

## Problem Statement -

Setup a CI/CD pipeline using the tools your choice(or preferably the mentioned tools).

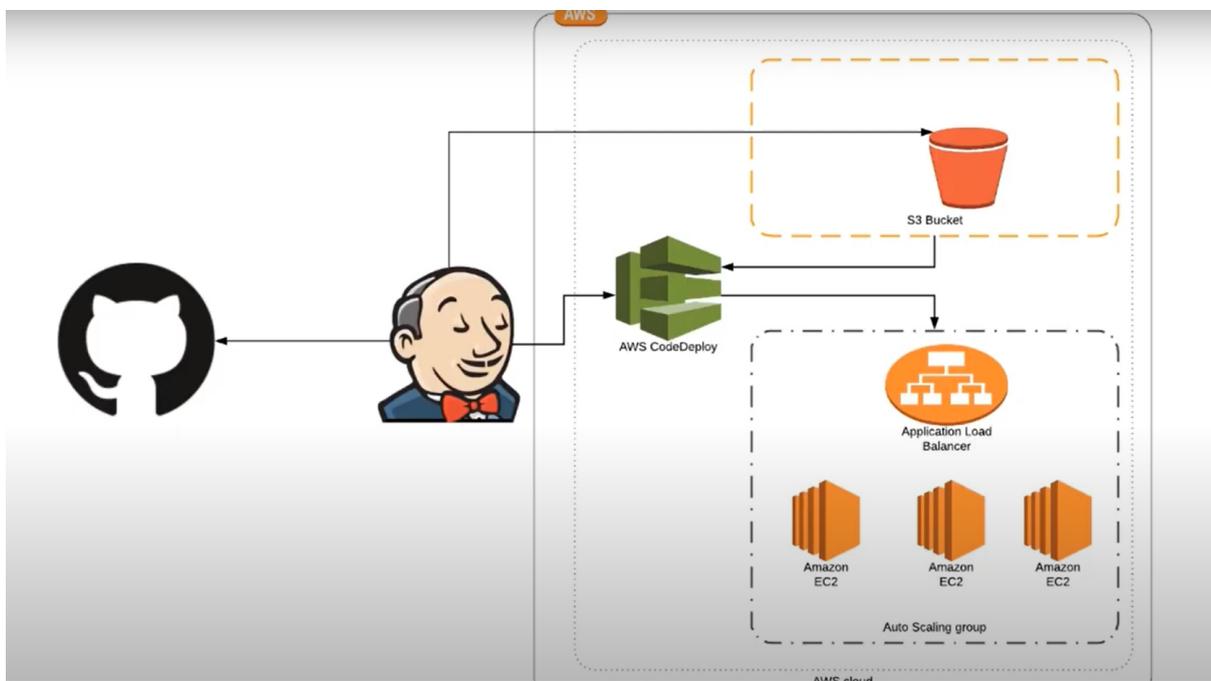
1. It should deploy a simple web application to a server on a code push to a repository.
2. The deployed web application should be reachable on any web browser.
3. Make it scalable such that when load increases the number of servers scale up and down making sure the new servers have the updated code.

## Additional requirements

1. Setup to be done using AWS, Jenkins, CodeDeploy
2. Jenkins should not be on the same server as the application being deployed to

**Tools** 1. Jenkins 2. Git/Bitbucket 3. AWS EC2 4. AWS CodeDeploy

## Solution -



# Step 1 - Create Two IAM roles as -

1. In first provide [AmazonEC2RoleforAWSCodeDeploy](#) and [AmazonS3FullAccess](#)
2. In second provide [codedeplyrole](#) role.

This roles will be used further.

1.

The screenshot shows the AWS IAM console interface. The left sidebar contains navigation options like Dashboard, Access management, User groups, Users, Roles, Policies, Identity providers, Account settings, Access reports, Access analyzer, Archive rules, Analyzers, Settings, Credential report, Organization activity, and Service control policies (SCPs). The main content area displays the details for the role 'ec2codedeploy'. The Role ARN is 'am:aws:iam::616448194411:role/ec2codedeploy'. The Role description is 'Allows EC2 instances to call AWS services on your behalf.' The Instance Profile ARNs is 'am:aws:iam::616448194411:instance-profile/ec2codedeploy'. The Path is '/'. The Creation time is '2022-01-07 23:33 UTC+0530' and the Last activity is '2022-01-09 17:15 UTC+0530 (Today)'. The Maximum session duration is '1 hour'. The Permissions tab is active, showing 'Permissions policies (2 policies applied)'. The attached policies are 'AmazonEC2RoleforAWSCodeDeploy' and 'AmazonS3FullAccess', both AWS managed policies. There is also a 'Permissions boundary (not set)' and an option to 'Generate policy based on CloudTrail events'. The bottom of the screenshot shows the Windows taskbar with the search bar and system tray.

2.

The screenshot shows the AWS IAM console interface. The left sidebar contains navigation options like Dashboard, Access management, User groups, Users, Roles, Policies, Identity providers, Account settings, Access reports, Access analyzer, Archive rules, Analyzers, Settings, Credential report, Organization activity, and Service control policies (SCPs). The main content area displays the details for the role 'codedeplyrole'. The Role ARN is 'am:aws:iam::616448194411:role/codedeplyrole'. The Role description is 'Allows CodeDeploy to call AWS services such as Auto Scaling on your behalf.' The Instance Profile ARNs is '/'. The Path is '/'. The Creation time is '2022-01-07 23:36 UTC+0530' and the Last activity is '2022-01-09 15:23 UTC+0530 (Today)'. The Maximum session duration is '1 hour'. The Permissions tab is active, showing 'Permissions policies (1 policy applied)'. The attached policy is 'AWSCodeDeployRole', an AWS managed policy. There is also a 'Permissions boundary (not set)' and an option to 'Generate policy based on CloudTrail events'. The bottom of the screenshot shows the Windows taskbar with the search bar and system tray.

# Step 2 - Create 4 ec2 instances and put user data in it -

Step 1: Choose an Amazon Machine Image (AMI)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. You can select an AMI provided by AWS, our user community, or the AWS Marketplace, or you can select one of your own AMIs.

Search for an AMI by entering a search term e.g. "Windows"

Search by Systems Manager parameter

Quick Start

My AMIs

AWS Marketplace

Community AMIs

Free tier only

Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type - ami-052cef05d01020f1d (64-bit x86) / ami-07a4db51302058e77 (64-bit Arm)

Amazon Linux 2 comes with five years support. It provides Linux kernel 5.10 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras. This AMI is the successor of the Amazon Linux AMI that is now under maintenance only mode and has been removed from this wizard.

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Amazon Linux 2 AMI (HVM) - Kernel 4.14, SSD Volume Type - ami-0002bdad91f793433 (64-bit x86) / ami-005d492f446757bfe (64-bit Arm)

Amazon Linux 2 comes with five years support. It provides Linux kernel 4.14 tuned for optimal performance on Amazon EC2, systemd 219, GCC 7.3, Glibc 2.26, Binutils 2.29.1, and the latest software packages through extras. This AMI is the successor of the Amazon Linux AMI that is now under maintenance only mode and has been removed from this wizard.

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

macOS Monterey 12.0.1 - ami-052e449e86ad9ead5

The macOS Monterey AMI is an EBS-backed, AWS-supported image. This AMI includes the AWS Command Line Interface, Command Line Tools for Xcode, Amazon SSM Agent, and Homebrew. The AWS Homebrew Tap includes the latest versions of multiple AWS packages included in the AMI.

Root device type: ebs Virtualization type: hvm ENA Enabled: Yes

Feedback English (US) © 2022, Amazon Internet Services Private Ltd. or its affiliates. Privacy Terms Cookie preferences

1640586101195.pdf Show all

Type here to search 15°C Haze 17:36 09-01-2022

Step 2: Choose an Instance Type

Amazon EC2 provides a wide selection of instance types optimized to fit different use cases. Instances are virtual servers that can run applications. They have varying combinations of CPU, memory, storage, and networking capacity, and give you the flexibility to choose the appropriate mix of resources for your applications. Learn more about instance types and how they can meet your computing needs.

Filter by: All instance families Current generation Show/Hide Columns

Currently selected: t2.micro (- ECU, 1 vCPUs, 2.5 GHz, -, 1 GiB memory, EBS only)

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance	IPv6 Support
<input type="checkbox"/>	t2	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
<input checked="" type="checkbox"/>	t2	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	t2	t2.small	1	2	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	t2	t2.medium	2	4	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	t2	t2.large	2	8	EBS only	-	Low to Moderate	Yes
<input type="checkbox"/>	t2	t2.xlarge	4	16	EBS only	-	Moderate	Yes
<input type="checkbox"/>	t2	t2.2xlarge	8	32	EBS only	-	Moderate	Yes
<input type="checkbox"/>	t3	t3.nano	2	0.5	EBS only	Yes	Up to 5 Gigabit	Yes
<input type="checkbox"/>	t3	t3.micro	2	1	EBS only	Yes	Up to 5 Gigabit	Yes

Cancel Previous Review and Launch Next: Configure Instance Details

Feedback English (US) © 2022, Amazon Internet Services Private Ltd. or its affiliates. Privacy Terms Cookie preferences

1640586101195.pdf Show all

Type here to search 15°C Haze 17:37 09-01-2022

Step 3: Configure Instance Details

Tenancy: Shared - Run a shared hardware instance  
 Credit specification: Unlimited  
 File systems: Add file system / Create new file system

Advanced Details

Enclave:  Enable  
 Metadata accessible: Enabled  
 Metadata version: V1 and V2 (token optional)  
 Metadata token response hop limit: 1  
 Allow tags in metadata: Disabled  
 User data:  As text  As file  Input is already base64 encoded

```

#!bin/bash
sudo yum -y update
sudo yum -y install ruby
sudo yum -y install wget
cd /home/ec2-user
Wget https://aws-codedeploy-ap-south-1.s3.ap-south-
  
```

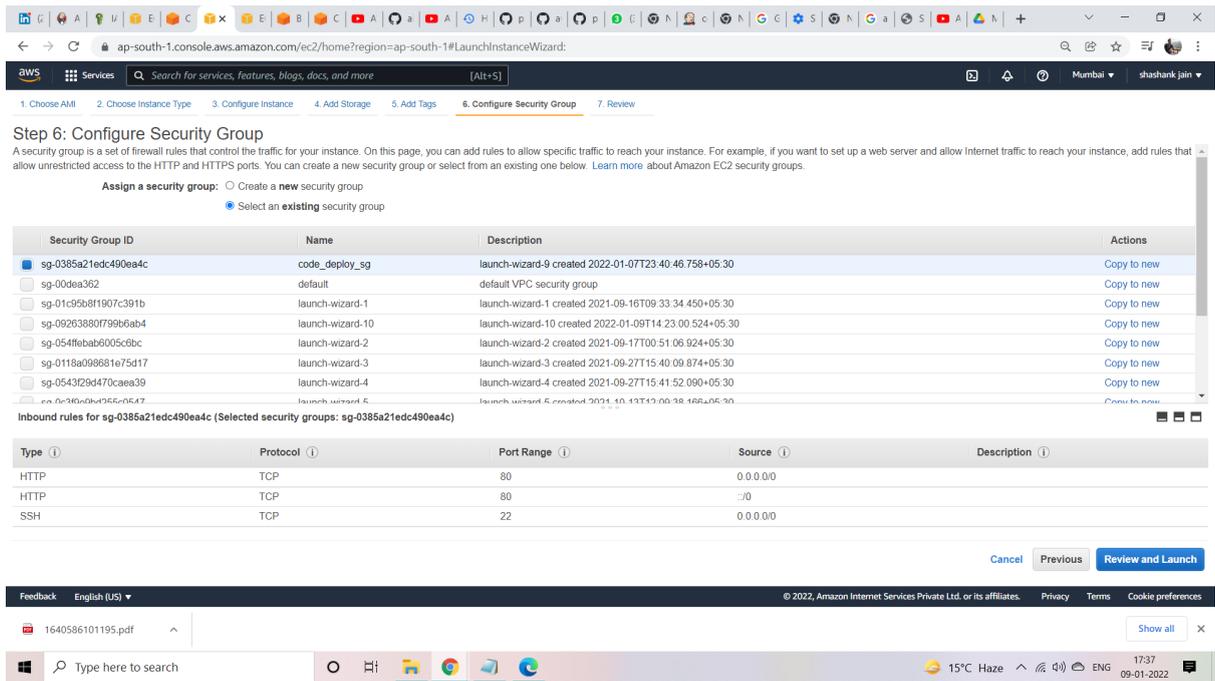
Buttons: Cancel, Previous, Review and Launch, Next: Add Storage

Launch Instance Wizard Summary

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public IP	IPv6 IPs	Key Name
ec2-code_de...	i-00599c5e78315249	t2.micro	ap-south-1a	running	2/2 checks ...	None	ec2-35-154-3-124.ap-s...	35.154.3.124	-	code_depl
ec2-code_de...	i-0760aa95f156002bd	t2.micro	ap-south-1a	running	2/2 checks ...	None	ec2-65-2-149-37.ap-so...	65.2.149.37	-	code_depl
ec2-code_de...	i-098560eae9dbc1117	t2.micro	ap-south-1a	running	2/2 checks ...	None	ec2-13-234-18-87.ap-s...	13.234.18.87	-	code_depl
ec2-code_de...	i-0ad996dcb29dcca5	t2.micro	ap-south-1a	running	2/2 checks ...	None	ec2-65-0-102-150.ap-s...	65.0.102.150	-	code_depl

Instance: i-00599c5e78315249 (ec2-code\_deploy) Public DNS: ec2-35-154-3-124.ap-south-1.compute.amazonaws.com

Description	Status Checks	Monitoring	Tags
Instance ID	i-00599c5e78315249		
Instance state	running		
Instance type	t2.micro		
Finding	Opt-in to AWS Compute Optimizer for recommendations. Learn more		
Private DNS	ip-172-31-43-41.ap-south-1.compute.internal		
Public DNS (IPv4)	ec2-35-154-3-124.ap-south-1.compute.amazonaws.com		
IPv4 Public IP	35.154.3.124		
IPv6 IPs	-		
Elastic IPs	-		
Availability zone	ap-south-1a		



User data i have used -

```
#!/bin/bash
```

```
sudo yum -y update
```

```
sudo yum -y install ruby
```

```
sudo yum -y install wget
```

```
cd /home/ec2-user
```

```
Wget
```

```
https://aws-codedeploy-ap-south-1.s3.ap-south-1.amazonaws.com/atest/install
```

```
sudo chmod +x ./install
```

