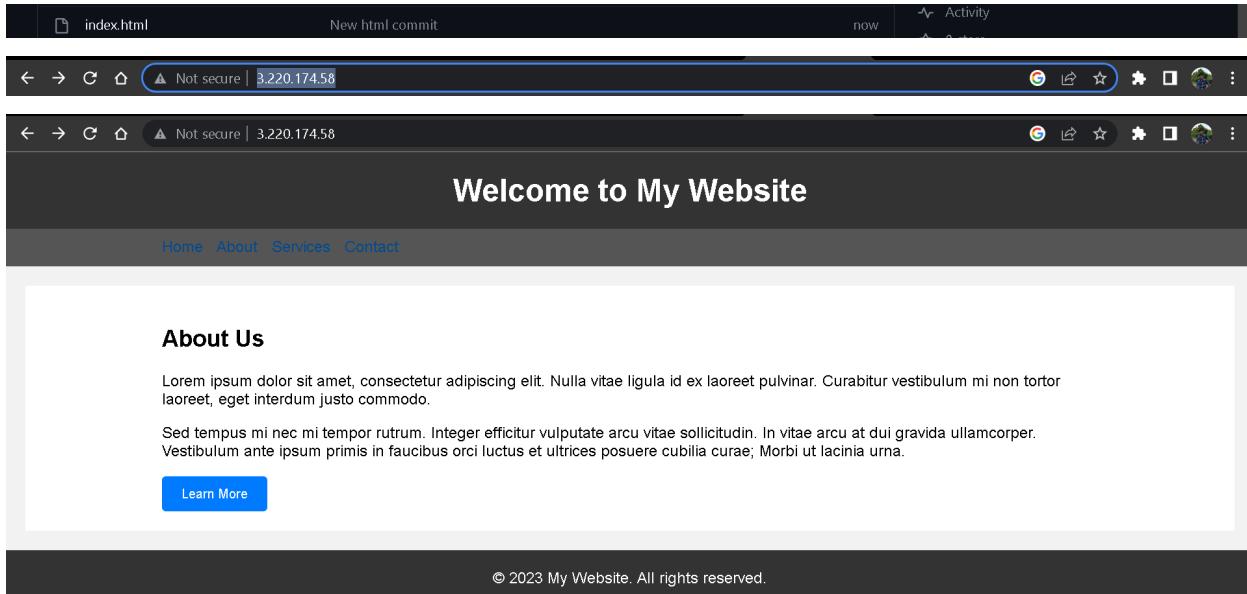
```
admin@ip-172-31-2-20:~/techgalena$ git commit -m "New html commit"
```

```
[master a0dd261] New html commit
```

```
1 file changed, 112 insertions(+), 3 deletions(-)
```

```
rewrite index.html (100%)
```

```
admin@ip-172-31-2-20:~/techgalena$ git push origin master
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 1.12 KiB | 1.12 MiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:kmwaseem15/techgalena.git
  3b0059b..a0dd261  master -> master
```



----- Jenkins End -----

Copy “index.html” file from Jenkins Machine To New Debian Based AWS EC2 Machine/var/www/html/

Create New EC2 Instance: Login on PuTTY & Install: Apache2

```
admin@ip-172-31-15-56:~$ sudo apt-get update
```

```
admin@ip-172-31-15-56:~$ sudo apt-get install apache2
```

Now: Remove the default “index.html” file to the path /var/www/html

```
admin@ip-172-31-15-56:~$ sudo rm /var/www/html/index.html
```

----- End -----

Virtual Private Cloud (AWS):

1. AWS → Name: ditiss-lab → Auto Generate → IP 172.30.0.0/16
→ No IPv6 → No. Of Availability Zones [Single Zone(1)]
→ No. of public subnets: 1 → No. of private subnets: 1
→ NAT Gateway: 1AZ → Click on: Create VPC → View VPC

Now Create Two EC2 Instance:

1. EC2 - 01

- Launch Instance → Name: Private
- OS: Debian
- Create a new key-pair (.pem): Name - private
- RSA → .pem
- Networking: Auto Assign Public IP Enable
 - Allow SSH Traffic from: Anywhere
 - Allow HTTP Traffic From Internet
 - Allow HTTPS Traffic From Internet
 - Subnets: private
- Launch Instance

2. EC2 - 02

- Launch Instance → Name: Public
- OS: Debian
- Create a new key-pair (.pem): Name - public
- RSA → .ppk
- Networking: ditiss-vpc
 - Allow SSH Traffic from: Anywhere
 - Subnets: public
 - Auto Assign IP: Enable
- Launch Instance
- Copy IP → Paste in PuTTY
- Login as passwordless SSH

nano key:

```
Paste private Key Of EC2 - 01 (i.e private.pem)
chmod 400 key
ssh -i key admin@Private IP of EC2 - 01
sudo apt-get install update
```

```
sudo apt-get install apache2
sudo rm /var/www/html/index.html
sudo nano /var/www/html/index.html
sudo apt-get install curl
curl (Private IP) → We will see private machine's browser
sudo apt-get install links
links (Private IP) → We will see private machine's browser
```

Remove VPC:

Terminate both Instances

Then Delete VPC Created

Will show 3 error red marks:

1. Remove NAT gateway At left side Actions Delete
2. EC2 Instance → Elastic IP's → Select all
→ Release Elastic IP address

There should be nothing in: 1. Route Table 2. Elastic IP 3. VPC 4. EC2 Instances

5. Total No. Of Active Regions: Not more than 1

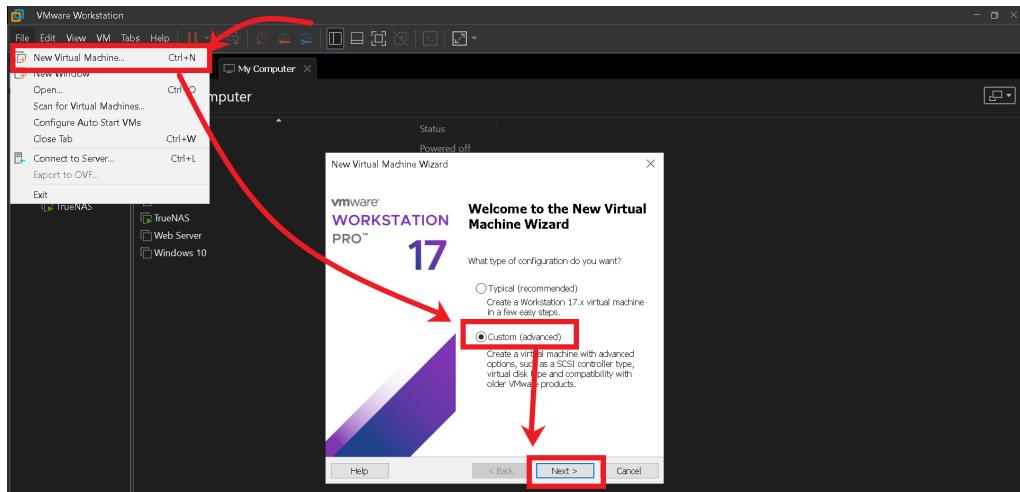
----- VPC End -----

Set Up ZFS Volume on TrueNAS:

