

Deploy Django Application using Code Pipeline

Steps

Create a repo on github clone it and paste Django project in that.

The screenshot shows a GitHub repository page for 'Python_Project'. The repository is public and has 1 branch and 0 tags. The main branch has 1 commit from 'Pratiksha57' made yesterday. The commit message says 'files are added'. The commit details show files like employee_app, employee_pro, scripts, templates, .gitignore, appspec.yml, manage.py, and requirements.txt being added yesterday. The repository has 0 stars, 1 watching, and 0 forks. There are sections for About, Releases, and Packages, all of which are currently empty.

--> Launch Ubuntu instance

The screenshot shows the AWS EC2 'Launch an instance' wizard. In the 'Name and tags' step, the name 'myubuntu' is entered. In the 'Application and OS Images (Amazon Machine Image)' section, the 'ubuntu' AMI is selected. The 'Summary' section shows 1 instance being launched with the following configuration: Software Image (AMI) is Canonical, Ubuntu, 24.04, amd64; Virtual server type (instance type) is t2.micro; Firewall (security group) is New security group; and Storage (volumes) is not specified. There are 'Cancel' and 'Launch instance' buttons at the bottom.

--> use Linux key pair and Linux security group

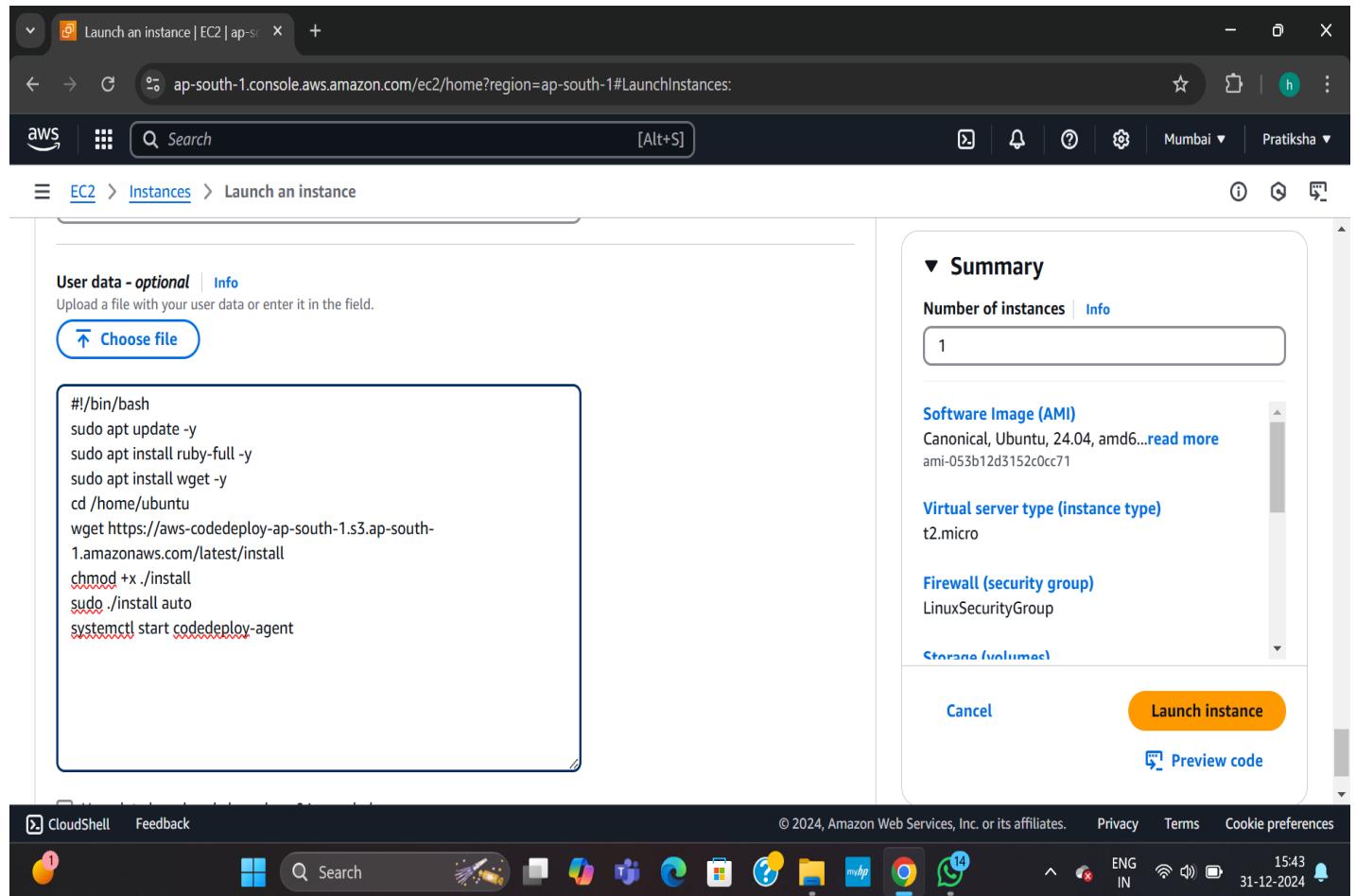
The screenshot shows the 'Launch an instance' wizard in the AWS Management Console. The left sidebar lists 'Key pair name - required' (selected 'LinuxKeyPair'), 'Network settings' (selected 'Subnet vpc-09867797ad23c076c'), 'Auto-assign public IP' (selected 'Enable'), and 'Firewall (security groups)' (selected 'Select existing security group'). The right sidebar displays the 'Summary' section with 1 instance, AMI 'Canonical, Ubuntu, 24.04, amd6...', instance type 't2.micro', security group 'LinuxSecurityGroup', and a large orange 'Launch instance' button.