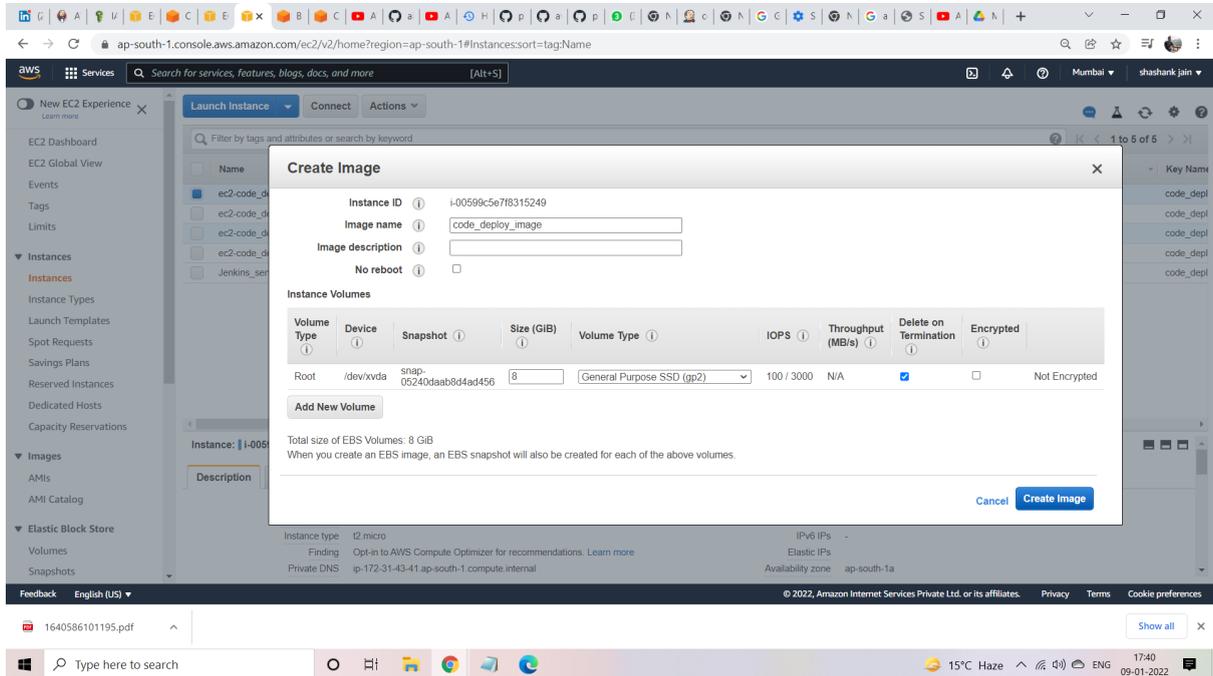
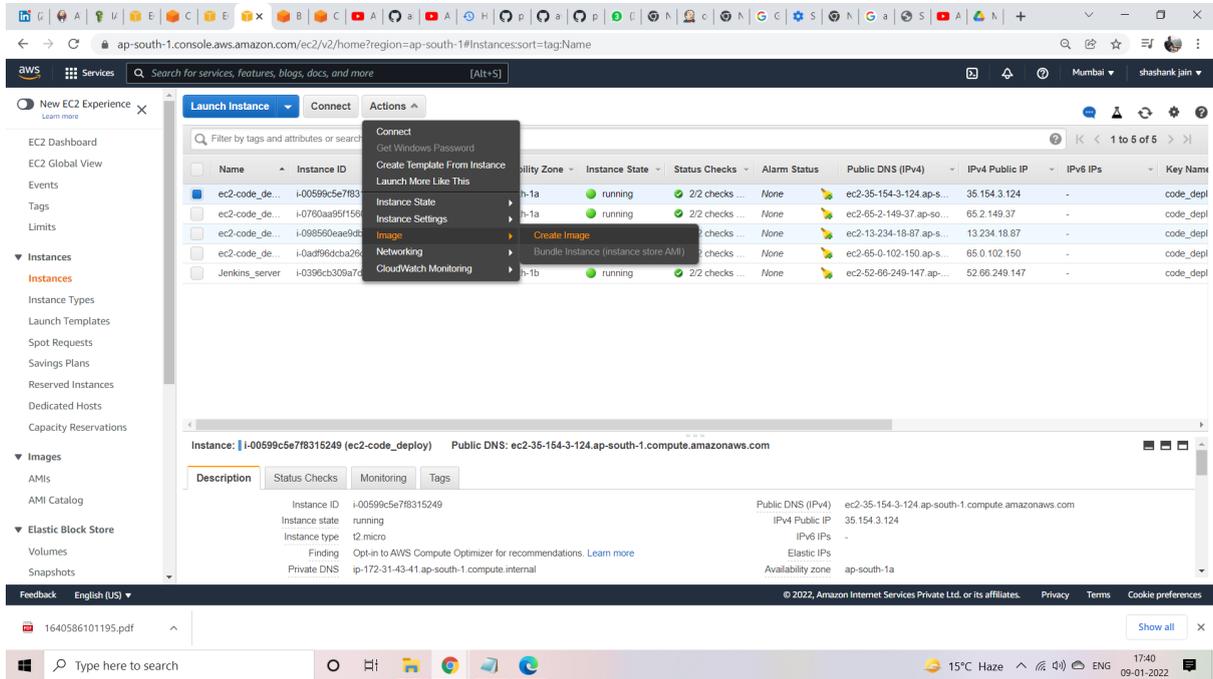
```
sudo ./install auto
```

```
sudo yum install -y python-pip
```

```
sudo pip install awscli
```

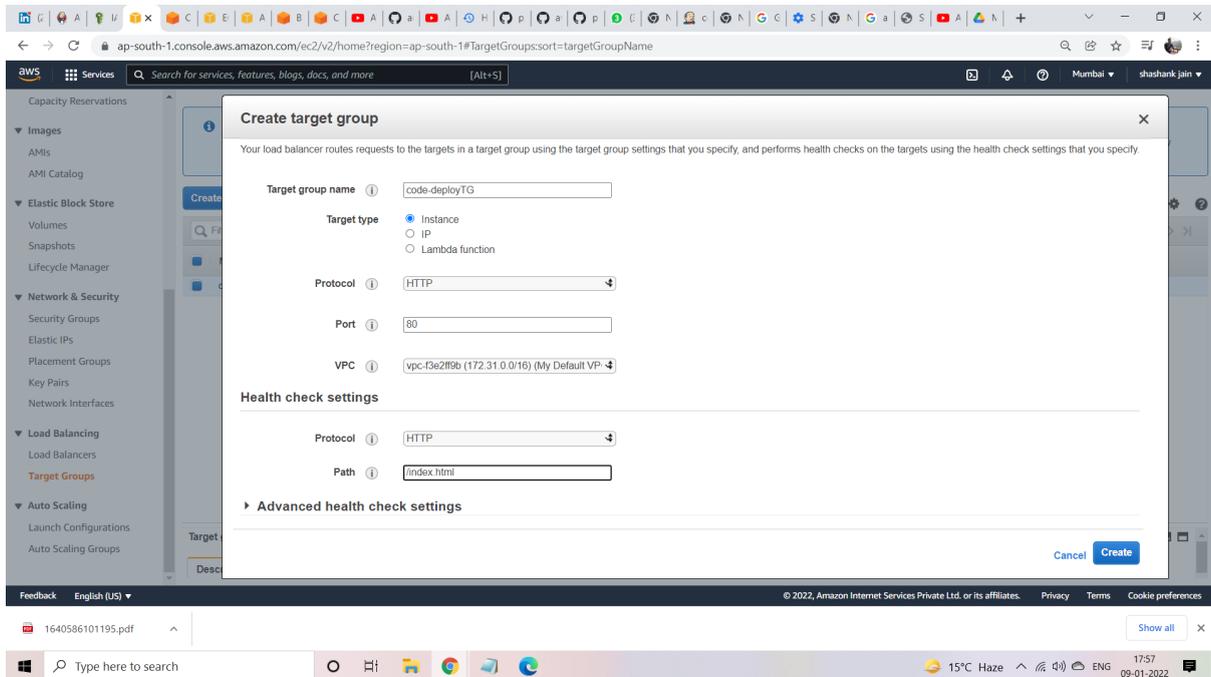
- This user data includes installation of some software like aws codedeploy , aws cli etc.
- add first IAM role to the instances .
- Create Security Group and provide http 80 and ssh 22 rules.
- Than create instances.

# Step 3 - Create an of the instance which we will use in AutoScalingGroup.



## Step 4 - Create Application Load Balancer.

1. Create target group than load balancer , in target group give path to /index.html , which is the app file containing code over git.

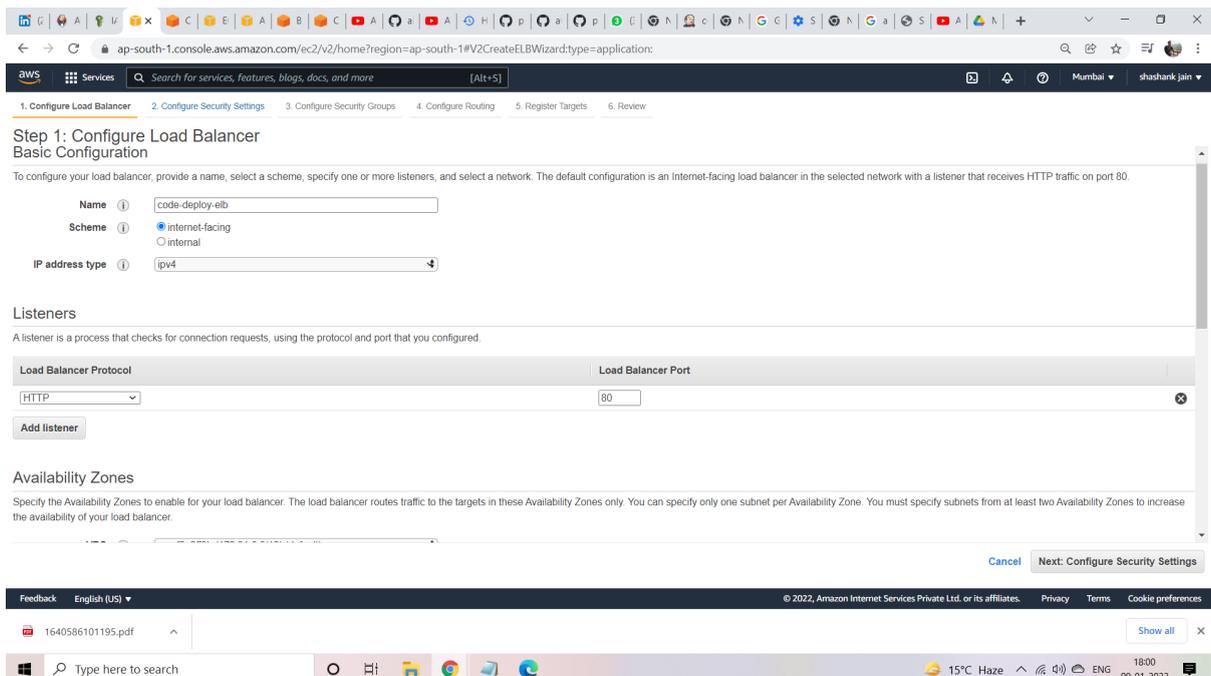


The screenshot shows the AWS Management Console interface with a 'Create target group' dialog box open. The dialog box contains the following fields and options:

- Target group name:** code-deployTG
- Target type:** Instance (selected), IP, Lambda function
- Protocol:** HTTP
- Port:** 80
- VPC:** vpc-f3a2f89b (172.31.0.0/16) (My Default VPC)
- Health check settings:**
  - Protocol:** HTTP
  - Path:** /index.html
- Advanced health check settings:** (collapsed)

Buttons for 'Cancel' and 'Create' are visible at the bottom right of the dialog box. The background shows the AWS console navigation menu and the top navigation bar.

## Now Create Application Load Balancer -



The screenshot shows the AWS Management Console interface with the 'Create Application Load Balancer' wizard. The wizard is currently on 'Step 1: Configure Load Balancer' and shows the following configuration:

- 1. Configure Load Balancer:** (selected)
- 2. Configure Security Settings**
- 3. Configure Security Groups**
- 4. Configure Routing**
- 5. Register Targets**
- 6. Review**

**Step 1: Configure Load Balancer**  
Basic Configuration

To configure your load balancer, provide a name, select a scheme, specify one or more listeners, and select a network. The default configuration is an Internet-facing load balancer in the selected network with a listener that receives HTTP traffic on port 80.

- Name:** code-deploy-elb
- Scheme:** internet-facing (selected), internal
- IP address type:** ipv4

**Listeners**  
A listener is a process that checks for connection requests, using the protocol and port that you configured.

Load Balancer Protocol	Load Balancer Port
HTTP	80

**Availability Zones**  
Specify the Availability Zones to enable for your load balancer. The load balancer routes traffic to the targets in these Availability Zones only. You can specify only one subnet per Availability Zone. You must specify subnets from at least two Availability Zones to increase the availability of your load balancer.

Buttons for 'Cancel' and 'Next: Configure Security Settings' are visible at the bottom right of the wizard.

ap-south-1.console.aws.amazon.com/ec2/v2/home?region=ap-south-1#V2CreateELBWizard?type=application:

1. Configure Load Balancer 2. Configure Security Settings 3. **Configure Security Groups** 4. Configure Routing 5. Register Targets 6. Review

### Step 3: Configure Security Groups

A security group is a set of firewall rules that control the traffic to your load balancer. On this page, you can add rules to allow specific traffic to reach your load balancer. First, decide whether to create a new security group or select an existing one.

**Assign a security group**

- Create a **new** security group
- Select an **existing** security group

Security group name:

Description:

Type	Protocol	Port Range	Source
HTTP	TCP	80	Custom 0.0.0.0/0

Feedback English (US) © 2022, Amazon Internet Services Private Ltd. or its affiliates. Privacy Terms Cookie preferences

1640586101195.pdf

Type here to search 15°C Haze ENG 18:04 09-01-2022

ap-south-1.console.aws.amazon.com/ec2/v2/home?region=ap-south-1#V2CreateELBWizard?type=application:

1. Configure Load Balancer 2. Configure Security Settings 3. Configure Security Groups 4. **Configure Routing** 5. Register Targets 6. Review

### Step 4: Configure Routing

on this load balancer. You can edit or add listeners after the load balancer is created.

**Target group**

Target group:

Name:

Target type

- Instance
- IP
- Lambda function

Protocol:

Port:

Protocol version

- HTTP1  
Send requests to targets using HTTP/1.1. Supported when the request protocol is HTTP/1.1 or HTTP/2.
- HTTP2  
Send requests to targets using HTTP/2. Supported when the request protocol is HTTP/2 or gRPC, but gRPC-specific features are not available.
- gRPC  
Send requests to targets using gRPC. Supported when the request protocol is gRPC.

**Health checks**

Feedback English (US) © 2022, Amazon Internet Services Private Ltd. or its affiliates. Privacy Terms Cookie preferences

1640586101195.pdf

Type here to search 15°C Haze ENG 18:05 09-01-2022

ap-south-1.console.aws.amazon.com/ec2/v2/home?region=ap-south-1#V2CreateELBWizard?type=application

1. Configure Load Balancer 2. Configure Security Settings 3. Configure Security Groups 4. Configure Routing 5. Register Targets 6. Review

### Step 5: Register Targets

Register targets with your target group. If you register a target in an enabled Availability Zone, the load balancer starts routing requests to the targets as soon as the registration process completes and the target passes the initial health checks.

**Registered targets**  
To deregister instances, select one or more registered instances and then click Remove.

Remove

Instance	Name	Port	State	Security groups	Zone
No instances available.					

**Instances**  
To register additional instances, select one or more running instances, specify a port, and then click Add. The default port is the port specified for the target group. If the instance is already registered on the specified port, you must specify a different port.

Add to registered on port 80

Search Instances

Instance	Name	State	Security groups	Zone	Subnet ID	Subnet CIDR
i-0ad9f96dca26dcca5	ec2-code_deploy	running	launch-wizard-10	ap-south-1a	subnet-59f0ca31	172.31.32.0/20
i-098560eae9dbc1117	ec2-code_deploy	running	launch-wizard-10	ap-south-1a	subnet-59f0ca31	172.31.32.0/20
i-0760aa96f156002bd	ec2-code_deploy	running	launch-wizard-10	ap-south-1a	subnet-59f0ca31	172.31.32.0/20
i-00599c5e7f8315249	ec2-code_deploy	running	launch-wizard-10	ap-south-1a	subnet-59f0ca31	172.31.32.0/20
i-0fe8dc7ded70c8812	Code_deploy_ASG	running	AutoScaling-Security-Gr...	ap-south-1a	subnet-59f0ca31	172.31.32.0/20

Cancel Previous Next: Review

Feedback English (US) © 2022, Amazon Internet Services Private Ltd. or its affiliates. Privacy Terms Cookie preferences

1640586101195.pdf Show all

Type here to search 15°C Haze 18:06 09-01-2022

We have added the targets to already created instances.

ap-south-1.console.aws.amazon.com/ec2/v2/home?region=ap-south-1#V2CreateELBWizard?type=application

1. Configure Load Balancer 2. Configure Security Settings 3. Configure Security Groups 4. Configure Routing 5. Register Targets 6. Review

### Step 6: Review

Please review the load balancer details before continuing.