1. Download TrueNAS .iso file from the Official Site.

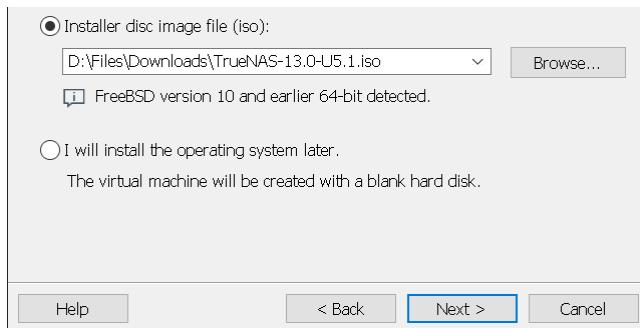


2. Open VMWare → Click on: New Virtual Machine → Custom → Next

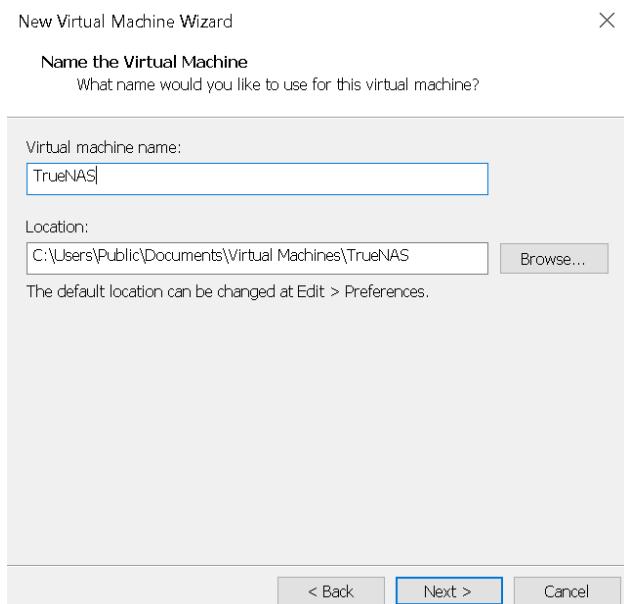


3. Next

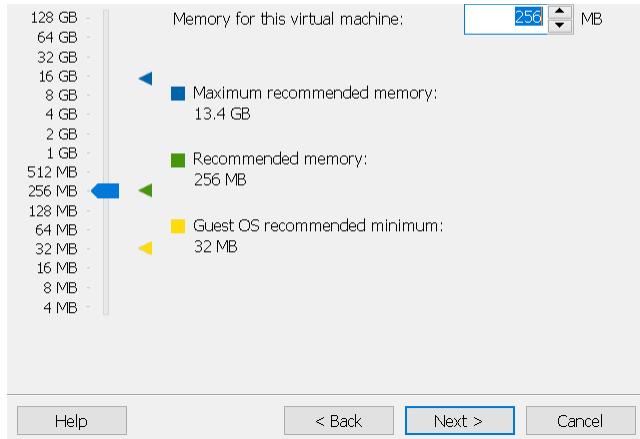
4. Select: TrueNAS.iso file. → Next



5. Virtual Machine Name: TrueNAS Location: C:\TrueNAS → Next

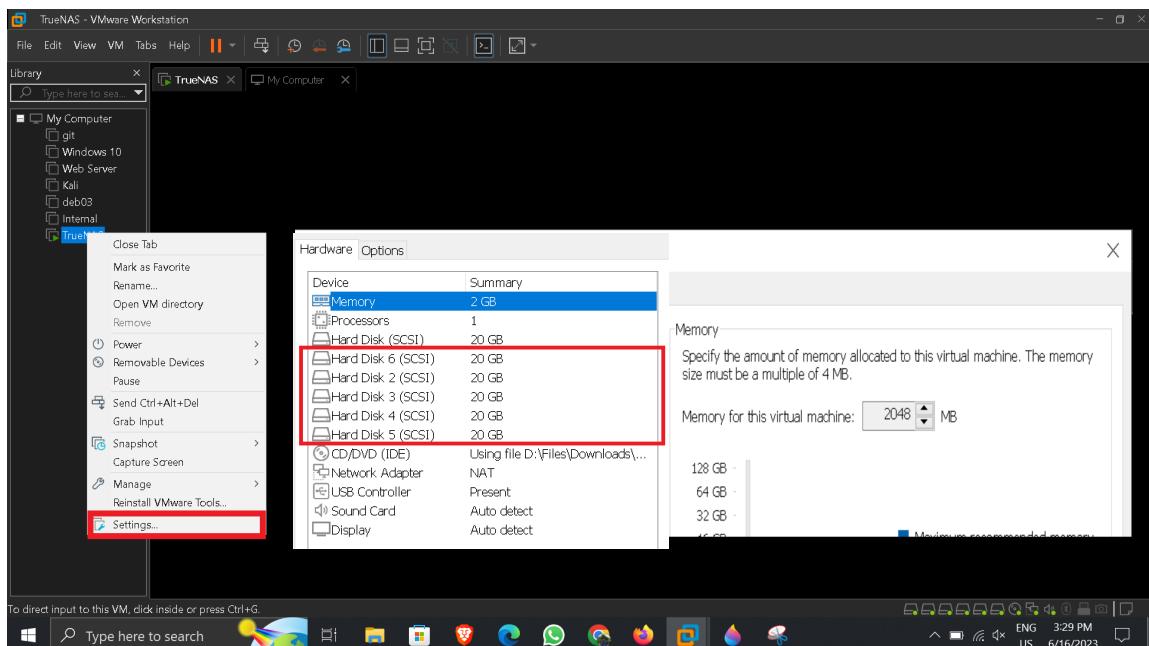


6. Next → RAM:2048 → Next

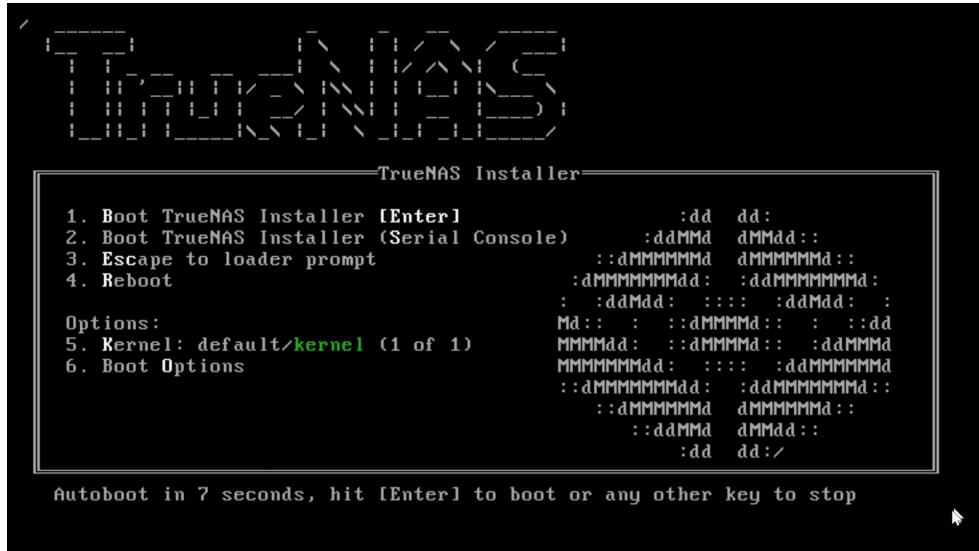


7. Next → Next → Next → Next → Next → Next → Finish

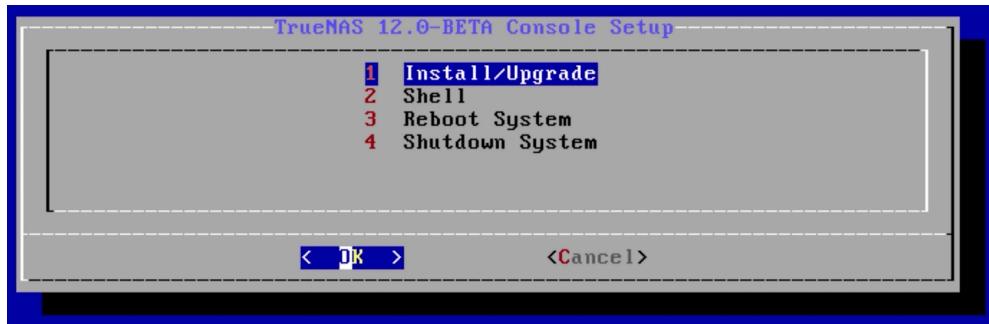
8. Right Click on: TrueNAS Settings Add 5 Disks (Size: > 5GB)



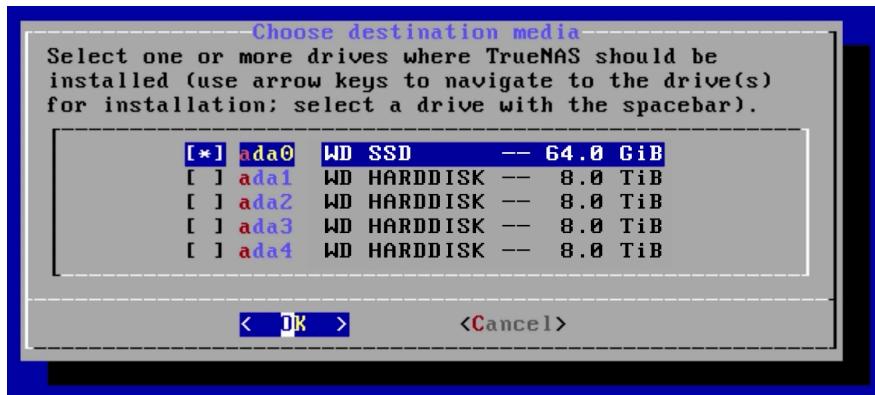
9. Power On: TrueNAS Machine → Wait for some time and boot will start



10. Select Option 1 Click: OK



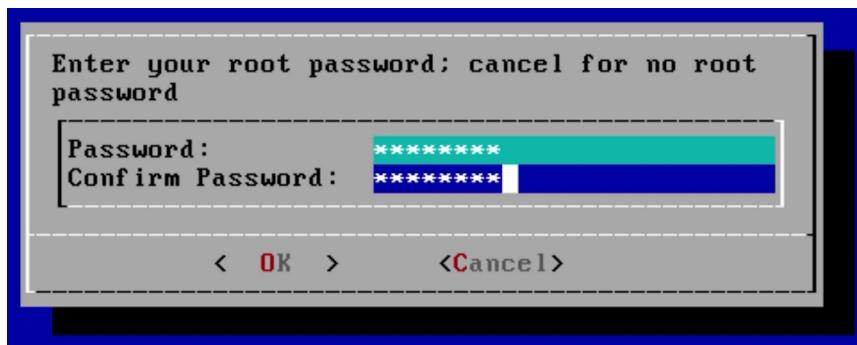
11. Select Disk [*Use Space Key To Select*] → Click: OK



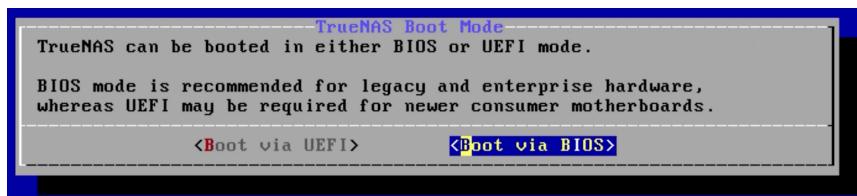
12. Select: Yes



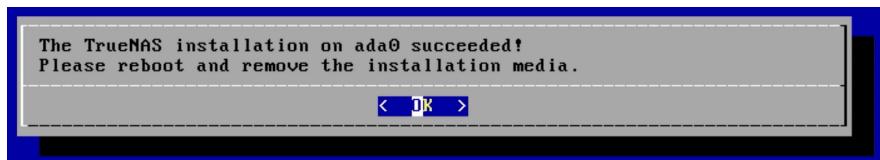
13. Assign Password. Click : OK



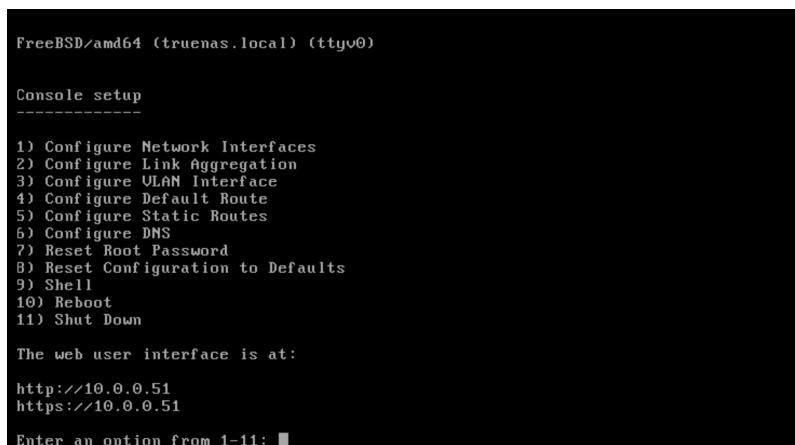
14. Select: Boot via BIOS



15. Wait for Installation & To Reboot Click: OK



16. After Reboot Configure IP.



17. IP configured.

```
FreeBSD/amd64 (truenas.local) (ttyv0)

Console setup
-----
1) Configure Network Interfaces
2) Configure Link Aggregation
3) Configure VLAN Interface
4) Configure Default Route
5) Configure Static Routes
6) Configure DNS
7) Reset Root Password
8) Reset Configuration to Defaults
9) Shell
10) Reboot
11) Shut Down

The web user interface is at:

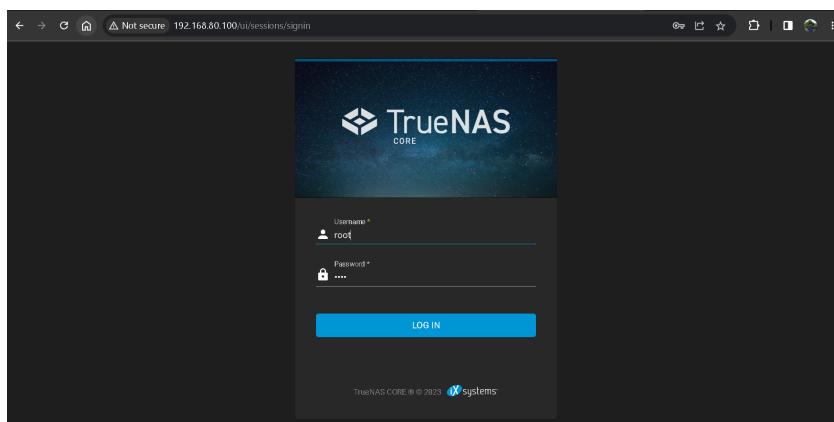
http://192.168.80.100
https://192.168.80.100

Enter an option from 1-11: █
```

18. Now go to browser: Enter: Configured IP [Eg. <http://192.168.80.100>]

Username: **root** | Password : What you configured on stage: 13 [Eg. **toor**]

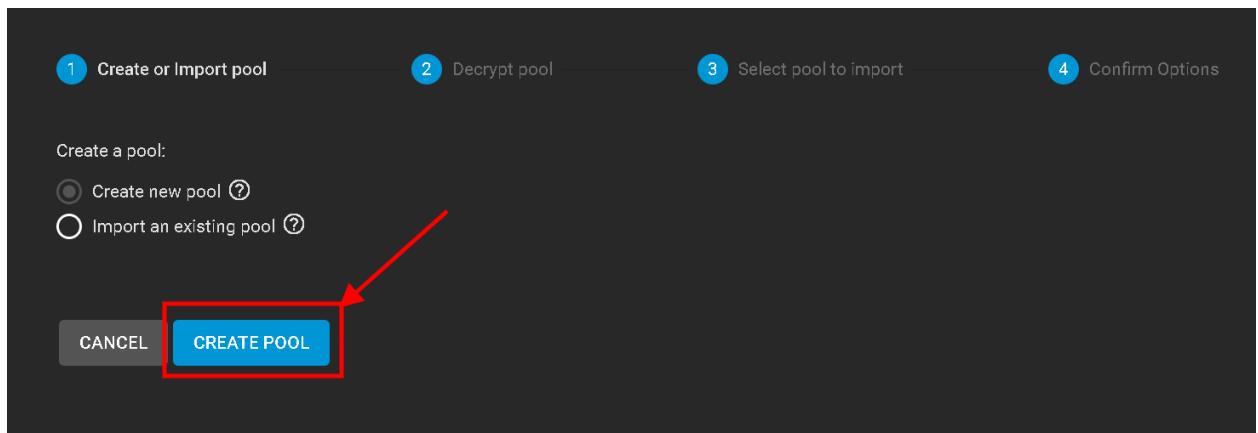
Press: Enter



19. Storage → Pool → ADD

The screenshot shows the TrueNAS CORE web interface. The left sidebar has 'Storage' selected (marked with a red box and number 1). Under 'Storage', 'Pools' is also highlighted (marked with a red box and number 2). In the main content area, there's a 'Pools' section with the message 'No pools'. To the right, there's a large red arrow pointing from the 'Pools' section towards a blue 'ADD' button, which is also enclosed in a red box with number 3.

20. Click: Create Pool



21. Provide a name for the pool, select the disk or disks you want to use for the pool, and choose the RAID level or redundancy mechanism (e.g., RAIDZ1, RAIDZ2, etc.).

The screenshots illustrate the process of creating a ZFS pool:

- Step 1:** Pool creation form. Name is set to "zfs-kmw". An "Encryption" checkbox is present but unchecked.
- Step 2:** Warning message: "Warning: There are 5 disks available that have non-unique serial numbers. Non-unique serial numbers can be caused by a cabling issue and adding such disks to a pool can result in lost data." A checkbox labeled "Show disks with non-unique serial numbers" is checked.
- Step 3:** Disk selection table. Five disks (da1, da2, da3, da4, da5) are listed and all are selected (indicated by checked checkboxes in the first column).
- Step 4:** Confirmation table. Shows "No data to display" and "0 selected / 0 total".

A screenshot of the TrueNAS Core interface under the Storage section. On the left, a sidebar lists System, Tasks, Network, Storage (selected), Pools, Snapshots, and VMware-Snapshots. The main area shows a table of disks: da1, da2, da3, da4, and da5. All five disks are checked. A red box highlights the disk selection table.

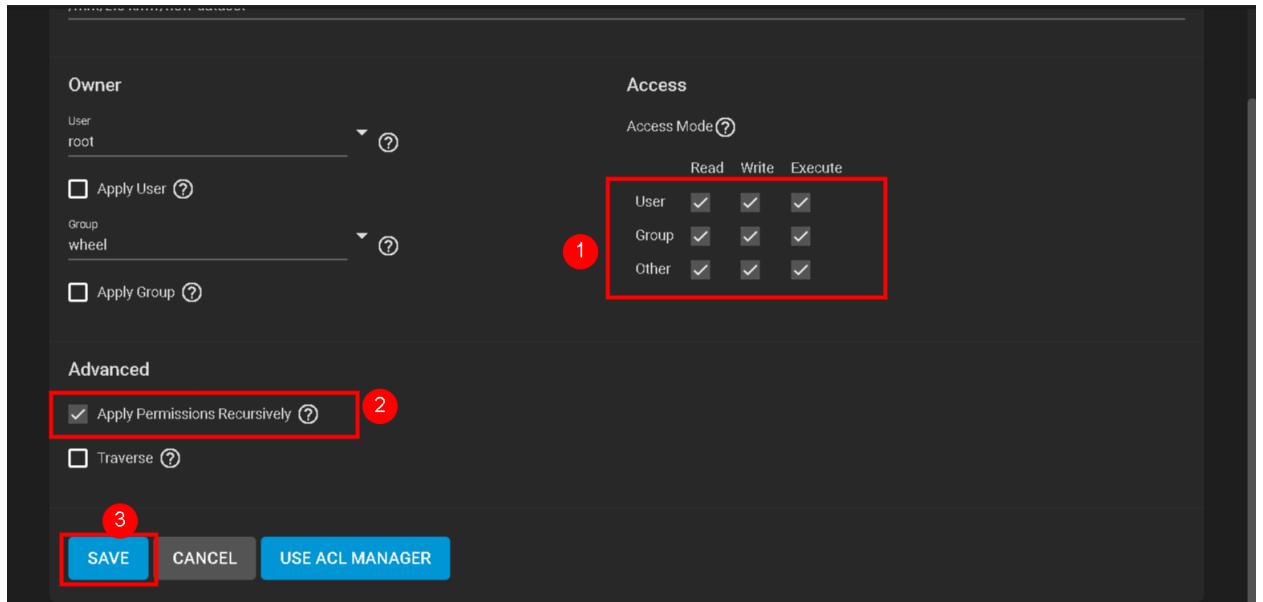
A screenshot of the TrueNAS Core interface showing the 'Mirror' pool configuration dialog. It displays 'Estimated raw capacity: 18 GiB'. Below the dialog, the 'Disks' tab is selected in the sidebar. A red box highlights the 'CREATE' button, which is circled with number 2. Above the dialog, a red box highlights the 'Mirror' configuration itself, which is circled with number 1.