**Load balancer** [Edit](#)

- Name: codedeployelb1
- Scheme: internet-facing
- Listeners: Port 80 - Protocol:HTTP
- IP address type: ipv4
- VPC: vpc-f3e2ff9b
- Subnets: subnet-59f0ca31, subnet-e76e05ab, subnet-539f4428
- Tags:

**Security groups** [Edit](#)

- Security groups: code-deploy-sg

**Routing** [Edit](#)

- Target group: New target group
- Target group name: code-deploy-TG
- Port: 80
- Target type: instance
- Protocol: HTTP
- Protocol version: HTTP1
- Health check protocol: HTTP
- Path: /index.html
- Health check port: traffic.port
- Healthy threshold: 5

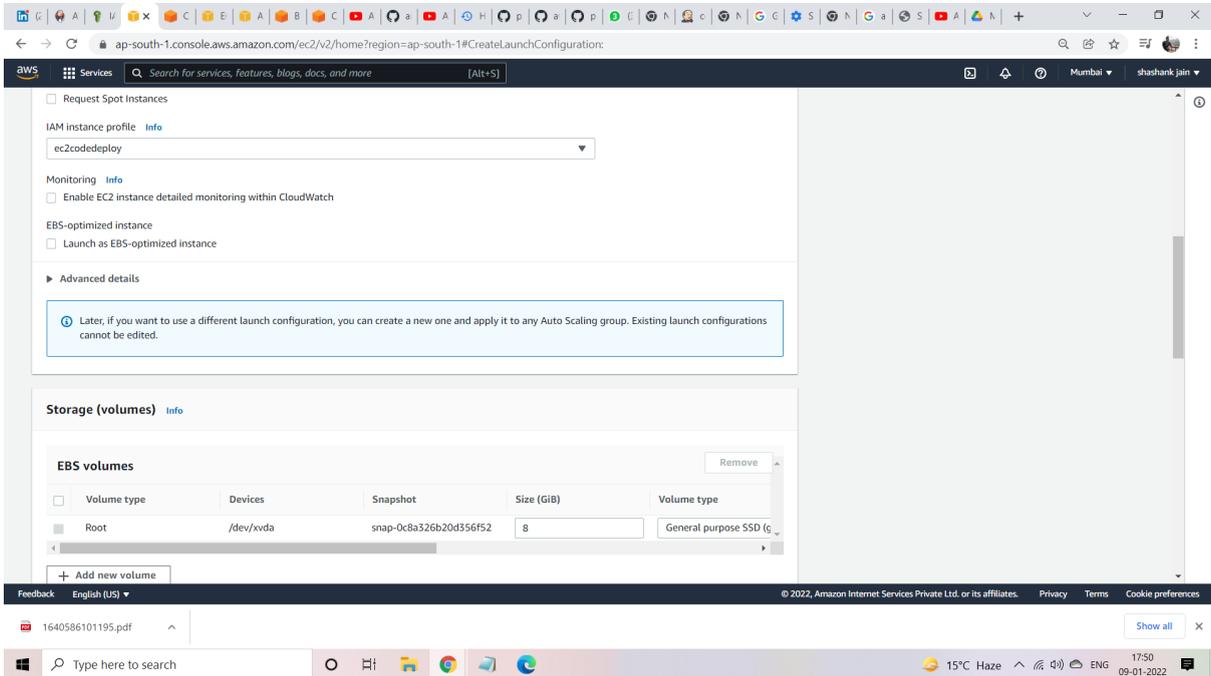
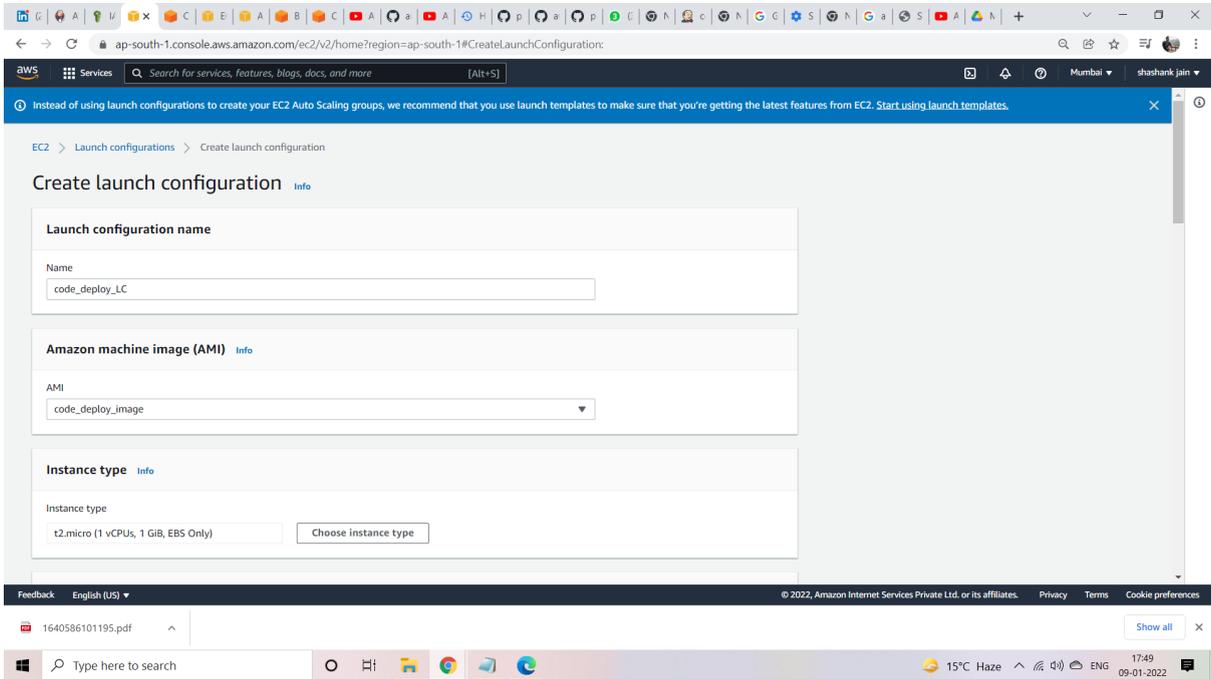
Cancel Previous Create

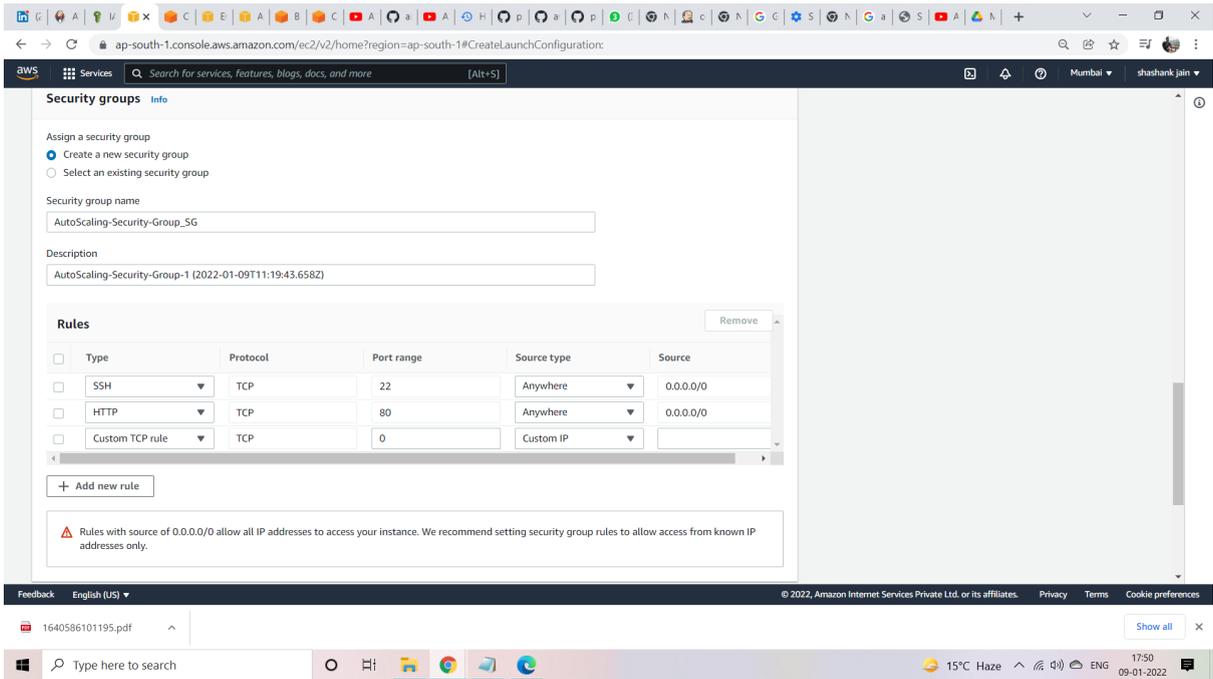
Feedback English (US) © 2022, Amazon Internet Services Private Ltd. or its affiliates. Privacy Terms Cookie preferences

1640586101195.pdf Show all

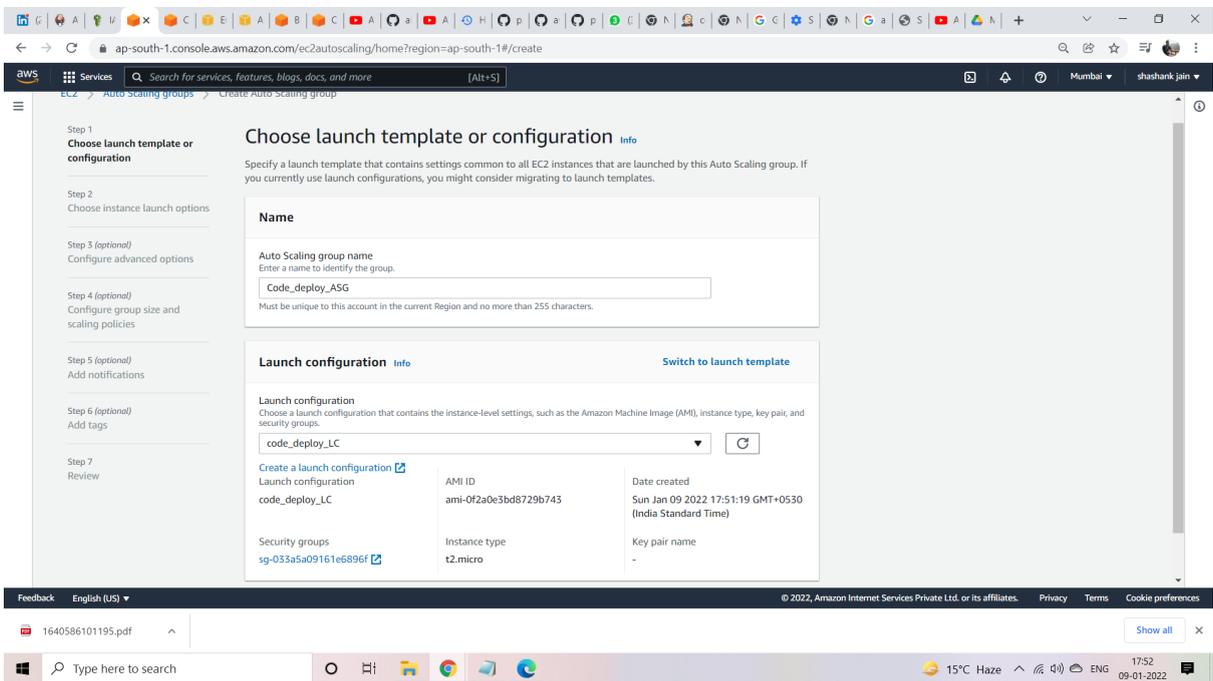
Type here to search 15°C Haze 18:06 09-01-2022

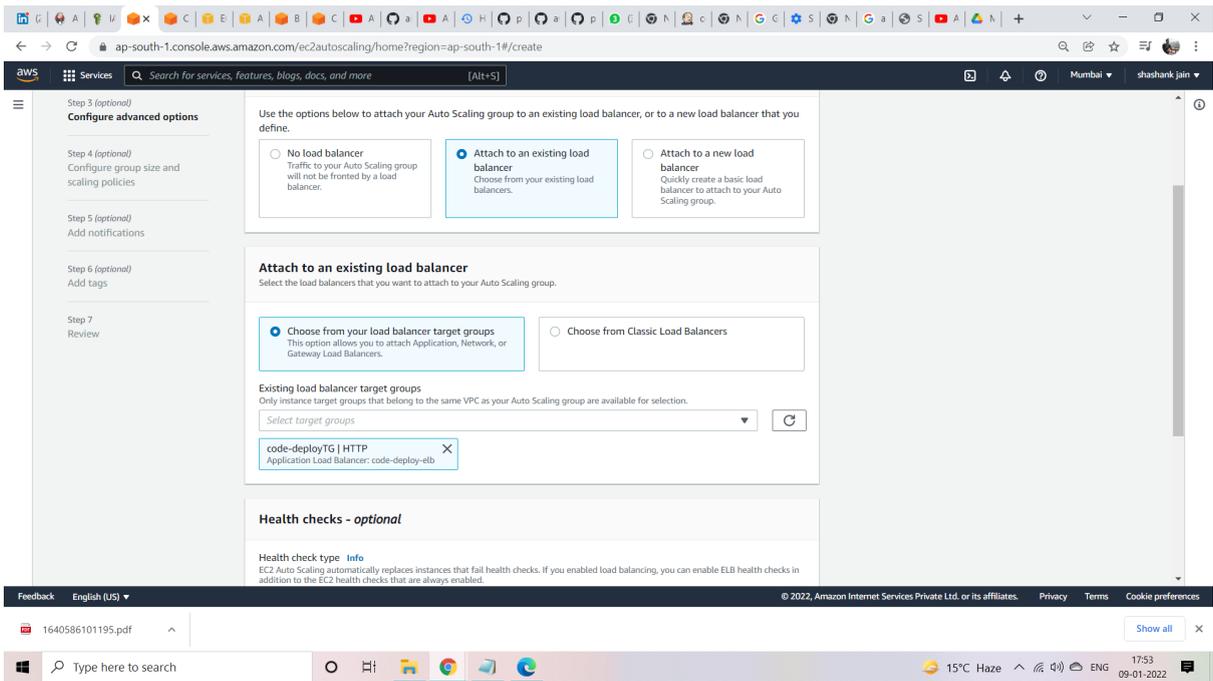
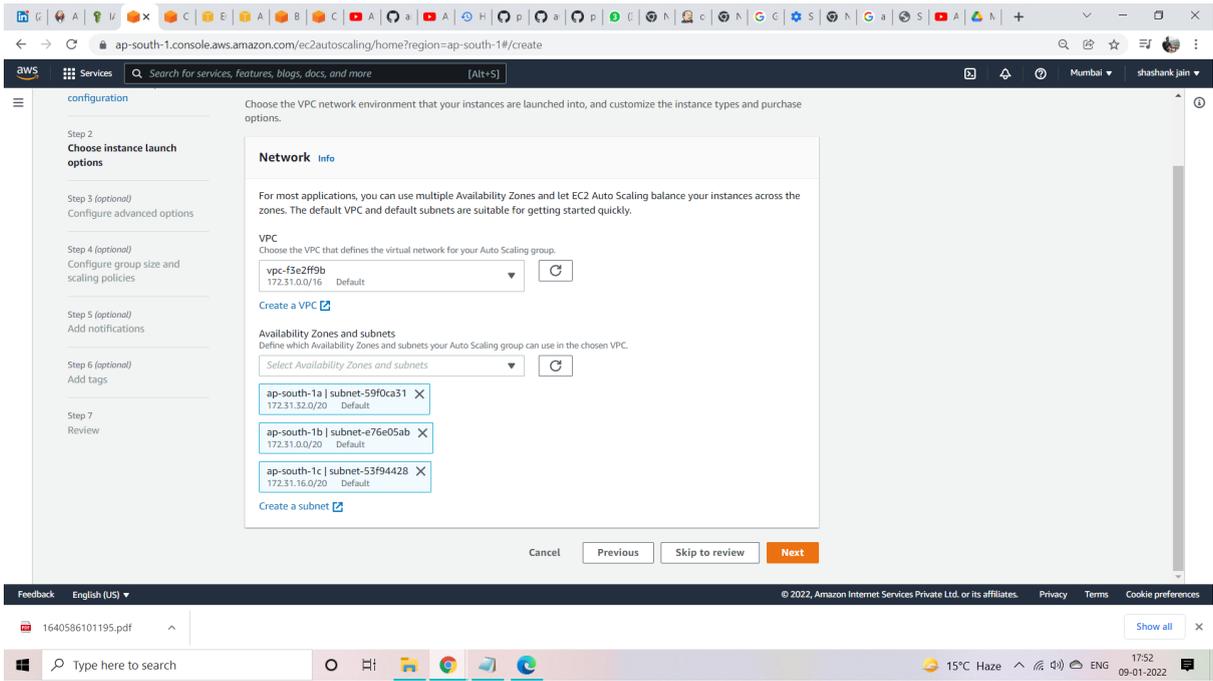
# Step 5 - Create Launch Configurations and then create Auto scaling group.



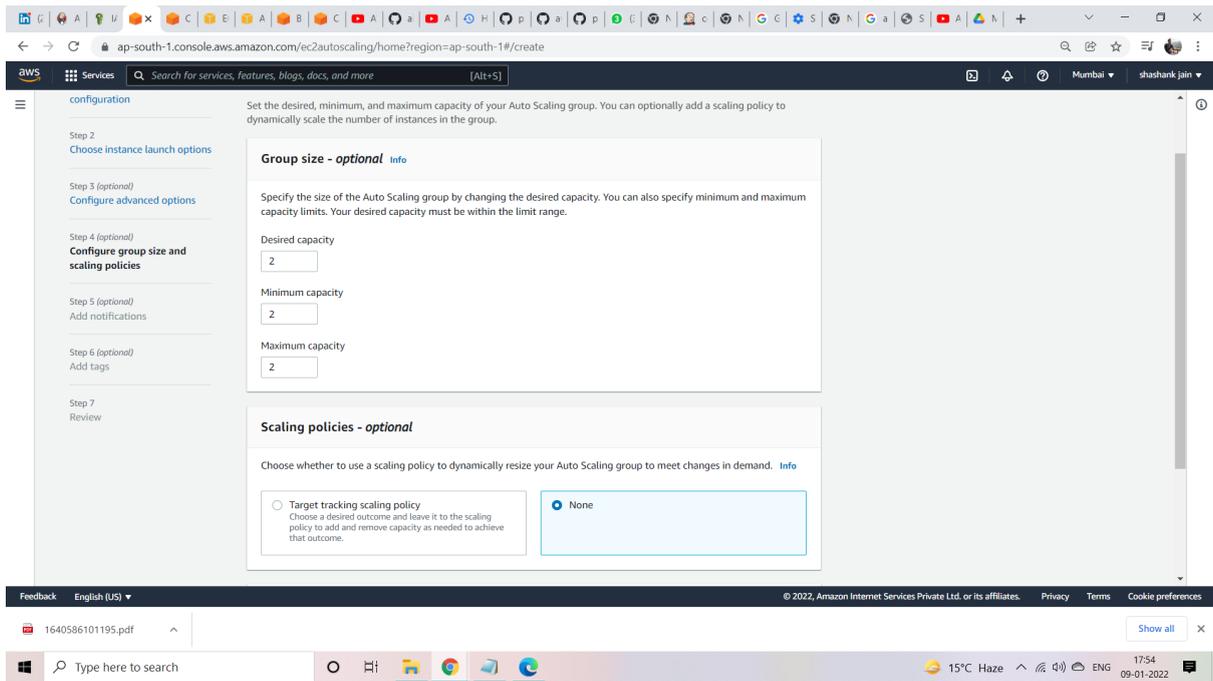


Launch Configuration created.  
Now create AutoScalingGroup

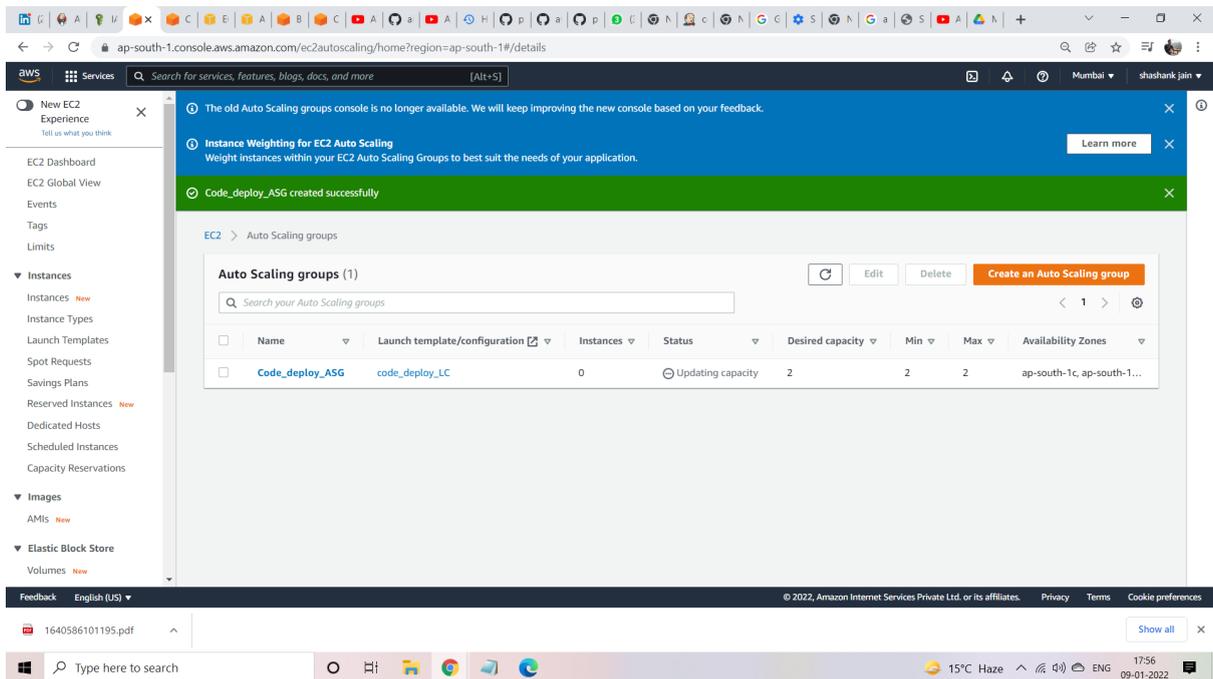




I have attached Application Load Balancer to it which is already Created



Give Desired and max capacity to 2



**Auto Scaling Group** is created.

We can see as soon as ASG created , the two instances are up.

Browser window showing the AWS Management Console. The address bar displays: `ap-south-1.console.aws.amazon.com/ec2/v2/home?region=ap-south-1#Instances:search=Code_deploy_ASG;sort=tag:Name`

The console shows a list of EC2 instances with the following data:

Name	Instance ID	Instance Type	Availability Zone	Instance State	Status Checks	Alarm Status	Public DNS (IPv4)	IPv4 Public IP	IPv6 IPs	Key Name
Code_deploy...	i-08fe4edc6e538141	t2.micro	ap-south-1b	running	2/2 checks ...	None	ec2-3-7-69-12.ap-south...	3.7.69.12	-	code_depl
Code_deploy...	i-0fe8dc7ded70c8812	t2.micro	ap-south-1a	running	2/2 checks ...	None	ec2-3-110-131-15.ap-s...	3.110.131.15	-	code_depl

The detailed view for instance `i-0fe8dc7ded70c8812` (Code\_deploy\_ASG) is shown below:

Description		Status Checks	Monitoring	Tags
Instance ID	i-0fe8dc7ded70c8812			
Instance state	running			
Instance type	t2.micro			
Finding	Opt-in to AWS Compute Optimizer for recommendations. <a href="#">Learn more</a>			
Private DNS	ip-172-31-42-7.ap-south-1.compute.internal			
Public DNS (IPv4)	ec2-3-110-131-15.ap-south-1.compute.amazonaws.com			
IPv4 Public IP	3.110.131.15			
IPv6 IPs	-			
Elastic IPs	-			
Availability zone	ap-south-1a			

The Windows taskbar at the bottom shows the search bar, task icons, system tray with weather (14°C Haze), time (18:35), and date (09-01-2022).

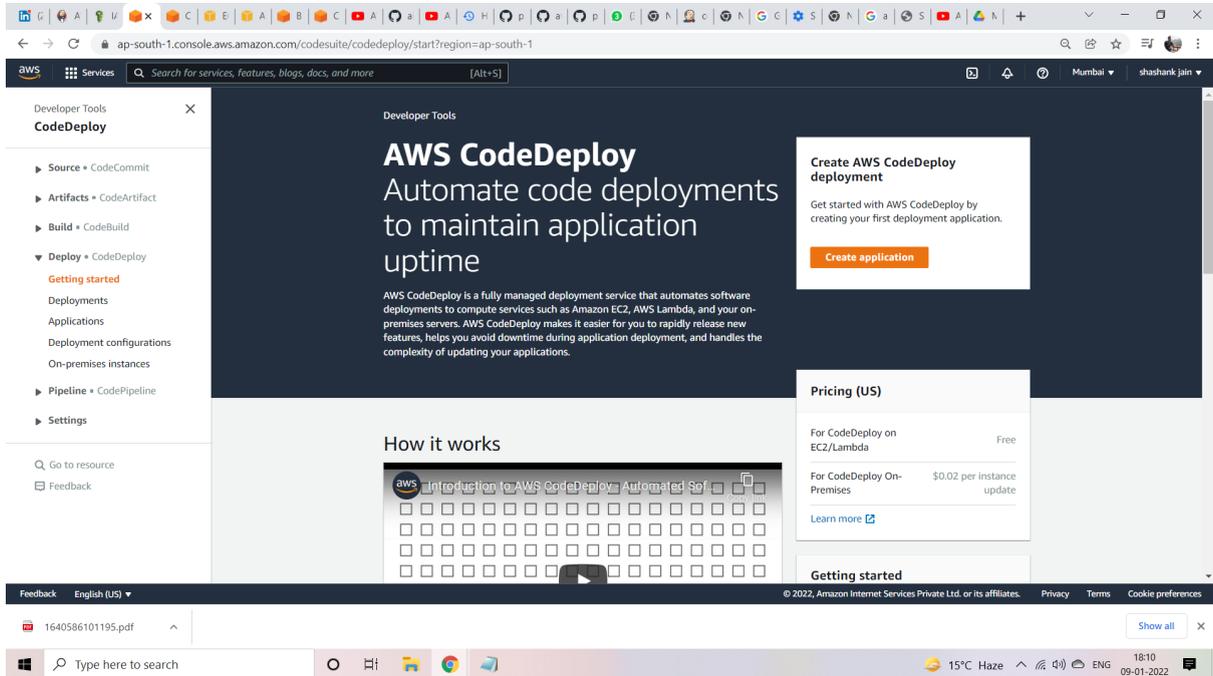
## **Step 6 - Create one Jenkins Instance and install jenkins software in it.**

I have used the below link to install JENKINS server into the ec2 instance -

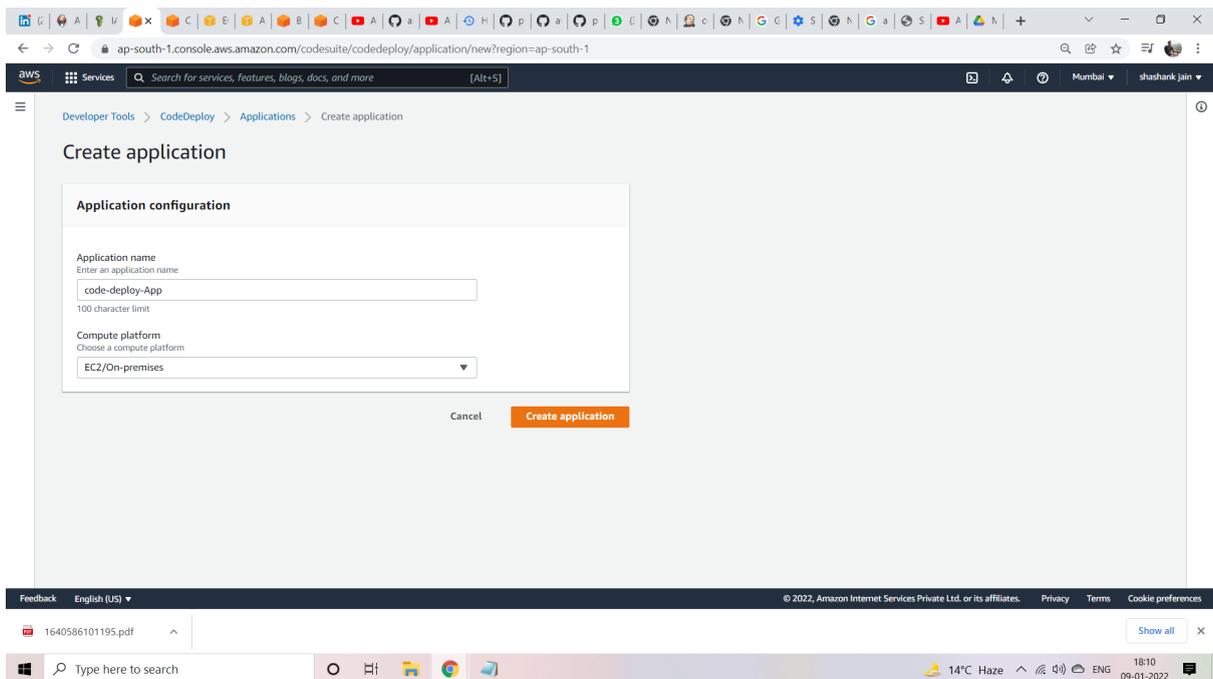
<https://www.jenkins.io/doc/tutorials/tutorial-for-installing-jenkins-on-AWS/>