A screenshot of a 'Warning' dialog box. It contains the text: 'The contents of all added disks will be erased.' Below this, there are two buttons: 'Confirm' (circled with number 1) and 'CANCEL'. To the right of the 'Confirm' button is another button labeled 'CREATE POOL' (circled with number 2). A red box highlights the 'CREATE POOL' button.

A screenshot of the TrueNAS Core interface showing the 'Storage / Pools' page. It lists a single pool: 'zfs-kmw (System Dataset Pool)' with status 'ONLINE'. The table columns include Name, Type, Used, Available, Compression, Compression Ratio, Readonly, Dedup, and Con. On the right, a context menu is open over the pool entry, with 'Add Dataset' highlighted by a red box. Other options in the menu include Add Zvol, Edit Options, Edit Permissions, User Quotas, Group Quotas, and Create Snapshot.

A screenshot of the 'Name and Options' dialog for creating a new dataset. The 'Name*' field is filled with 'new-dataset' and is highlighted by a red box. The 'Type' dropdown is set to 'FILESYSTEM'. The 'Compression' dropdown is set to 'lz4'. The 'Compression Ratio' dropdown is set to '19.23'. The 'Readonly' checkbox is unchecked. The 'Dedup' dropdown is set to 'OFF'. The 'Con' dropdown is set to 'No Comment'. A red box highlights the 'new-dataset' input field.

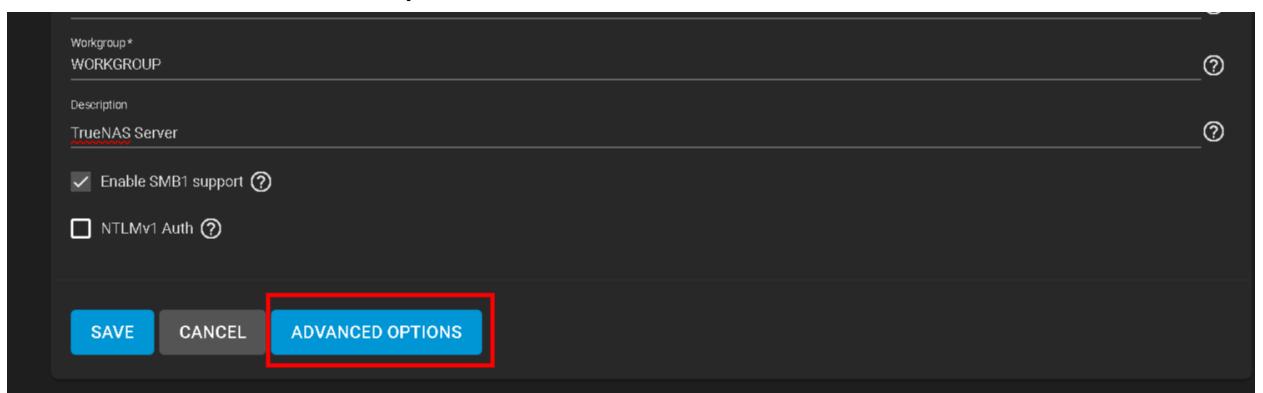
A screenshot of the TrueNAS Core interface showing the 'Storage / Pools' page. It lists the 'zfs-kmw' pool and a new dataset 'new-dataset'. The table columns are the same as the previous screenshot. On the right, a context menu is open over the 'new-dataset' entry, with 'Edit Permissions' highlighted by a red box. Other options in the menu include Add Zvol, Edit Options, User Quotas, Group Quotas, Delete Dataset, and Create Snapshot.



22. Now Go To Services: Enable SMB



23. Here: Click on “Advance Option”



Other Options

UNIX Charset: UTF-8 (1)

Log Level: Minimum (2)

Use Syslog Only (3)

Local Master

Enable Apple SMB2/3 Protocol Extensions

Administrators Group

Guest Account: nobody

File Mask: 777

Directory Mask: 777

Bind IP Addresses

Auxiliary Parameters

SAVE CANCEL BASIC OPTIONS

Sharing / SMB

TrueNAS CORE® © 2023 - iXsystems, Inc.

Samba

Filter Samba

ADD (3)

Name	Path	Description	Enabled
No data to display			

Sharing (1)

Apple Shares (AFP) (2)

Block Shares (iSCSI)

Unix Shares (NFS)

WebDAV Shares

Windows Shares (SMB)

Basic

Path: /mnt/zfs-kmw/new-dataset (1)

Name: new-dataset

Purpose: Default share parameters

Enabled (2)

SUBMIT CANCEL ADVANCED OPTIONS

Sharing

Apple Shares (AFP)

Block Shares (iSCSI)

Unix Shares (NFS)

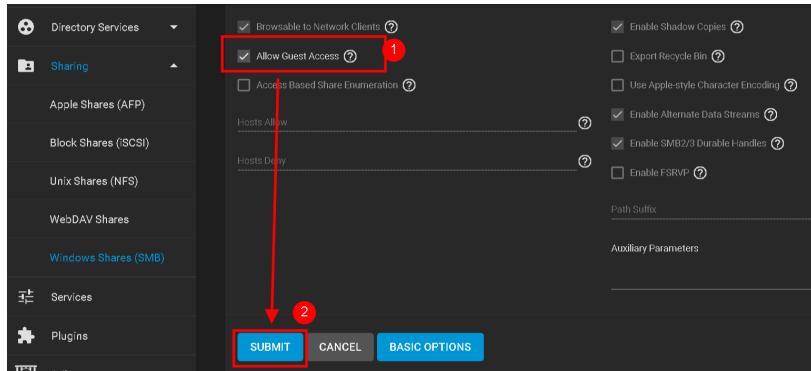
WebDAV Shares

Windows Shares (SMB)

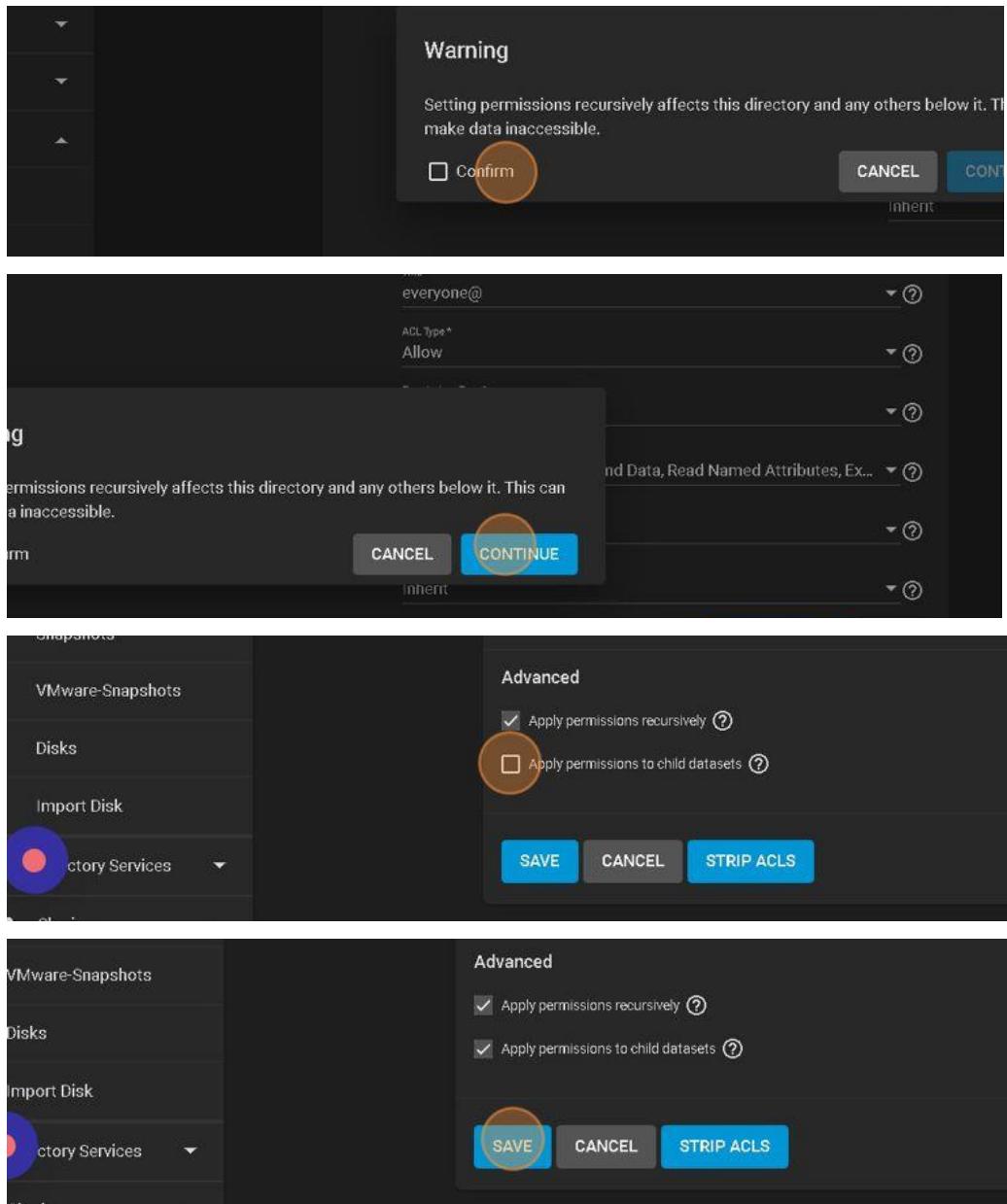
Services

Plugins

Jails



After this it will ask you to go to the ACL: Check : Confirm



3. Remove NAT gateway At left side Actions Delete
4. EC2 Instance → Elastic IP's → Select all
→ Release Elastic IP address

----- TrueNAS End -----

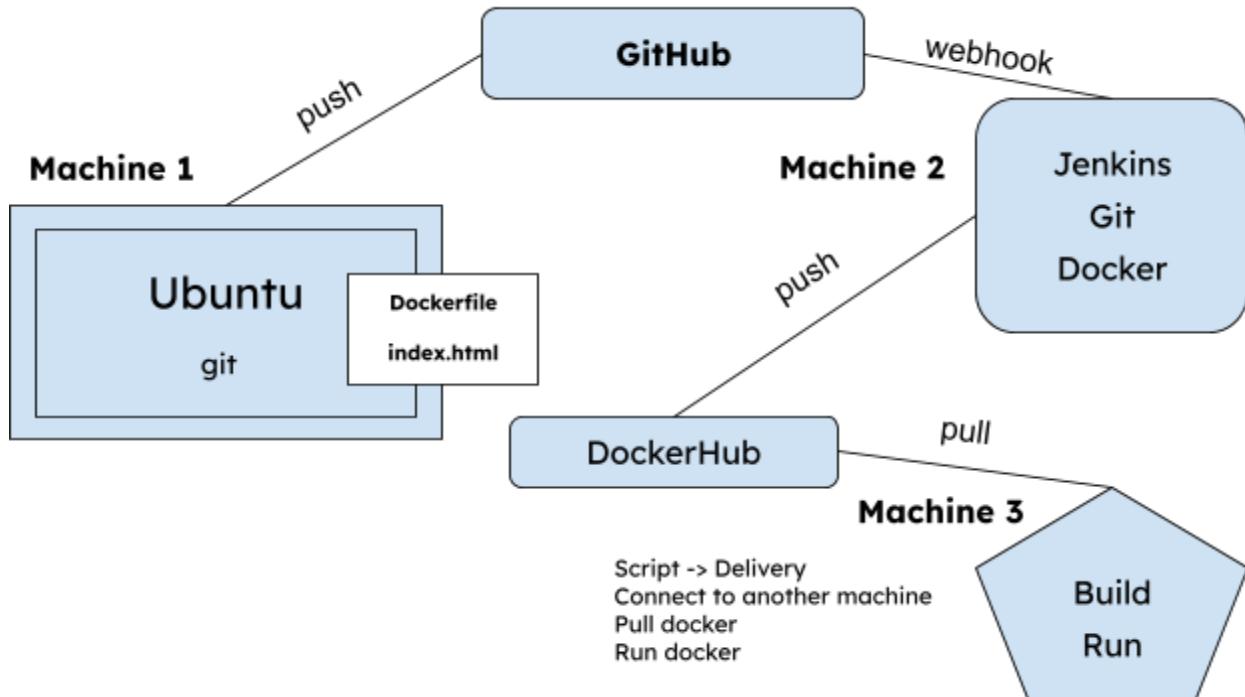
Docker Installation:

```
sudo apt-get update  
sudo apt-get install docker.io  
  
sudo docker --version  
  
sudo docker run hello-world  
sudo docker  
  
sudo docker images  
sudo docker ps -a  
  
sudo docker pull centos  
sudo docker run -it centos  
  
sudo docker rmi -f hello-world
```

```
sudo docker pull httpd  
  
mkdir web  
cd web  
nano index.html  
    <h1>Welcome To My Webpage</h1>  
    <h3>Thank You..!</h3>  
  
nano Dockerfile  
    FROM httpd:latest  
    COPY index.html /usr/local/apache2/htdocs  
    EXPOSE 80  
  
docker build -t httpd .  
docker run -d -p 8080:80 httpd
```

----- Docker End -----

Local → GitHub → Jenkins on EC2 → DockerHub → EC2:



Local Machine 1:

```
apt-get update
apt-get install git
mkdir techgalena
cd techgalena
root@debian:~/techgalena# ls -l
total 0

nano index.html
|- <h1>  !..Nginx Via Jenkins Through GitHub & Docker...!</h1>

nano Dockerfile
|- FROM nginx:latest
|- COPY index.html /usr/share/nginx/html/

root@debian:~/techgalena# ls -l
total 8
-rw-r--r-- 1 root root 58 Jun 20 06:55 Dockerfile
```

```
-rw-r--r-- 1 root root 61 Jun 20 06:55 index.html
git add .
git commit -m "message"
git push origin master
```

Make a webhook between GitHub & Jenkins

JENKINS, Machine 2:

```
apt-get update
apt-get install git
apt-get install docker.io
apt-get install jenkins

ssh-keygen
cat /root/.ssh/id_rsa.pub

// Copy this key and paste it in Machine 3
at "~/.ssh/authorized_keys" //

sudo usermod -aG docker admin
sudo usermod -aG docker jenkins
sudo chmod 666 /var/run/docker.sock
sudo chown root:docker /var/run/docker.sock
sudo service jenkins restart
sudo service docker restart
```

Machine 3:

Install: docker.io

```
sudo chmod 666 /var/run/docker.sock
sudo chown nobody:nogroup /var/run/docker.sock
sudo chmod 700 ~/.ssh
sudo chmod 600 ~/.ssh/authorized_keys

sudo usermod -aG docker admin
sudo usermod -aG docker root
```

Open Jenkins in Browser on port 8080:

The screenshot shows the Jenkins interface for creating a new item. A red box highlights the 'New Item' button in the top-left corner. Below it, a search bar and navigation links are visible. The main area is titled 'Enter an item name' with a required field for 'Project'. A red arrow points to the 'Project' input field, labeled '1'. Another red arrow points to the 'Pipeline' option, labeled '2'. A third red arrow points to the 'OK' button at the bottom right, labeled '3'.

Enter an item name

Project *(Required field)*

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.

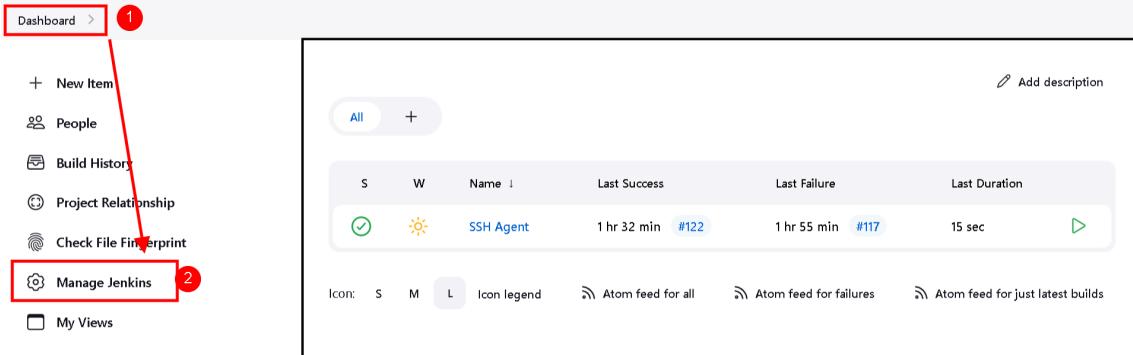
Organization Folder
Creates a set of multibranch project subfolders by scanning for repositories.

If you want to create a new item from other existing, you can use this option:

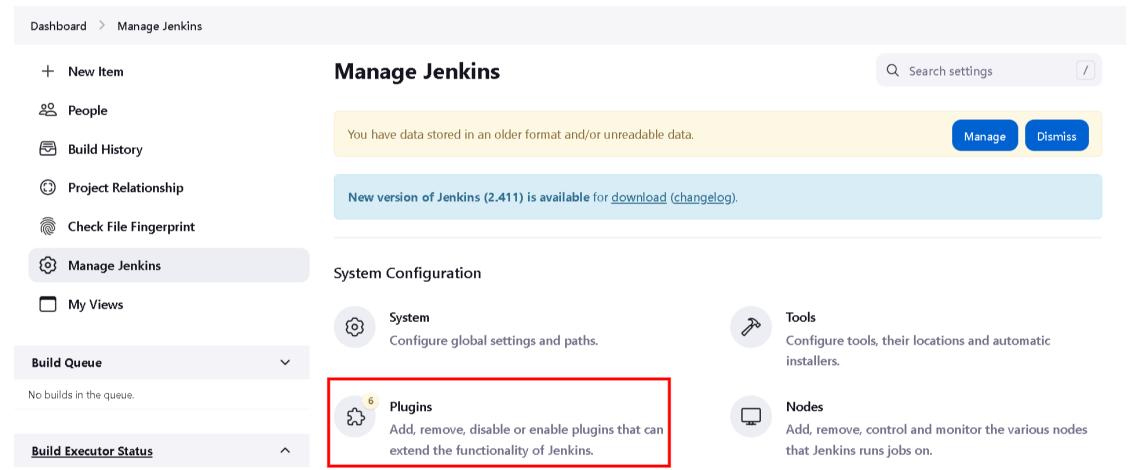
Copy from
Type to autocomplete

OK

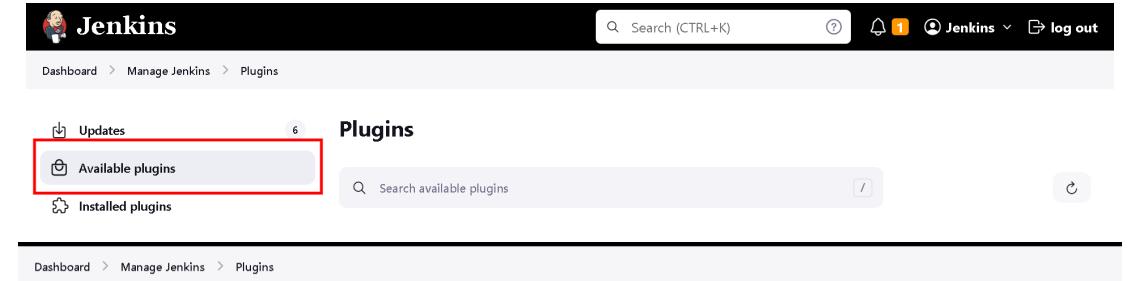
Install 2 Plugins: 1. Docker Pipeline 2. SSH Agent