**NOTE** - i have also installed git in it , using Command -  
- yum install git -y

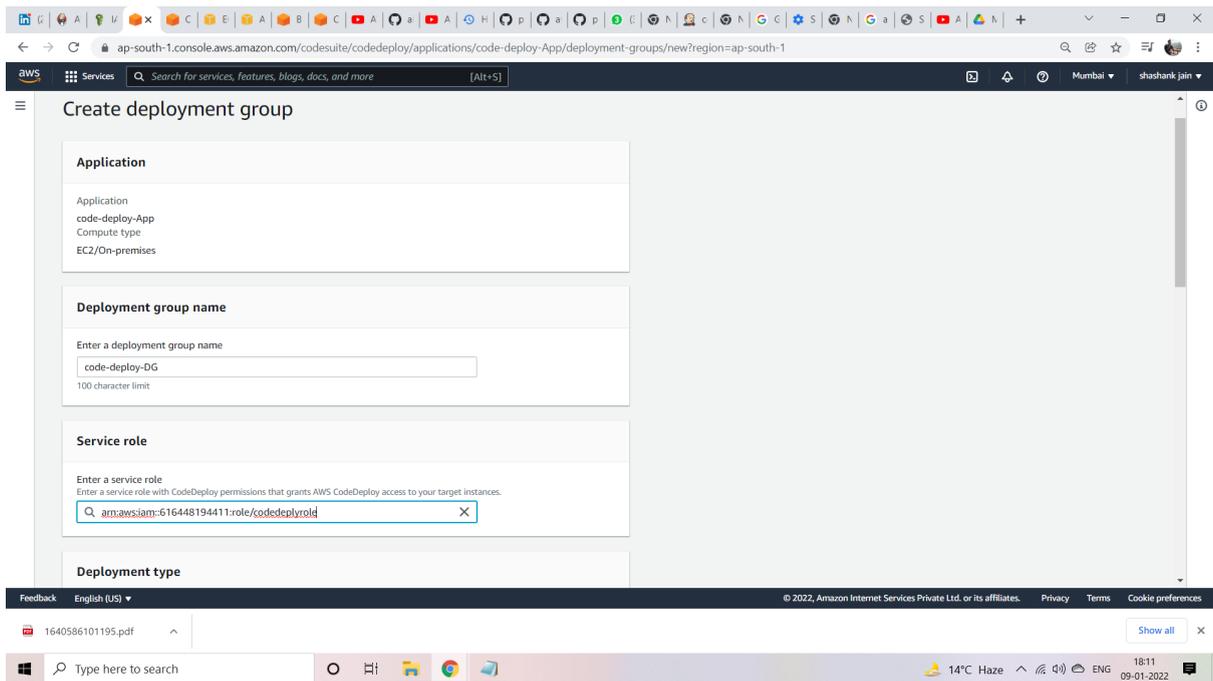
# Step 7 - Create CodeDeploy



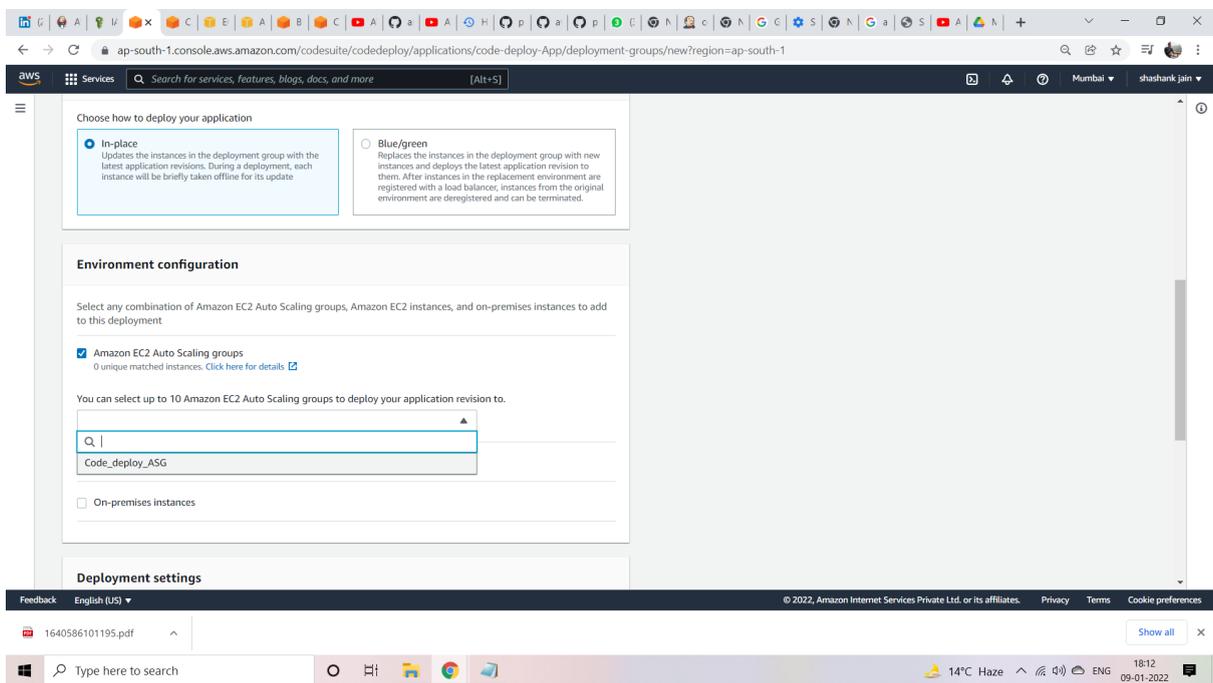
## Create Application



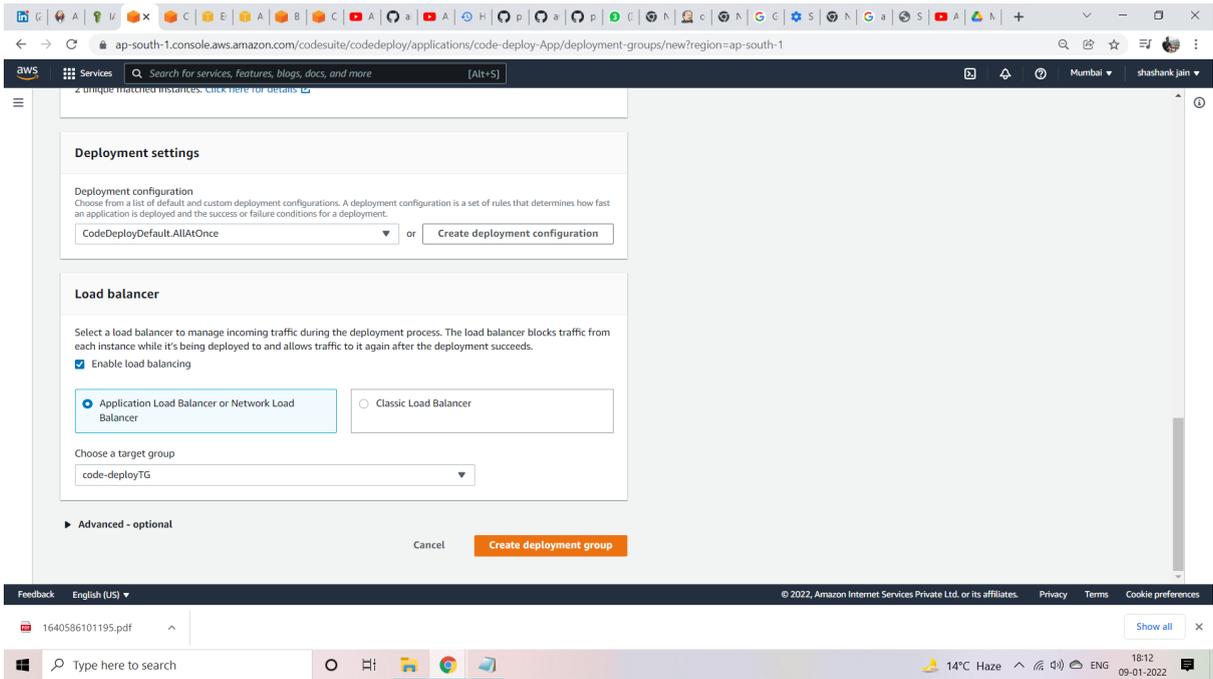
## Now Create Deployment Group.



## Now Choose Amazon Ec2 ASG which we have Created earlier.

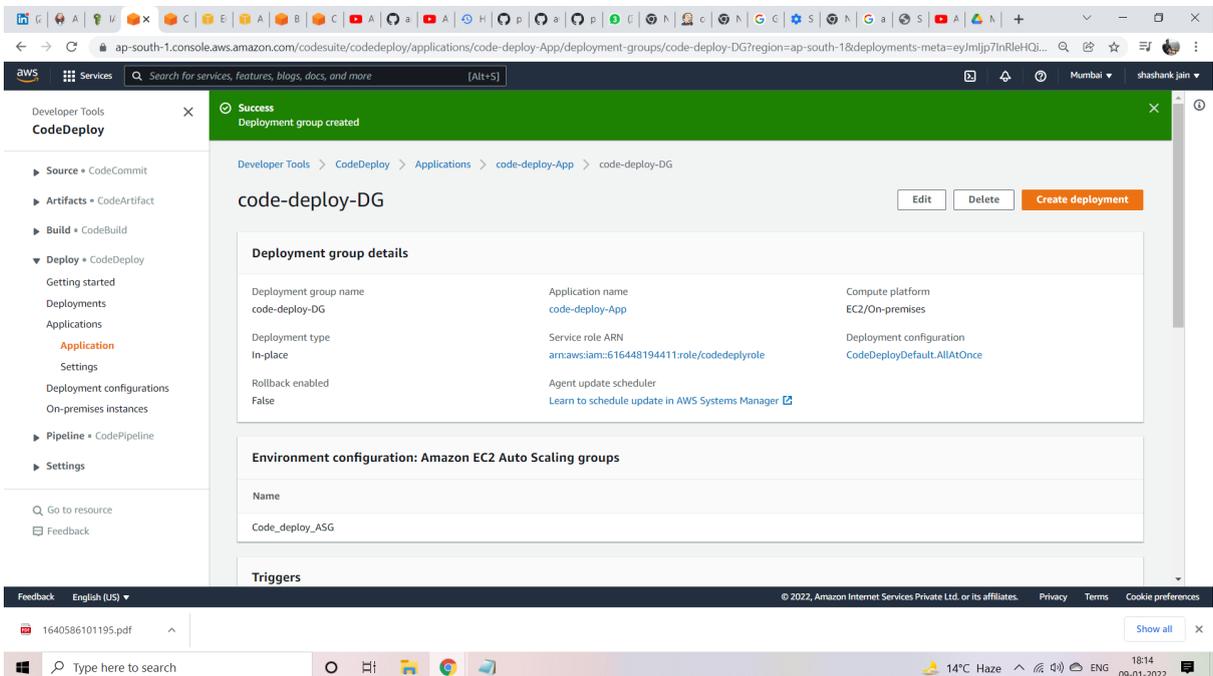


## Now choose Application Load balancer which we have created earlier.



Deployment Group Is Created.

Now we will use Jenkins for Creating a pipeline that will use git as a source and will run the CodeDeploy.

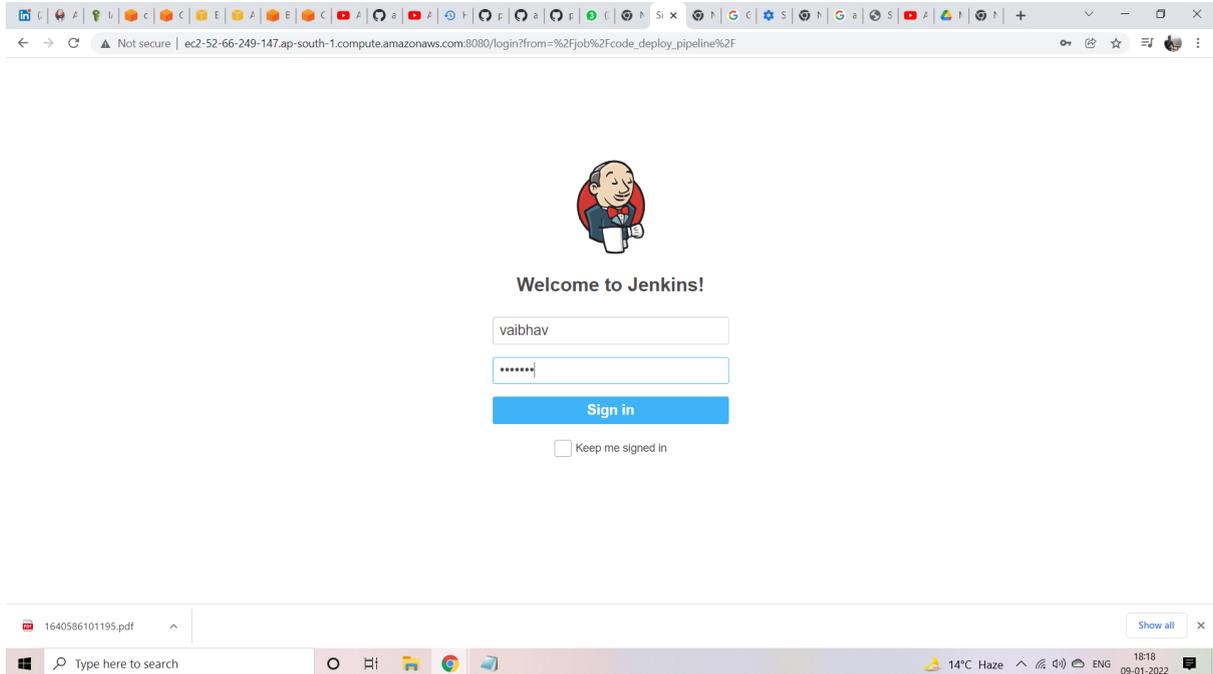


# Step 8 - Create an S3 bucket.

The screenshot displays the AWS S3 console interface. The browser address bar shows the URL: `s3.console.aws.amazon.com/s3/buckets/code-deploy-buck1?region=ap-south-1&tab=objects`. The console header includes the AWS logo, a search bar, and the user's name 'shashank jain'. The left sidebar shows the 'Amazon S3' navigation menu with options like Buckets, Access Points, and Storage Lens. The main content area is titled 'code-deploy-buck1' and has tabs for Objects, Properties, Permissions, Metrics, Management, and Access Points. The 'Objects' tab is active, showing a list of objects. There is one object listed: 'code-deploy-folder/' of type 'Folder'. Above the list are various action buttons such as 'Copy S3 URI', 'Copy URL', 'Download', 'Open', 'Delete', 'Actions', 'Create folder', and 'Upload'. A search bar and a 'Show versions' toggle are also present.

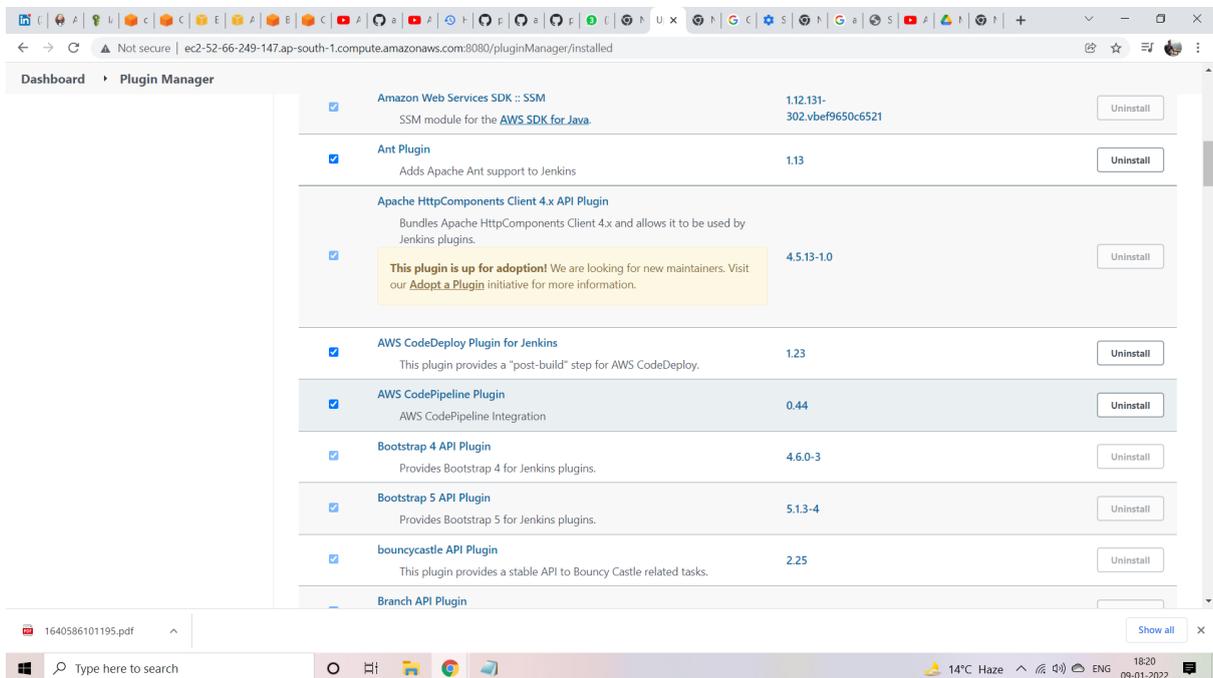
<input type="checkbox"/>	Name	Type	Last modified	Size	Storage class
<input type="checkbox"/>	code-deploy-folder/	Folder	-	-	-

# Step 9 - Go to Jenkins Server



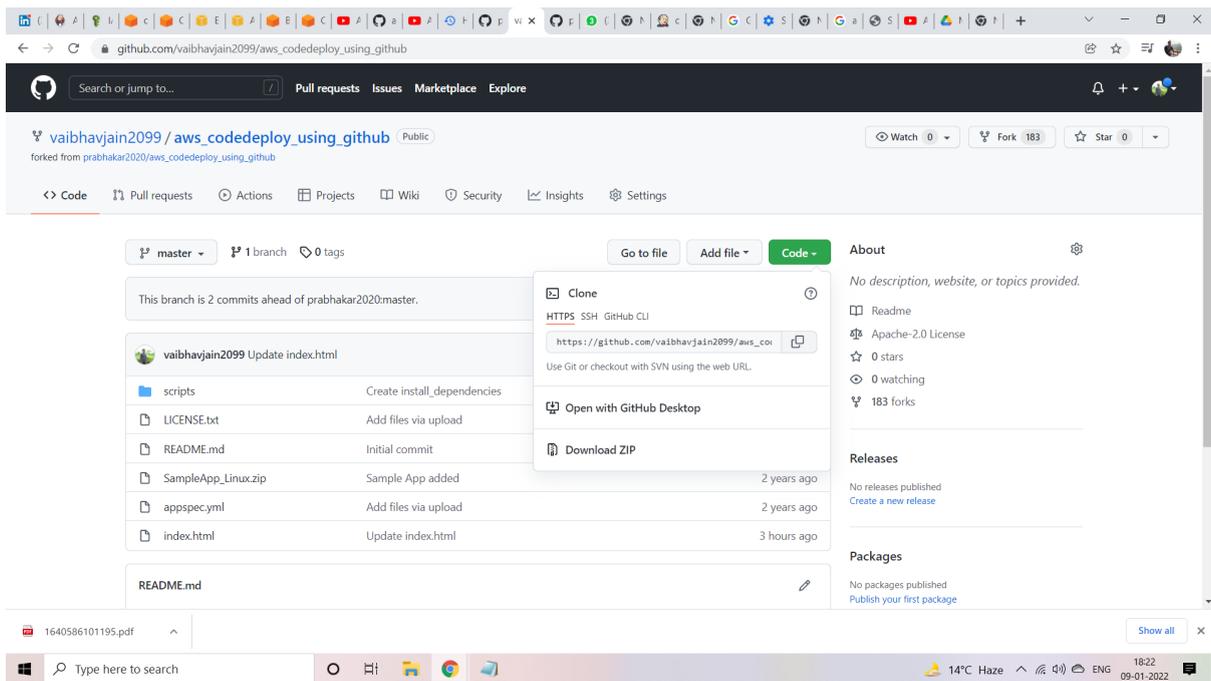
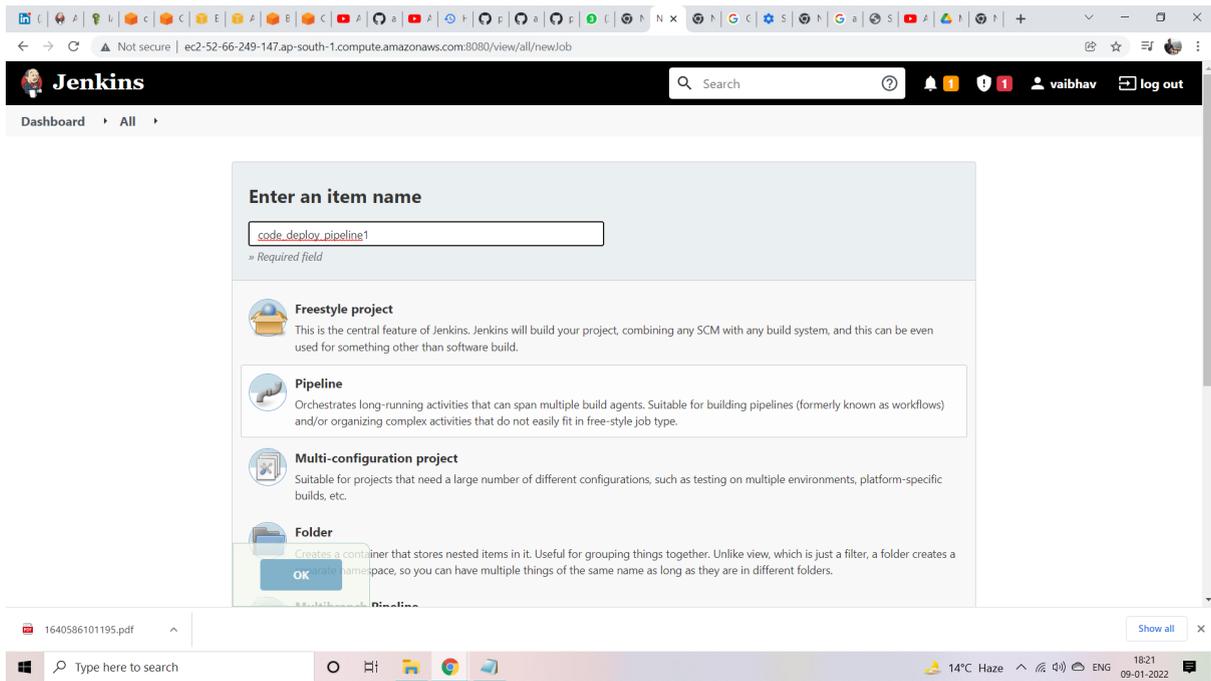
The screenshot shows the Jenkins login page in a web browser. The browser's address bar displays the URL: `ec2-52-66-249-147.ap-south-1.compute.amazonaws.com:8080/login?from=%2Fjob%2Fcode_deploy_pipeline%2F`. The page features the Jenkins logo (a cartoon character) and the text "Welcome to Jenkins!". Below the logo, there is a text input field containing the username "vaibhav", a password input field with masked characters, and a blue "Sign in" button. A checkbox labeled "Keep me signed in" is located below the sign-in button. The browser's taskbar at the bottom shows the search bar, task icons, and system tray with a temperature of 14°C and the date 09-01-2022.