The screenshot shows the Jenkins Dashboard. A red box highlights the 'Manage Jenkins' link in the sidebar, and a red arrow points from it to the 'Manage Jenkins' link in the main content area. The main content area displays a table of build status for various projects, with a 'SSH Agent' project highlighted.



The screenshot shows the 'Manage Jenkins' page under 'System Configuration'. A red box highlights the 'Plugins' section, which contains a sub-section titled 'Available plugins'. This section lists 6 available plugins, with 'Docker Pipeline' being the first item.



The screenshot shows the 'Available plugins' page for the 'Docker Pipeline' plugin. A red box highlights the search bar at the top, which contains the text 'docker pipe'. A red arrow points from the search bar to the plugin's title 'Docker Pipeline'. Another red arrow points from the plugin's title to its description box. A third red arrow points from the description box to the 'Install without restart' button at the bottom.

Dashboard > Manage Jenkins

Manage Jenkins

You have data stored in an older format and/or unreadable data.

New version of Jenkins (2.411) is available for download ([changelog](#)).

System Configuration

- System**: Configure global settings and paths.
- Tools**: Configure tools, their locations and automatic installers.
- Nodes**: Add, remove, control and monitor the various nodes that Jenkins runs jobs on.

Plugins

Add, remove, disable or enable plugins that can extend the functionality of Jenkins.

Updates: 6

Available plugins: 6

Installed plugins: 0

Search available plugins:

Docker Pipeline 563.vd5d2e5c4007f

pipeline DevOps Deployment docker

Released: 6 mo 23 days ago

This plugin is up for adoption! We are looking for new maintainers. Visit our [Adopt a Plugin](#) initiative for more information.

Install without restart Download now and install after restart

The screenshot shows the Jenkins Plugins page. A red box highlights the 'Available plugins' section, which contains a search bar with 'ssh agent'. Below it, a specific plugin entry for 'SSH Agent' is shown with a red box around its title and a red arrow pointing to the 'Install' button. The 'Install without restart' button is also highlighted with a red box.

Configure Credentials for: Docker Login

The screenshot shows the Jenkins Manage Jenkins page. A red box highlights the 'Manage Jenkins' link in the left sidebar. A red arrow points from this link to the 'Security' section in the main content area, which is also highlighted with a red box.

Stores scoped to Jenkins

The screenshot shows the Jenkins Security page. A red box highlights the 'System' store selection under 'Store'. A red arrow points to the '(global)' link in the 'Domains' section, which is also highlighted with a red box.

The screenshot shows the Jenkins Global credentials (unrestricted) page. A red arrow points to the '+ Add Credentials' button, which is highlighted with a red box.

New credentials

Kind

Username with password 1

Scope 2

Global (Jenkins, nodes, items, all child items, etc)

Username 3

docker_username

Treat username as secret ?

Password 4

ID 5

992040

Description 6

Enter Any Number/Char of Your Choice

Create 6

Configure Credentials for: SSH

Dashboard > 1

- + New Item
- People
- Build History
- Project Relationship
- Check File Fingerprint
- Manage Jenkins 2
- My Views

All + Add description

S	W	Name	Last Success	Last Failure	Last Duration
✓	✖	SSH Agent	1 hr 32 min #122	1 hr 55 min #117	15 sec

Icon: S M L Icon legend Atom feed for all Atom feed for failures Atom feed for just latest builds

Dashboard > Manage Jenkins

Security 3

Secure Jenkins; define who is allowed to access/use the system.

Credentials 4

Configure credentials

Stores scoped to Jenkins

P Store I Domains

System 5 Click On (global)

(global)

Icon: S M L

New credentials

Kind SSH Username with private key 1

Scope Global (Jenkins, nodes, items, all child items, etc) 2

ID 45678 3

Description

Username root 4

Treat username as secret

Private Key Enter directly 5

Key No Stored Value 6 Add 7

Key Enter New Secret Below -----BEGIN OPENSSH PRIVATE KEY-----
b3B1bnNzaC1rZXktdjEAAAAABG5vbmlJAAAEBm9uZQAAAAAAAAABAAAB1wAAAAdzc2gtcn
NhAAAAAwEAAQAAAYEAmA1M4TFZiQLm37DB6PWh60kMxF5ObLsP0r53alW03kV7R7sgoaJhzi 8

Passphrase

Create

General

Enabled

Description
nginx project

Plain text: [Preview](#)

Discard old builds ?

Do not allow concurrent builds

Do not allow the pipeline to resume if the controller restarts

GitHub project

Project url ?

[Advanced](#) ▾

Pipeline speed/durability override ?

Preserve stashes from completed builds ?

This project is parameterized ?

Throttle builds ?

Build Triggers

Build after other projects are built ?

Build periodically ?

GitHub hook trigger for GITScm polling ?

Poll SCM ?

Pipeline

Definition

Pipeline script

Script ?

```
1 pipeline {
2     agent any
3
4     stages {
5         stage('Checkout') {
6             steps {
7                 echo "Performing Checkout"
8                 checkout([$class: 'GitSCM',
9                           branches: [[name: 'master']],
10                          userRemoteConfigs: [[url: 'https://github.com/kmwaseem15/techgalena.git']]])
11             }
12         }
13
14         stage('Build Docker Image') {
15             steps {
16                 echo "Building Docker Image"
17             }
18     }
19 }
```

Use Groovy Sandbox ?

[Pipeline Syntax](#)

[Save](#) [Apply](#)

JENKINS Pipeline Stages:

```
pipeline {
    agent any

    stages {
        stage('Checkout') {
            steps {
                echo "Performing Checkout"
                checkout([$class: 'GitSCM',
                          branches: [[name: 'master']],
                          userRemoteConfigs: [[url:
'https://github.com/onkgithub/iacsd.git']]])
            }
        }

        stage('Build and Push to Docker Hub') {
            steps {
                script {
                    // Build Docker image
                    sh 'docker build -t 9766791539/nginx .'

                    // Log in to Docker Hub using credentials
                    // Here credentialsID: 'ID You Gave During Docker Login Configuration (992040)'
                    withDockerRegistry(credentialsId: '992010') {
                        // Push Docker image to Docker Hub
                        sh 'docker push 9766791539/nginx'
                    }
                }
            }
        }

        stage('Pull on Remote Machine') {
            steps {
                sshagent(['992040']) {
                    // SSH into the remote machine and pull the Docker
                    image
                    // Here credentialsID: 'ID You Gave SSH Configuration (45678)'
                    sh 'ssh admin@54.147.9.237 "docker pull
9766791539/nginx"'
                }
            }
        }
    }
}
```

```
        }
        stage('Kill Running Containers') {
            steps {
                // Here sshagent(['ID You Gave During SSH Configuration (45678)'])
                sshagent(['992040']) {
                    // SSH into the remote machine and kill all
                    running containers
                    sh 'ssh admin@54.147.9.237 "docker kill \$(ssh
                    admin@54.147.9.237 docker ps -q)"'
                }
            }
        }

        stage('Build and Run on Remote Machine') {
            steps {
                // Here sshagent(['ID You Gave During SSH Configuration (45678)'])
                sshagent(['992040']) {
                    // Copy the Dockerfile to the remote machine
                    sh 'ssh admin@54.147.9.237 "cd /home/admin/docker"'
                    sh 'scp /var/lib/jenkins/workspace/docker/Dockerfile
                    admin@54.147.9.237:/home/admin/docker'
                    sh 'scp /var/lib/jenkins/workspace/docker/index.html
                    admin@54.147.9.237:/home/admin/docker'
                    // SSH into the remote machine and build/run the Docker
                    image/container
                    sh 'ssh admin@54.147.9.237 "cd /home/admin/docker &&
                    docker build -t myapp . && docker run -d -p 8080:80 myapp"'
                }
            }
        }
    }
}
```

**After setting up all the machines: Got to Machine 1:
Push all files for first time.**

```
git add .
git commit -m "message"
git push origin master
```

From Second times onward you will only have to push index.html only.

```
git add index.html  
git commit -m "message"  
git push origin master
```

After Doing Above Things on Machine 1

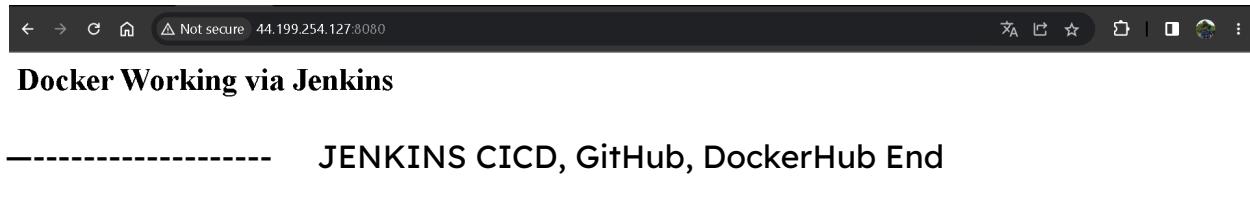
Your index.html content should be visible on Machine 3's IP in browser on port 8080

Input:



```
shuhari@debian: ~  
GNU nano 3.2 index.html Modified  
<h1> Docker Working via Jenkins </h1>
```

Output:



```
Not secure 44.199.254.127:8080  
Docker Working via Jenkins  
----- JENKINS CICD, GitHub, DockerHub End -----
```

Docker Swarm :

Docker Swarm is a container orchestration tool for managing multiple containers across multiple hosts. Its key benefit is high scalability and fault tolerance. It enables automatic load balancing, service discovery, and self-healing capabilities, ensuring reliable and efficient deployment of containerized applications. With Swarm, users can easily scale their applications up or down based on demand, providing flexibility and resilience in containerized environments.

```
docker swarm init  
docker swarm join --token //Run the command in node machines  
docker node ls  
docker service create -name cdac -p 80:80 -replicas 3 iacsd  
docker service
```

```
docker service rm cdac
```

Services Of AWS:

→ EC2:

Amazon Elastic Compute Cloud (Amazon EC2) is a web service provided by Amazon Web Services (AWS) that allows you to create and manage virtual servers in the cloud. EC2 provides scalable computing capacity, enabling you to quickly provision instances and scale them up or down based on your requirements.

Here are some key features and concepts related to EC2:

1. **Instances:** EC2 instances are virtual servers in the cloud. You can choose from a variety of instance types based on your workload requirements, such as compute-optimized, memory-optimized, storage-optimized, GPU instances, etc.
2. **Regions and Availability Zones:** EC2 is available in multiple geographic regions worldwide. Each region consists of multiple isolated locations called Availability Zones. You can choose the region and availability zone(s) where you want to deploy your EC2 instances.
3. **Amazon Machine Images (AMIs):** An AMI is a pre-configured template that contains the necessary software, operating system, and configuration settings to launch an EC2 instance. You can select a public AMI provided by AWS or create your own custom AMI.
4. **Instance Types:** EC2 provides various instance types optimized for different use cases. These instance types differ in terms of CPU, memory, storage, and networking capacity. You can choose the most suitable instance type based on your workload requirements.
5. **Security Groups:** Security groups act as virtual firewalls for your EC2 instances. They control inbound and outbound traffic by specifying rules that allow or deny traffic based on protocols, ports, and IP addresses.
6. **Elastic IP Addresses:** An Elastic IP address is a static, public IPv4 address that you can allocate and associate with your EC2 instances. It allows you to retain the same IP address even if you stop or terminate the instance.

7. **Elastic Block Store (EBS):** EBS provides persistent block-level storage volumes for EC2 instances. You can attach EBS volumes to your instances and use them as primary storage or additional storage for your data.
8. **Auto Scaling:** Auto Scaling allows you to automatically adjust the number of EC2 instances based on application demand. It helps you maintain performance, availability, and cost optimization by dynamically scaling your instances.
9. **Load Balancing:** EC2 offers load balancing services such as Elastic Load Balancing (ELB) to distribute incoming traffic across multiple EC2 instances. This enhances fault tolerance and ensures high availability for your applications.
10. **Integration with Other AWS Services:** EC2 integrates with various other AWS services, such as Amazon S3 for object storage, Amazon RDS for managed databases, Amazon VPC for networking, AWS CloudFormation for infrastructure as code, etc.

→ **AWS S3:**

AWS S3 or Amazon Simple Storage Service is a highly scalable and durable cloud storage service provided by Amazon Web Services (AWS). It allows you to store and retrieve large amounts of data, such as files, documents, images, videos, and backups, over the internet. S3 is designed to provide 99.999999999% (11 nines) durability for objects and 99.99% availability.

Key Features of AWS S3:

1. **Storage Classes:** S3 offers multiple storage classes optimized for different use cases, such as Standard, Intelligent-Tiering, Glacier, and Glacier Deep Archive. Each storage class has its own pricing and retrieval characteristics, allowing you to choose the appropriate class based on your data access requirements and cost considerations.
2. **Scalability and Durability:** S3 is designed to scale to accommodate virtually unlimited amounts of data. It automatically replicates data across multiple availability zones within a region to ensure durability. This redundancy provides high data durability and protects against hardware failures.

3. **Security and Access Control:** S3 allows you to manage access to your data through various mechanisms. You can use AWS Identity and Access Management (IAM) to control user permissions, create access policies, and manage encryption keys. S3 also supports server-side encryption for data at rest.
4. **Versioning and Lifecycle Policies:** S3 provides versioning capabilities, allowing you to preserve, retrieve, and restore every version of every object in your bucket. Lifecycle policies enable you to define rules to automatically transition or expire objects based on their age, which helps optimize storage costs.
5. **Data Transfer and Transfer Acceleration:** AWS offers multiple options to transfer data into and out of S3. You can use direct uploads, AWS Snowball devices for large-scale offline data transfer, or AWS DataSync for automated data transfer. Additionally, S3 Transfer Acceleration uses CloudFront's global network to speed up data transfers.
6. **Event Notifications and Logging:** S3 can generate event notifications to trigger actions or workflows when certain events occur, such as object creation or deletion. You can configure event notifications to integrate with other AWS services like AWS Lambda, Amazon SNS, or Amazon SQS. S3 also provides server access logs for auditing and troubleshooting.
7. **Integration with AWS Ecosystem:** S3 integrates with various AWS services, making it a foundation for many cloud-based solutions. For example, you can use it with AWS Lambda for serverless computing, Amazon Athena for querying data using SQL, or Amazon Redshift for data warehousing.

AWS S3 is widely used by individuals and organizations for various purposes, including website hosting, data backups, media storage and distribution, big data analytics, content delivery, and archival storage. Its flexibility, scalability, and robustness make it a popular choice for managing and storing data in the cloud.

→ **AWS Lambda:**

AWS Lambda is a serverless computing service provided by Amazon Web Services (AWS). It allows you to run your code without provisioning or managing servers. With Lambda, you can execute your code in

response to events such as changes to data in an Amazon S3 bucket, updates to a DynamoDB table, or even HTTP requests.

Key Features of AWS Lambda:

- 1. Serverless Execution:** Lambda abstracts the underlying infrastructure, allowing you to focus on writing and deploying code without worrying about server management.
- 2. Event-Driven Model:** Lambda functions are triggered by events such as changes to data, time-based schedules, or custom events generated by other AWS services.
- 3. Scalability:** AWS Lambda automatically scales your code in response to incoming requests. It can handle a few requests per day to thousands of requests per second, based on demand.
- 4. Pay-per-Use Pricing:** With Lambda, you only pay for the compute time consumed by your code. Billing is based on the number of requests and the time it takes to execute your code.
- 5. Language Support:** AWS Lambda supports multiple programming languages, including JavaScript (Node.js), Python, Java, C#, PowerShell, and Ruby. You can choose the language that best suits your needs.
- 6. Integration with AWS Services:** Lambda can integrate with other AWS services, such as Amazon S3, DynamoDB, API Gateway, SNS, SQS, and more. This allows you to build serverless architectures and microservices easily.
- 7. Versioning and Aliases:** Lambda allows you to version your functions, enabling you to deploy updates without impacting the existing production environment. Aliases let you point to specific versions of your function for testing or canary deployments.
- 8. Monitoring and Logging:** AWS provides CloudWatch for monitoring and logging Lambda functions. You can view metrics, set up alarms, and monitor the performance and resource usage of your functions.

Lambda is widely used for various use cases, such as real-time file processing, data transformations, web and mobile backends, IoT data processing, chatbots, and more. It offers a highly scalable and cost-effective way to build serverless applications.

Docker Hub:

Docker Hub is a cloud-based registry service provided by Docker, the popular containerization platform. It serves as a central repository for Docker images, allowing developers to store, manage, and distribute container images. Docker Hub offers both free and paid plans, providing a platform for sharing containerized applications and collaborating with other developers.

What is Docker?

Docker is an Open Platform for developing, shipping and running the applications.

Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage applications.

By taking advantage of Docker's methodologies for shipping, testing, and deploying code quickly, you can significantly reduce the delay between writing code and running it in production.

Key features and functionalities of Docker Hub include:

1. **Image Hosting:** Docker Hub allows users to store and host Docker images securely in the cloud. It provides storage space for public and private repositories, making it easy to share and distribute container images.
2. **Versioning and Tagging:** Docker images can have multiple versions and tags associated with them. Docker Hub allows developers to tag their images with meaningful labels, making it easier to manage different versions and variations of container images.
3. **Repository Organization:** Docker Hub provides a repository-centric organization, allowing users to group related images together within repositories. This helps to maintain a structured and organized collection of container images.
4. **Collaboration and Sharing:** Docker Hub allows developers to share their Docker images with others by making them public or sharing them privately with specific users or teams. This enables collaboration and simplifies the process of sharing containerized applications.
5. **Automated Builds:** Docker Hub supports automated builds, which means it can automatically build Docker images based on the source code or Dockerfile

provided by the user. This feature is useful for continuous integration and deployment workflows.