## - Install AWS Code Deploy plugin in it



The screenshot displays the Jenkins Plugin Manager interface. The browser's address bar shows the URL: `ec2-52-66-249-147.ap-south-1.compute.amazonaws.com:8080/pluginManager/installed`. The page title is "Dashboard - Plugin Manager". A list of installed plugins is shown, each with a checkbox, a description, version number, and an "Uninstall" button. The "AWS CodeDeploy Plugin for Jenkins" is highlighted in blue. A yellow warning box is visible over the "Apache HttpComponents Client 4.x API Plugin" entry, stating: "This plugin is up for adoption! We are looking for new maintainers. Visit our Adopt a Plugin initiative for more information." The taskbar at the bottom shows the search bar, task icons, and system tray with a temperature of 14°C and the date 09-01-2022.

Plugin Name	Description	Version	Action
<input checked="" type="checkbox"/> Amazon Web Services SDK :: SSM	SSM module for the <a href="#">AWS SDK for Java</a>	1.12.131-302.vbef9650c6521	Uninstall
<input checked="" type="checkbox"/> Ant Plugin	Adds Apache Ant support to Jenkins	1.13	Uninstall
<input checked="" type="checkbox"/> Apache HttpComponents Client 4.x API Plugin	Bundles Apache HttpComponents Client 4.x and allows it to be used by Jenkins plugins.	4.5.13-1.0	Uninstall
<input checked="" type="checkbox"/> AWS CodeDeploy Plugin for Jenkins	This plugin provides a "post-build" step for AWS CodeDeploy.	1.23	Uninstall
<input checked="" type="checkbox"/> AWS CodePipeline Plugin	AWS CodePipeline Integration	0.44	Uninstall
<input checked="" type="checkbox"/> Bootstrap 4 API Plugin	Provides Bootstrap 4 for Jenkins plugins.	4.6.0-3	Uninstall
<input checked="" type="checkbox"/> Bootstrap 5 API Plugin	Provides Bootstrap 5 for Jenkins plugins.	5.1.3-4	Uninstall
<input checked="" type="checkbox"/> bouncycastle API Plugin	This plugin provides a stable API to Bouncy Castle related tasks.	2.25	Uninstall
<input checked="" type="checkbox"/> Branch API Plugin			Uninstall



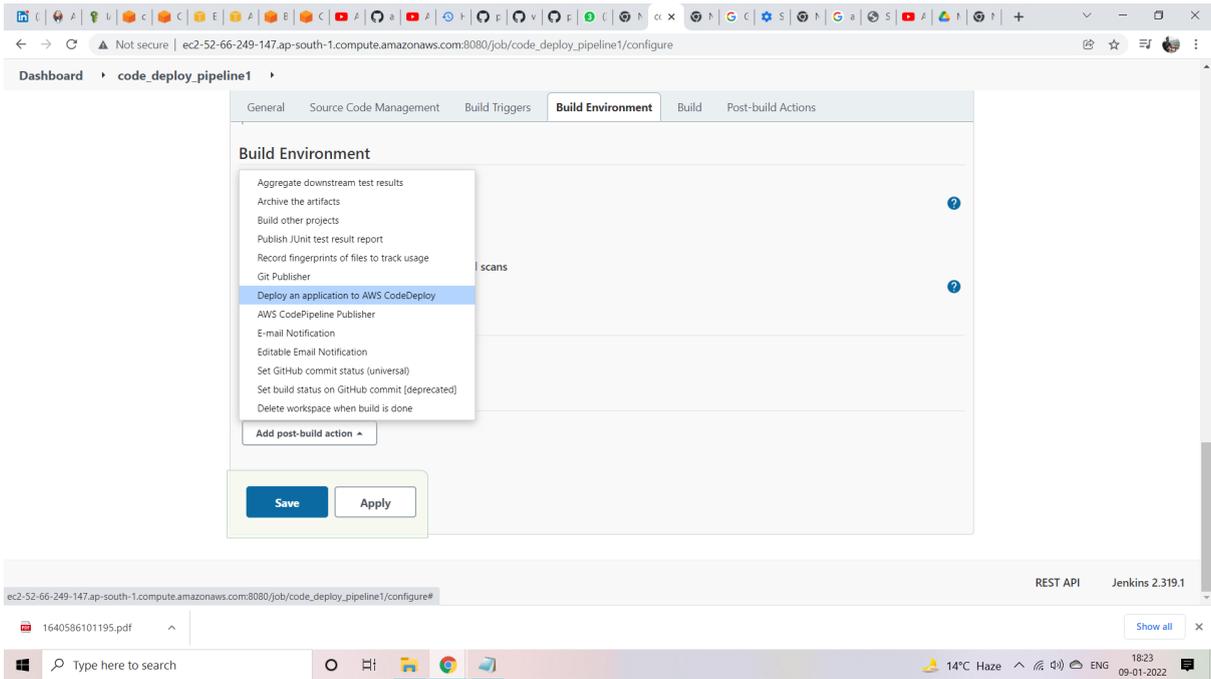
[https://github.com/vaibhavjain2099/aws\\_codedeploy\\_using\\_github](https://github.com/vaibhavjain2099/aws_codedeploy_using_github)

Create a Jenkins pipeline by taking GIT as a source, Trigger - Poll SCM ( \* \* \* \* \* ) which means it will run job in every minute. Put S3 Bucket name etc.

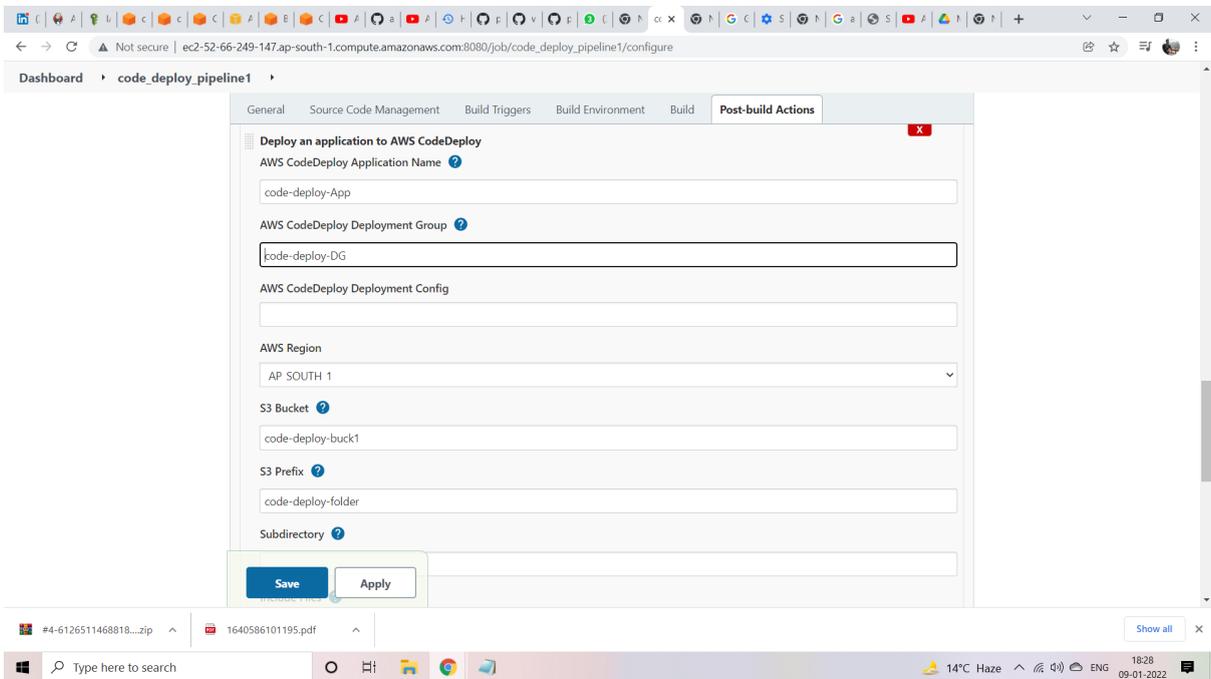
The screenshot shows the AWS CodeDeploy console configuration page for a pipeline named 'code\_deploy\_pipeline1'. The 'Source Code Management' tab is active. Under 'General', there are checkboxes for 'Disable this project' and 'Execute concurrent builds if necessary'. The 'Source Code Management' section has radio buttons for 'None', 'AWS CodePipeline', and 'Git' (which is selected). Below this, there is a 'Repositories' section with a 'Repository URL' field containing 'https://github.com/vaibhavjain2099/aws\_codedeploy\_using\_github.git' and a 'Credentials' dropdown menu with 'vaibhavjain2099/\*' selected. There are 'Add Repository', 'Save', and 'Apply' buttons at the bottom.

The screenshot shows the same AWS CodeDeploy console configuration page, but with the 'Build Triggers' tab active. Under 'Build Triggers', there are several checkboxes: 'Trigger builds remotely (e.g., from scripts)', 'Build after other projects are built', 'Build periodically', 'GitHub hook trigger for GITScm polling', and 'Poll SCM' (which is checked). Below these is a 'Schedule' field with the value '\*\*\*\*\*'. A warning message states: 'Do you really mean "every minute" when you say "\*\*\*\*\*"? Perhaps you meant "H\*\*\*\*\*" to poll once per hour. Would last have run at Sunday, January 9, 2022 12:53:31 PM UTC; would next run at Sunday, January 9, 2022 12:53:31 PM UTC.' There is also an 'Ignore post-commit hooks' checkbox. The 'Build Environment' section below has checkboxes for 'Delete workspace before build starts' and 'Use secret text(s) or file(s)'. 'Save' and 'Apply' buttons are visible at the bottom.

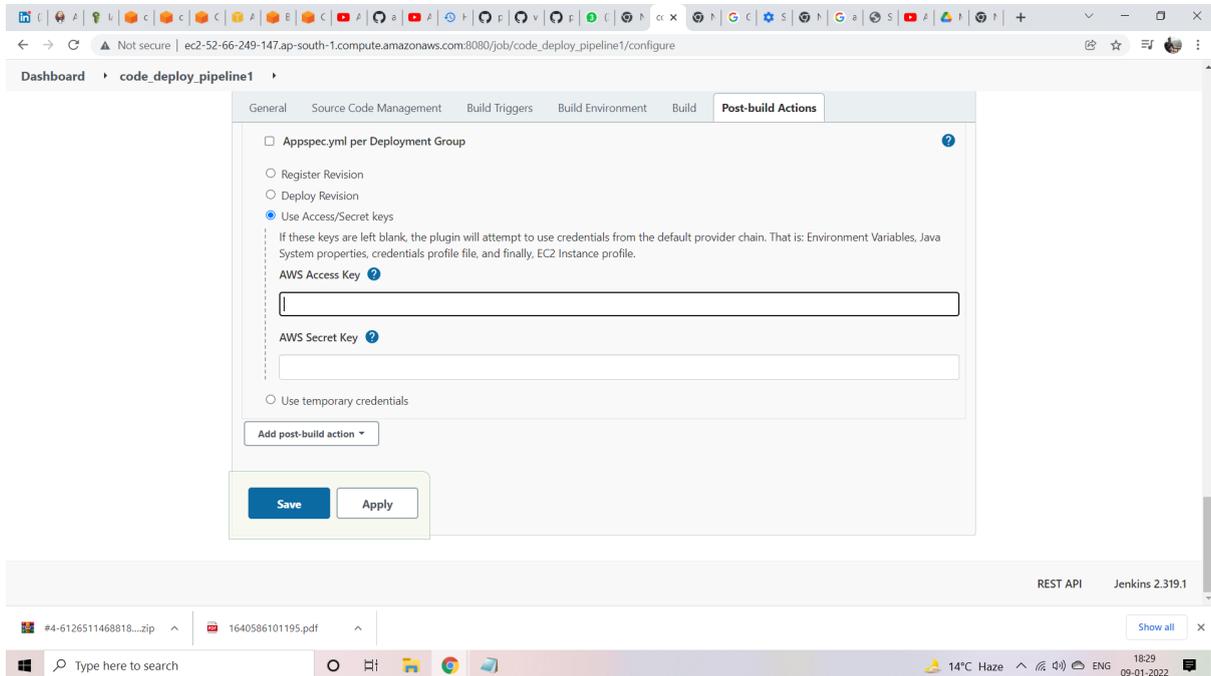
Choose post-build-action as deploy an application AWS CodeDeploy



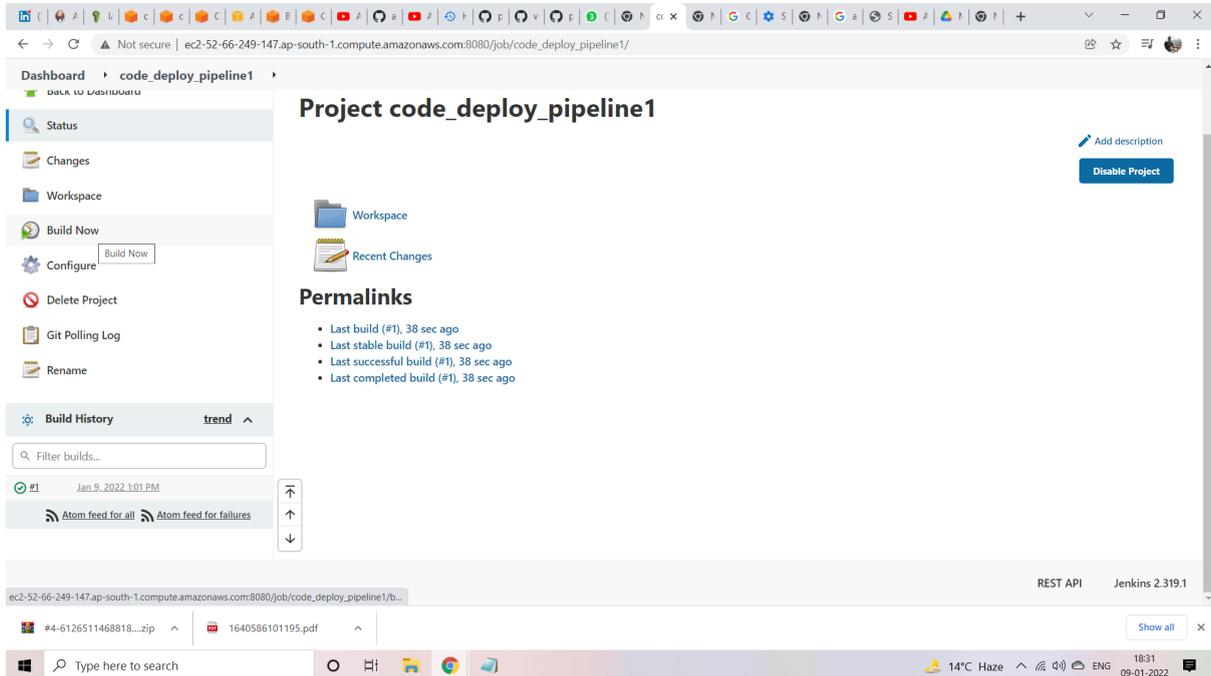
Then give all info of CodeDeploy like app name , deployment group name , s3 bucket and prefix name.