6. Webhooks and Triggers: Docker Hub provides webhooks and triggers that can be used to trigger actions or notifications based on events like image pushes, new tags, or repository updates. This allows for integration with other systems and automation of workflows.

7. Integration with Docker CLI: Docker Hub seamlessly integrates with the Docker command-line interface (CLI), making it easy to push and pull images to and from Docker Hub using familiar Docker commands.

8. Official Images and Certification: Docker Hub hosts a collection of official images maintained by Docker and its partners. These images are carefully curated, ensuring they are secure, up-to-date, and well-documented. Docker Hub also offers certification programs for images, indicating they meet certain quality standards and best practices.

Overall, Docker Hub plays a crucial role in the Docker ecosystem by providing a centralized platform for storing, sharing, and distributing Docker images. It simplifies the process of managing containerized applications and enables collaboration among developers and teams.

Docker Platform:

Docker provides the ability to package and run an application in a loosely isolated environment called container. The isolation and security allows you to run many containers simultaneously on a given host. Container are lightweight and contain everything needed to run the application, so you do not need to rely on what is currently installed on the host. You can easily share containers while you work, and be sure that everyone you share with gets the same container that works in the same way.

Docker provides tooling and a platform to manage the lifecycle of your containers:

Develop your application and its supporting components using containers. The container becomes the unit for distributing and testing your application. When you're ready, deploy your application into your production environment.

Commands in Docker:

1. ****docker run**:** Create and start a new container from an image.

```
docker run [options] [image] [command]
```

2. ****docker build**:** Build a new Docker image from a Dockerfile.

```
docker build [options] [path]
```

3. ****docker pull**:** Download an image from a registry.

```
docker pull [image]
```

4. ****docker push**:** Upload an image to a registry.

```
docker push [image]
```

5. ****docker images**:** List available Docker images on the host.

```
docker images [options]
```

6. ****docker ps**:** List running containers.

```
docker ps [options]
```

7. ****docker start**:** Start a stopped container.

```
docker start [container]
```

8. ****docker stop**:** Stop a running container.

```
docker rmi [image]
```

9. ****docker rm**:** Remove one or more containers.

```
docker rm [container]
```

10. ****docker rmi**:** Remove one or more images.

```
docker rmi [image]
```

11. ****docker exec**:** Run a command inside a running container.

```
docker exec [options] [container] [command]
```

12. ****docker logs**:** Fetch the logs of a container.

```
docker logs [options] [container]
```

13. ****docker inspect**:** Display detailed information about a container or image.

```
docker inspect [options] [object]
```

14. ****docker network**:** Manage Docker networks.

```
docker network [options]
```

15. ****docker-compose**:** Manage multi-container Docker applications with a YAML file.

```
docker-compose [options] [command]
```

These are just a few examples of the many Docker commands available. You can find more detailed information about these commands and additional ones in the Docker documentation (<https://docs.docker.com/>).

GitHub:

GitHub is a web-based platform for version control and collaboration that allows developers to store and manage their code repositories. It provides a hosting service for Git repositories, which is a distributed version control system used by many developers worldwide.

Here are some key features and functionalities of GitHub:

1. **Version Control:** GitHub uses Git, a widely used version control system, to track changes in code repositories. It allows developers to create branches, make changes, and merge them back into the main codebase.
2. **Code Collaboration:** GitHub enables developers to work collaboratively on projects. Multiple developers can contribute to a project by creating branches, submitting pull requests, and reviewing each other's code.
3. **Issue Tracking:** GitHub provides a built-in issue tracking system that allows users to create and manage issues, bug reports, feature requests, and other tasks related to a project. This helps teams to organize and prioritize their work effectively.

4. **Project Management:** GitHub offers project management features like project boards, milestones, and labels to help teams track the progress of their work and manage tasks efficiently.
5. **Continuous Integration and Deployment:** GitHub integrates with various continuous integration and deployment tools, such as Travis CI, Jenkins, and GitHub Actions. These integrations enable developers to automate testing, build processes, and deployment pipelines.
6. **Community and Social Features:** GitHub fosters a vibrant developer community by providing features like social interactions, such as following other developers, starring repositories, and contributing to open-source projects.
7. **Documentation and Wikis:** GitHub allows developers to create documentation and wikis for their projects, making it easier to share knowledge and information with others.
8. **Integration with Other Tools:** GitHub integrates with a wide range of development tools and services, such as IDEs, code editors, project management tools, and chat platforms, enhancing the overall development workflow.

GitHub is widely used by individual developers, open-source projects, and organizations of all sizes. It provides a platform for collaboration, code sharing, and community building, making it an essential tool in the software development ecosystem.

Git:

Git is a distributed version control system (VCS) widely used for managing source code and project files. It was created by Linus Torvalds in 2005 to help with the development of the Linux kernel, but it has since become one of the most popular version control systems for software development.

Git allows multiple developers to work on the same project simultaneously and independently, and it tracks changes made to files over time. It provides features such as branching, merging, and version history tracking, which are essential for collaborative software development.

Here are some key concepts and commands related to Git:

1. Repository: A repository, or repo, is a collection of files and their revision history. It can be either a local repository on your computer or a remote repository hosted on a server.

2. Clone: To clone a repository means to create a local copy of a remote repository on your computer. The command to clone a repository is:

```
git clone <repository-url>
```

3. Commit: A commit represents a snapshot of changes made to the files in a repository. It records the changes and a descriptive message. The command to commit changes is:

```
git branch <branch-name>
```

4. Branch: A branch is a parallel version of a repository that allows developers to work on different features or bug fixes independently. The main branch is typically called "master" or "main," and additional branches can be created and merged later. To create a new branch, use:

```
git branch <branch-name>
```

5. Checkout: To switch between branches or to restore files to a previous commit, you can use the checkout command. For example, to switch to a different branch, use:

```
git checkout <branch-name>
```

6. Merge: Merging combines changes from different branches into one branch. This is typically done when a feature or bug fix is complete. To merge a branch into the current branch, use:

```
git push origin <branch-name>
```

7. Push: Pushing refers to sending local commits to a remote repository, making them accessible to other team members. The command to push changes is:

```
git push origin <branch-name>
```

8. Pull: Pulling is used to update the local repository with the latest changes from a remote repository. The command to pull changes is:

```
git pull origin <branch-name>
```

9. Status: Shows the current state of the repository, including modified files, untracked files, and the current branch.

```
git status
```

10. Log: Displays the commit history, including the commit hash, author, date, and commit message.

```
git log
```

11. Diff: Shows the differences between the current changes and the last commit or between different branches.

```
git diff
```

12. Remote: Manages remote repositories. It allows you to add, remove, and list remote repositories.

```
git remote add <name> <url>
git remote remove <name>
git remote -v
```

13. Fetch: Retrieves changes from a remote repository but does not merge them into the local branch.

```
git remote add <name> <url>
git remote remove <name>
git remote -v
```

14. Reset: Discards changes in the working directory and staging area to a specific commit. It can be used to undo commits.

```
git revert <commit>
```

15. Revert: Creates a new commit that undoes the changes made in a previous commit.

```
git revert <commit>
```

16. Tag: Marks a specific commit with a tag to give it a meaningful name, often used to indicate versions or releases.

```
git tag <tag-name>
```

17. Stash: Temporarily saves changes that are not ready to be committed, allowing you to switch branches or apply the changes later.

```
git stash
git stash apply
git stash pop
```

18. Remote Branches: Working with branches in a remote repository.

```
```
git push origin --delete <branch-name> # Deletes a remote branch
git checkout --track origin/<branch-name> # Creates a local
branch that tracks a remote branch
```

```

```
pipeline {
    agent any

    stages {
        stage('Checkout') {
            steps {
                echo "Performing Checkout"
                checkout([$class: 'GitSCM',
                          branches: [[name: 'master']],
                          userRemoteConfigs: [[url:
'https://github.com/kmwaseem15/techgalena.git']]])
            }
        }

        stage('Build Docker Image') {
            steps {
                echo "Building Docker Image"
                sh 'docker build -t kmwaseem15/nginx .'
            }
        }

        stage('Push to DockerHub') {
            steps {
                echo "Pushing to DockerHub"
                sh 'docker push kmwaseem15/nginx'
            }
        }

        stage('Pull on EC2') {
            steps {
                echo "Pulling on EC2"
                sshagent(['992030']) {
                    sh "ssh -o StrictHostKeyChecking=no -l root
18.207.244.248 'docker pull kmwaseem15/nginx'"
                }
            }
        }
    }
}
```

```
        }
    }

stage('Run Docker Container on EC2') {
    steps {
        echo "Running Docker Container on EC2"
        sshagent(['992030']) {
            script {
                // Kill all running Docker containers
                def killCommand = "ssh -o
StrictHostKeyChecking=no -l root 18.207.244.248 'docker kill
\$(docker ps -q)''"
                sh(killCommand)

                // Run the Docker container
                def runCommand = "ssh -o
StrictHostKeyChecking=no -l root 18.207.244.248 'docker run -d -p
8080:80 kmwaseem15/nginx'"
                try {
                    sh(runCommand)
                } catch (Exception e) {
                    echo "Failed to start the container:
${e.getMessage()}"
                    error "Failed to start the container"
                }
            }
        }
    }
}
```