Then for Authorization we can use Aws Access keys and Secret keys.

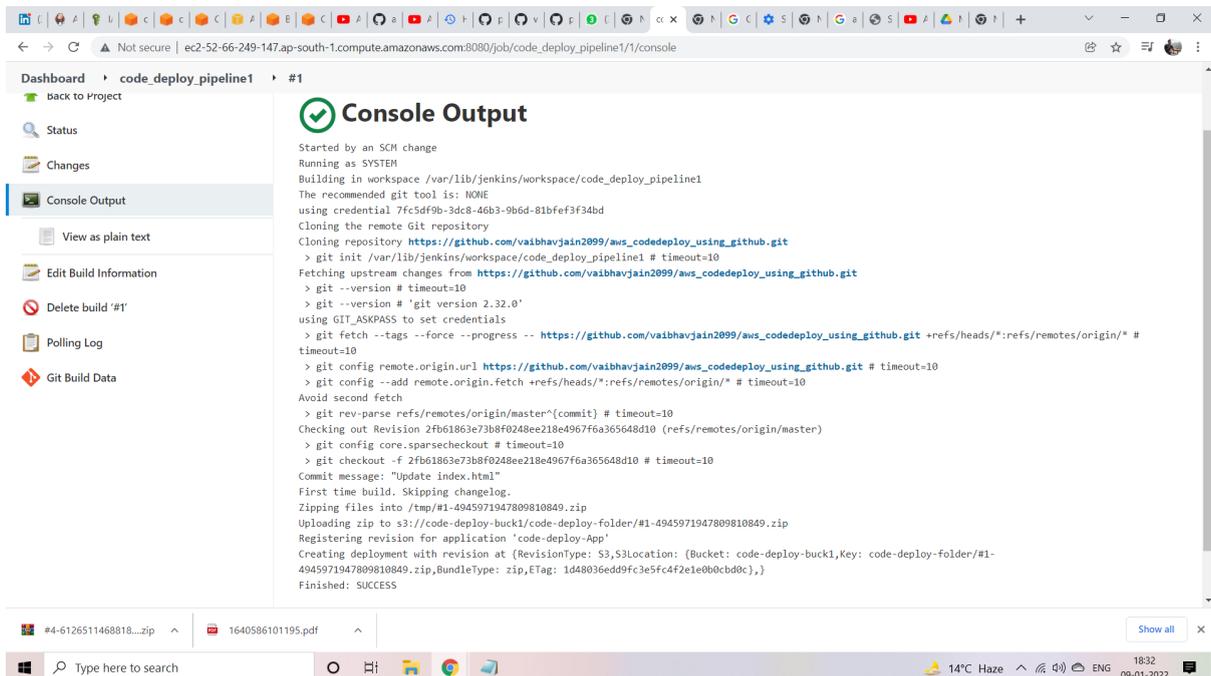


# Step 10 - Build Jenkins Pipeline



The screenshot shows the Jenkins Project Overview page for 'code\_deploy\_pipeline1'. The left sidebar contains navigation options: Status, Changes, Workspace, Build Now, Configure, Delete Project, Git Polling Log, and Rename. The main content area displays the project name, workspace information, and a list of permalinks for the last build, last stable build, last successful build, and last completed build, all of which occurred 38 seconds ago. The bottom of the page shows the REST API and Jenkins version (2.319.1).

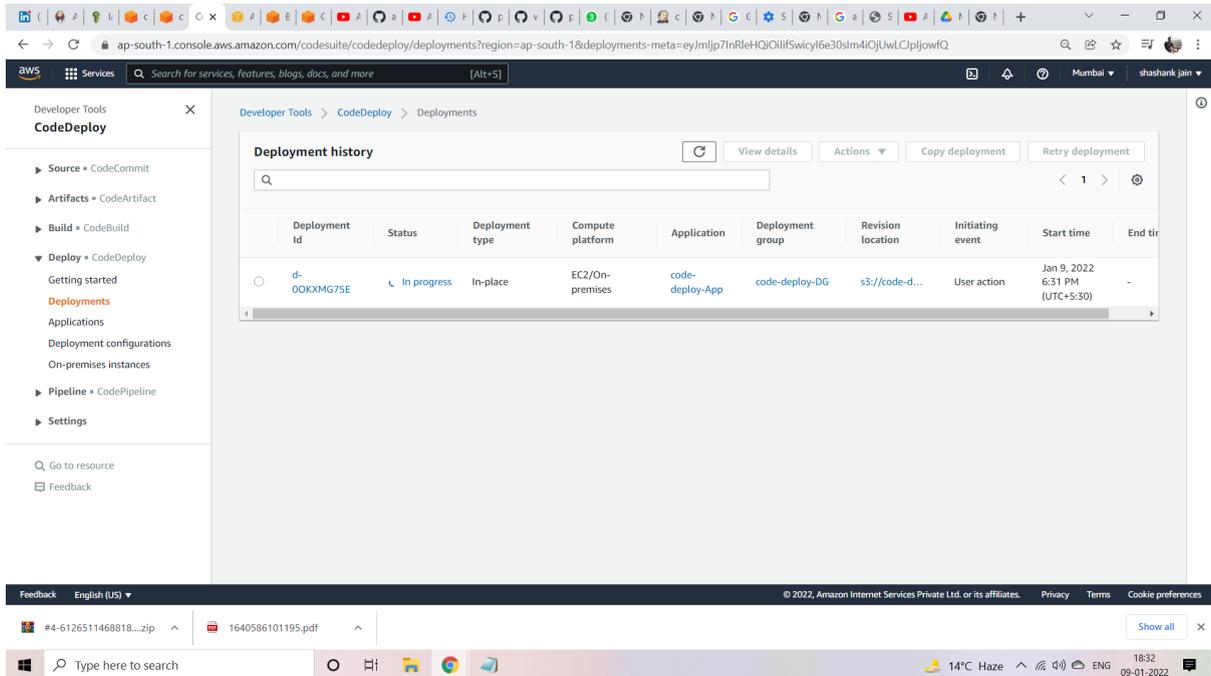
## Check console output.



The screenshot shows the Jenkins Console Output for build #1. The output text is as follows:

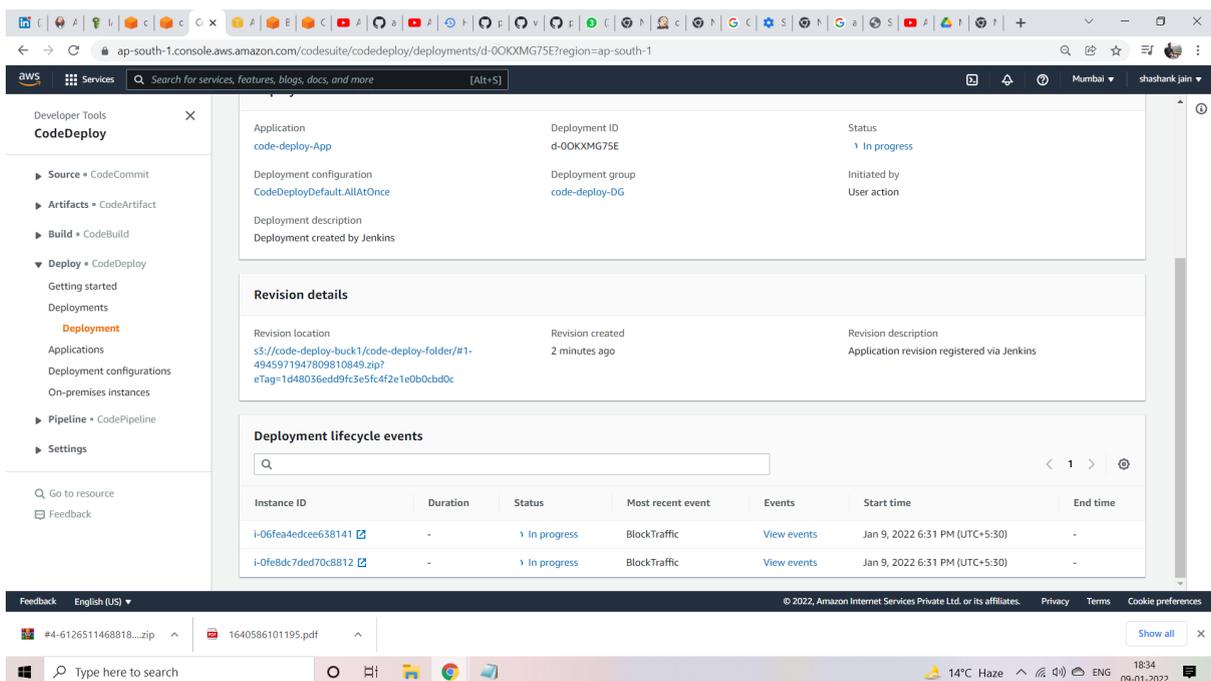
```
Started by an SCM change
Running as SYSTEM
Building in workspace /var/lib/jenkins/workspace/code_deploy_pipeline1
The recommended git tool is: NONE
using credential 7fc5df9b-3dc8-46b3-9b6d-81bfef3f34bd
Cloning the remote Git repository
Cloning repository https://github.com/vaibhavjain2099/aws_codedeploy_using_github.git
> git init /var/lib/jenkins/workspace/code_deploy_pipeline1 # timeout=10
Fetching upstream changes from https://github.com/vaibhavjain2099/aws_codedeploy_using_github.git
> git --version # timeout=10
> git --version # 'git version 2.32.0'
using GIT_ASKPASS to set credentials
> git fetch --tags --force --progress -- https://github.com/vaibhavjain2099/aws_codedeploy_using_github.git +refs/heads/*:refs/remotes/origin/* #
timeout=10
> git config remote.origin.url https://github.com/vaibhavjain2099/aws_codedeploy_using_github.git # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
Avoid second fetch
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 2fb61863e73b8f0248ee218e4967f6a365648d10 (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 2fb61863e73b8f0248ee218e4967f6a365648d10 # timeout=10
Commit message: "Update index.html"
First time build. Skipping changelog.
Zipping files into /tmp/#1-4945971947809810849.zip
Uploading zip to s3://code-deploy-buck1/code-deploy-folder/#1-4945971947809810849.zip
Registering revision for application 'code-deploy-App'
Creating deployment with revision at {RevisionType: S3,S3Location: {Bucket: code-deploy-buck1,Key: code-deploy-folder/#1-4945971947809810849.zip,BundleType: zip,ETag: 1d48036edd9fc3e5fc4f42e1e0bc0c},}
Finished: SUCCESS
```

As soon as we will hit the Build Now button the job will run and trigger for code deploy deployment . we can see here it is in progress.



The screenshot shows the AWS CodeDeploy console. The left sidebar contains navigation options: Source (CodeCommit), Artifacts (CodeArtifact), Build (CodeBuild), Deploy (CodeDeploy), Pipeline (CodePipeline), and Settings. The main area displays the 'Deployment history' table. A single deployment is listed with the following details:

Deployment Id	Status	Deployment type	Compute platform	Application	Deployment group	Revision location	Initiating event	Start time	End time
d-00KXMG75E	In progress	In-place	EC2/On-premises	code-deploy-App	code-deploy-DG	s3://code-d...	User action	Jan 9, 2022 6:31 PM (UTC+5:30)	-



The screenshot shows the AWS CodeDeploy console with the details for deployment ID d-00KXMG75E. The details are organized into sections:

- Application:** code-deploy-App
- Deployment ID:** d-00KXMG75E
- Status:** In progress
- Deployment configuration:** CodeDeployDefault.AllAtOnce
- Deployment group:** code-deploy-DG
- Initiated by:** User action
- Deployment description:** Deployment created by Jenkins

**Revision details:**

- Revision location:** s3://code-deploy-buck1/code-deploy-folder/#1-4945971947809810845.zip?eTag=1d48036edd9f63e5f472e1e0b0cbd0c
- Revision created:** 2 minutes ago
- Revision description:** Application revision registered via Jenkins

**Deployment lifecycle events:**

Instance ID	Duration	Status	Most recent event	Events	Start time	End time
i-06fea4edcee638141	-	In progress	BlockTraffic	<a href="#">View events</a>	Jan 9, 2022 6:31 PM (UTC+5:30)	-
i-0fe8dc7ded70c8812	-	In progress	BlockTraffic	<a href="#">View events</a>	Jan 9, 2022 6:31 PM (UTC+5:30)	-

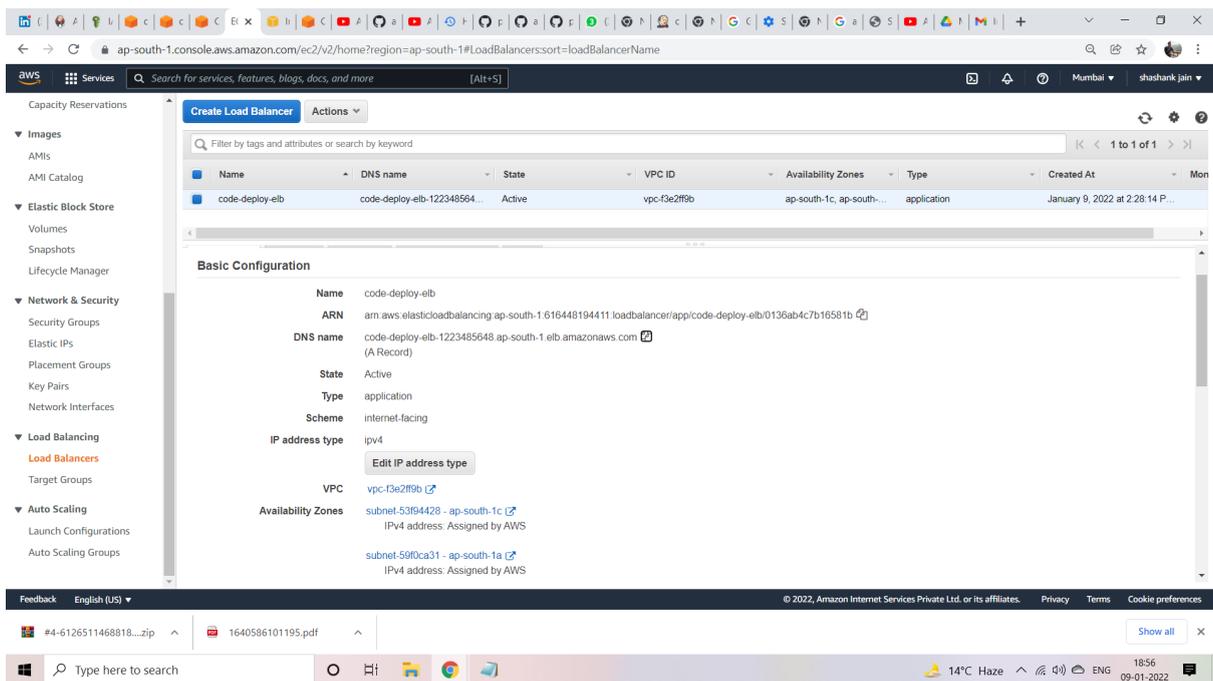
The screenshot displays the AWS CodeDeploy console interface. The main content area shows a deployment event with a table of events. The table has columns for Event, Duration, Status, Error code, Start time, and End time. The 'BlockTraffic' event is currently 'In progress', while all other events are 'Pending'.

Event	Duration	Status	Error code	Start time	End time
BeforeBlockTraffic	less than one second	Succeeded	-	Jan 9, 2022 6:31 PM (UTC+5:30)	Jan 9, 2022 6:31 PM (UTC+5:30)
BlockTraffic	-	In progress	-	Jan 9, 2022 6:31 PM (UTC+5:30)	-
AfterBlockTraffic	-	Pending	-	-	-
ApplicationStop	-	Pending	-	-	-
DownloadBundle	-	Pending	-	-	-
BeforeInstall	-	Pending	-	-	-
Install	-	Pending	-	-	-
AfterInstall	-	Pending	-	-	-
ApplicationStart	-	Pending	-	-	-
ValidateService	-	Pending	-	-	-
BeforeAllowTraffic	-	Pending	-	-	-
AllowTraffic	-	Pending	-	-	-
AfterAllowTraffic	-	Pending	-	-	-

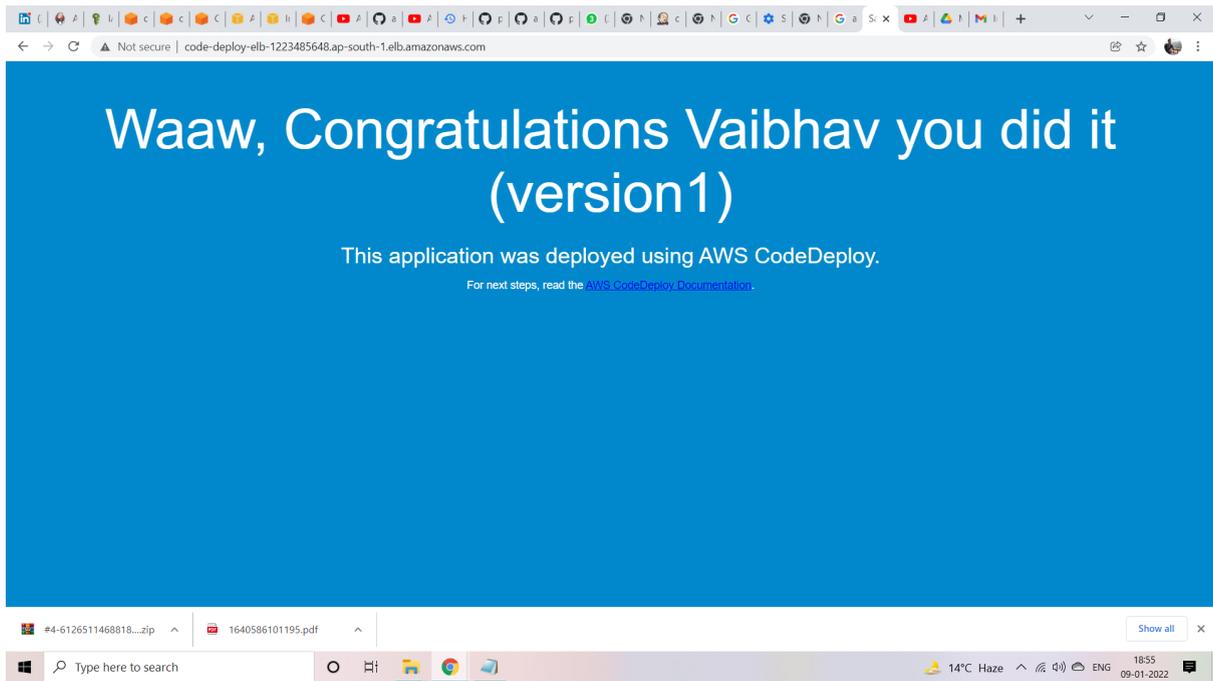
Here we can see , it take time , as we have choosed the option of build all at once.

## Step 11 -

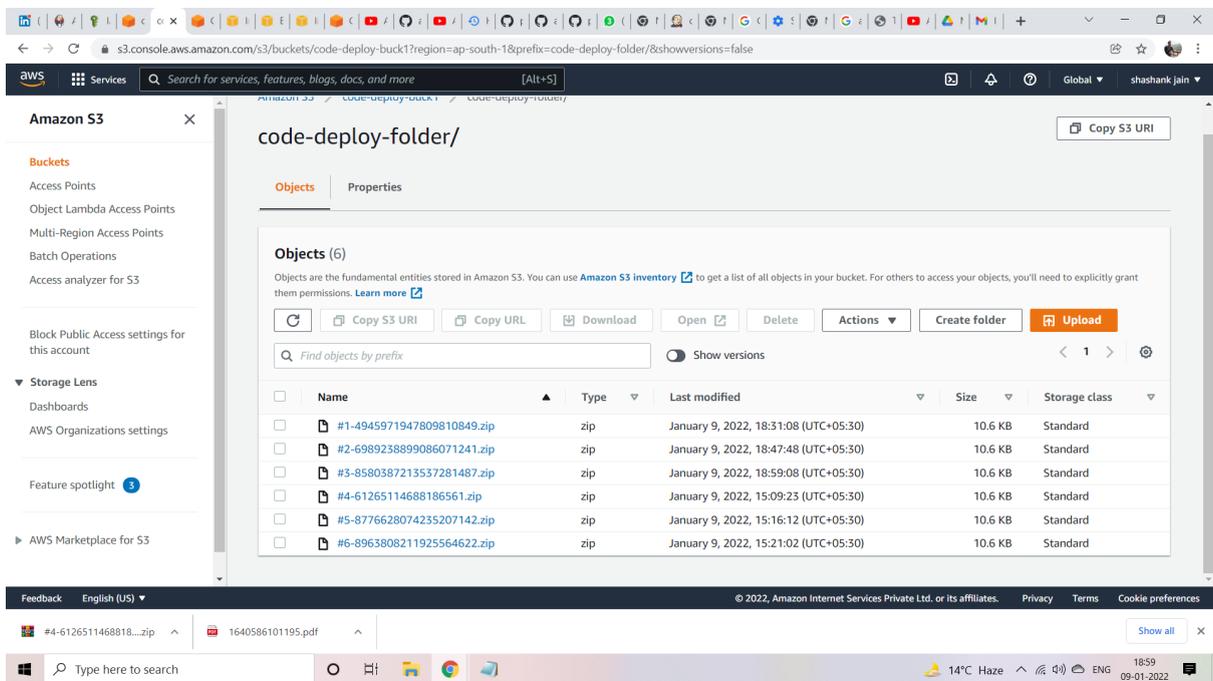
Go to AWS Load balancer portal and go to that particular ELB you have Created and copy the DNS and paste it on browser , your application will be running .

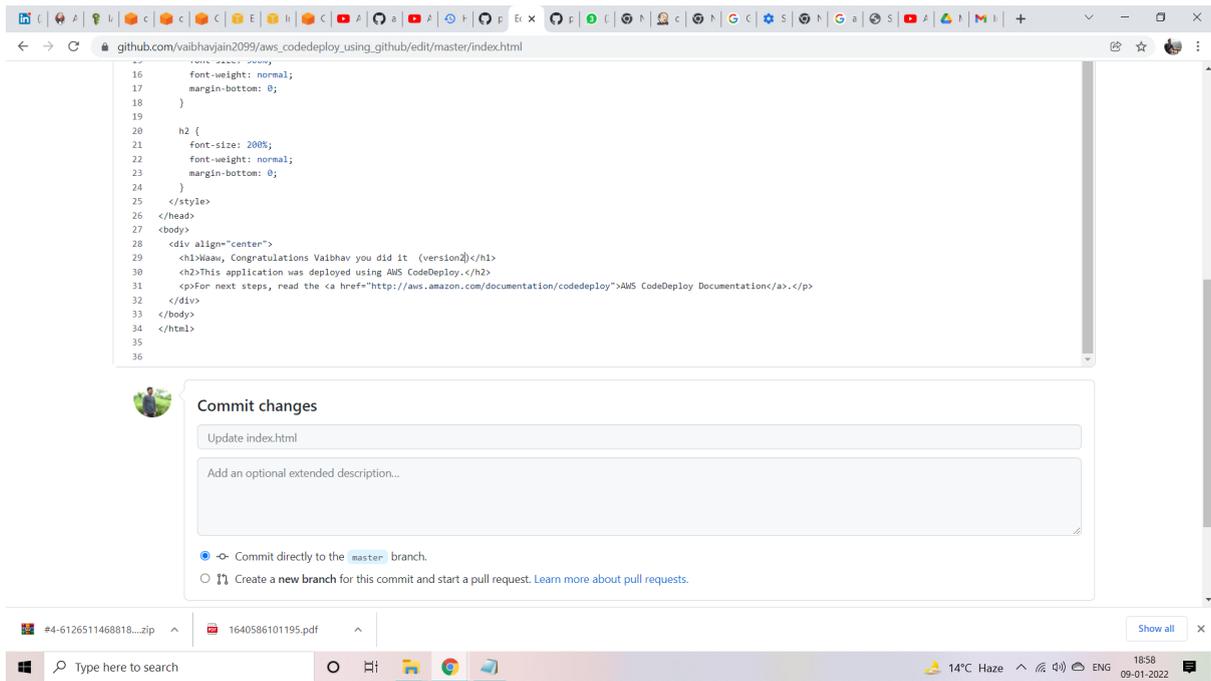


This is the first version of application. Now if i made any change to the code it will automatically reflect to the Web browser within some minutes.

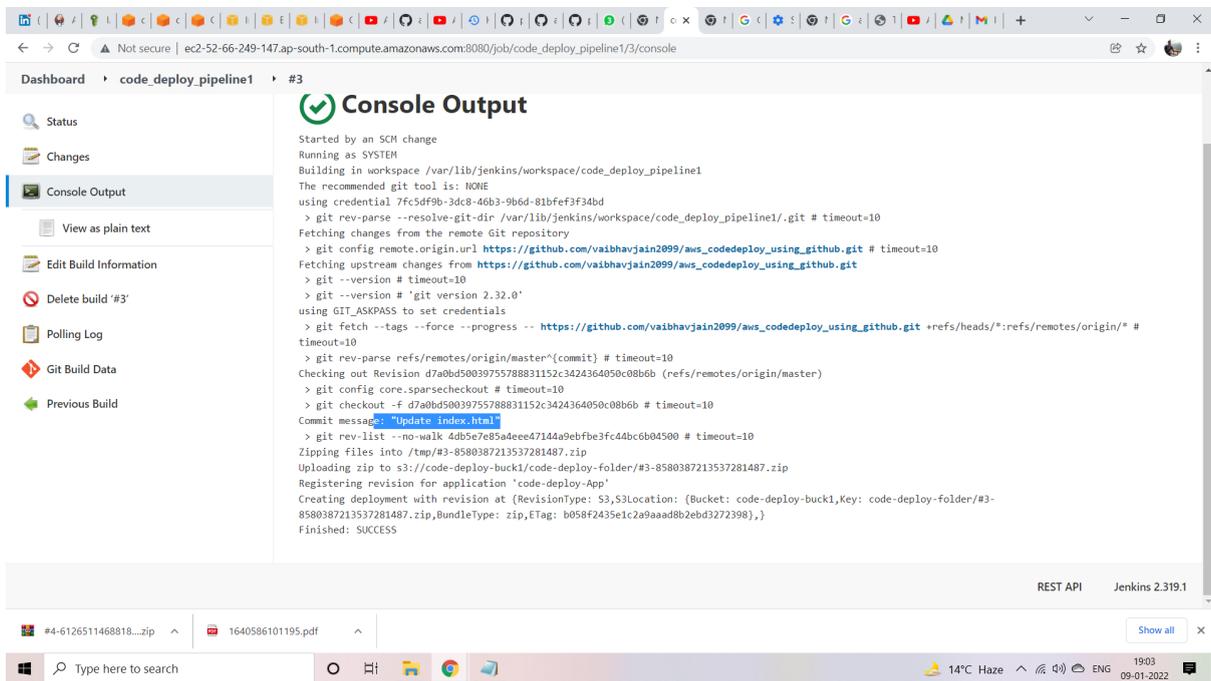


We can check that as soon as we run the jenkins job it run the deployment the data will get stored in s3 bucket.





We can see as soon as we changed the code in index.html file , a new deployment will be triggered by jenkins job .



# In progress

The screenshot shows the AWS CodeDeploy console's 'Deployment history' page. The left sidebar contains navigation options for Source, Artifacts, Build, Deploy, Applications, Pipeline, and Settings. The main content area displays a table of deployment records.

Deployment Id	Status	Deployment type	Compute platform	Application	Deployment group	Revision location	Initiating event	Start time	End time
d-THQHIM75E	In progress	In-place	EC2/On-premises	code-deploy-App	code-deploy-DG	s3://code-d...	User action	Jan 9, 2022 6:59 PM (UTC+5:30)	-
d-XM1GFL75E	Succeeded	In-place	EC2/On-premises	code-deploy-App	code-deploy-DG	s3://code-d...	User action	Jan 9, 2022 6:47 PM (UTC+5:30)	Jan 9, 2022 6:53 PM (UTC+5:30)
d-00KXMG75E	Failed	In-place	EC2/On-premises	code-deploy-App	code-deploy-DG	s3://code-d...	User action	Jan 9, 2022 6:31 PM (UTC+5:30)	Jan 9, 2022 6:36 PM (UTC+5:30)

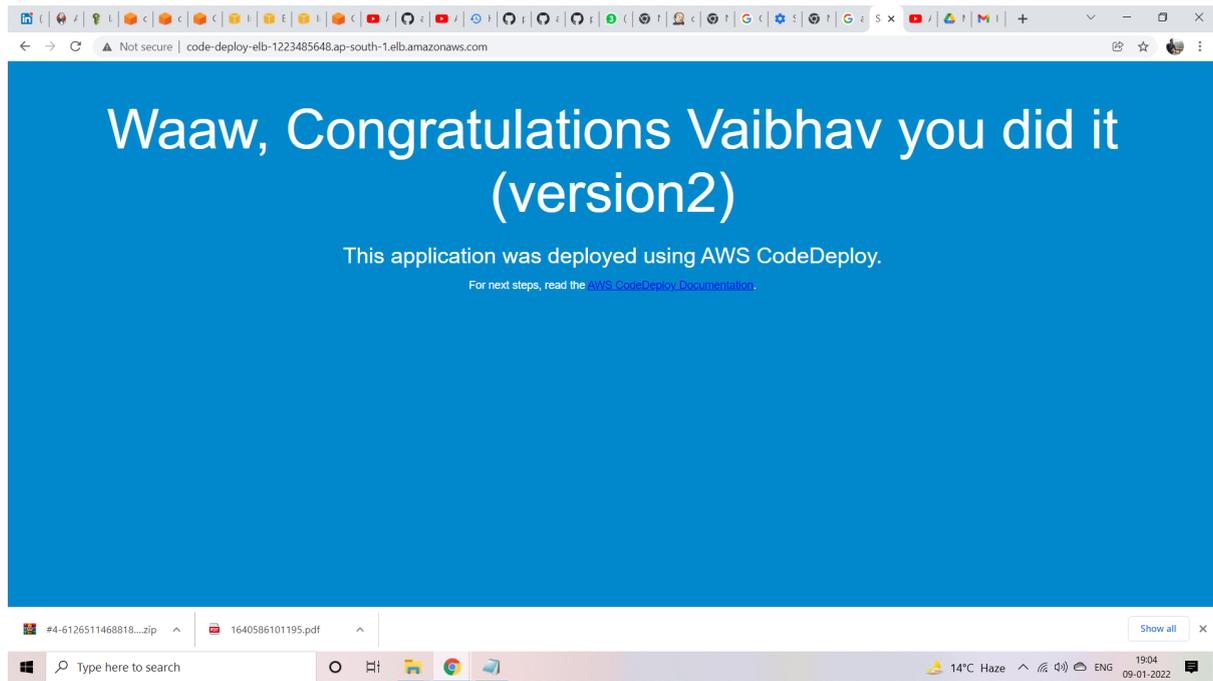
The screenshot shows the 'Revision details' page for a specific deployment. It provides information about the revision location, creation time, and a detailed event log.

Revision location: `s3://code-deploy-buck1/code-deploy-folder/#3-8580387213537281487.zip?eTag=b058f2435e1c2a9aaad8b2ebd3272398`

Revision created: 4 minutes ago

Revision description: Application revision registered via Jenkins

Event	Duration	Status	Error code	Start time	End time
BeforeBlockTraffic	less than one second	Succeeded	-	Jan 9, 2022 6:59 PM (UTC+5:30)	Jan 9, 2022 6:59 PM (UTC+5:30)
BlockTraffic	5 minutes 1 second	Succeeded	-	Jan 9, 2022 6:59 PM (UTC+5:30)	Jan 9, 2022 7:04 PM (UTC+5:30)
AfterBlockTraffic	less than one second	Succeeded	-	Jan 9, 2022 7:04 PM (UTC+5:30)	Jan 9, 2022 7:04 PM (UTC+5:30)
ApplicationStop	less than one second	Succeeded	-	Jan 9, 2022 7:04 PM (UTC+5:30)	Jan 9, 2022 7:04 PM (UTC+5:30)
DownloadBundle	-	Pending	-	-	-
BeforeInstall	-	Pending	-	-	-
Install	-	Pending	-	-	-
AfterInstall	-	Pending	-	-	-
ApplicationStart	-	Pending	-	-	-
ValidateService	-	Pending	-	-	-
BeforeAllowTraffic	-	Pending	-	-	-
AllowTraffic	-	Pending	-	-	-



And here is the Our Final Output .

We can see that it shows the recent changed done by us in the html file.

Since we have used Auto Scaling Group and using ELB and its url to access the application , we can confirm that it scalable such that when load increases the number of servers scale up and down making sure the new servers have the updated code.

Github repo used -

[https://github.com/vaibhavjain2099/aws\\_codedeploy\\_using\\_github.git](https://github.com/vaibhavjain2099/aws_codedeploy_using_github.git)

Pardon me if i missed any step in between.

**Thank You**