--> go in advanced details at the bottom-> under IAM instance profile select ece-to-codedeploy role

The screenshot shows the 'Advanced details' section of the 'Launch an instance' wizard. It includes fields for 'Domain join directory' (selected 'Select'), 'IAM instance profile' (selected 'ec2-code-deploy31'), 'Hostname type' (selected 'IP name'), 'DNS Hostname' checkboxes for IPv4 and IPv6, and 'Instance auto-recovery' (selected 'Select'). The right sidebar remains identical to the previous screenshot, showing the summary and launch instance button.

--> scroll at bottom--> write script in that

```
#!/bin/bash
sudo apt update -y
sudo apt install ruby-full -y
sudo apt install wget -y
cd /home/ubuntu
wget https://aws-codedeploy-ap-south-1.s3.ap-south-1.amazonaws.com/latest/install
chmod +x ./install
sudo ./install auto
systemctl start codedeploy-agent
```



The screenshot shows the AWS EC2 Instances page. On the left, a sidebar lists navigation options like Dashboard, EC2 Global View, Events, Instances (selected), Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, and Images. The main content area displays 'Instances (1/3)' with a table. The table has columns for Name, Instance ID, Instance state, Instance type, Status check, and Alarm status. It shows three rows: 'myUbuntu' (terminated, t2.micro), 'myubuntu' (running, t2.micro), and 'myLinux' (terminated, t2.micro). Below the table, a detailed view for 'myubuntu' is shown with tabs for Details, Status and alarms, Monitoring, Security, Networking, Storage, and Tags. The Details tab is selected, showing the Instance ID as i-03c9ae2c74ffb41ff, a Public IPv4 address of 15.207.248.22, and a Private IPv4 address of 172.31.1.107.

--> connect the instance
 --> open the putty
 --> connect the instance using putty
 --> then type the command
 --> sudo su
 --> sudo apt update
 --> check the status using 'systemctl status codedeploy-agent'

```
root@ip-172-31-1-107:~# /home/ubuntu
To see these additional updates run: apt list --upgradable
Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status

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

Ubuntu comes with ABSOLUTELY NO WARRANTY, to the extent permitted by
applicable law.

To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.

root@ip-172-31-1-107:~# sudo su
root@ip-172-31-1-107:~/home/ubuntu# sudo apt update
Hit:1 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble InRelease
Hit:2 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-updates InRelease
Hit:3 http://ap-south-1.ec2.archive.ubuntu.com/ubuntu noble-backports InRelease
Hit:4 http://security.ubuntu.com/ubuntu noble-security InRelease
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
58 packages can be upgraded. Run 'apt list --upgradable' to see them.
root@ip-172-31-1-107:~/home/ubuntu# systemctl status codedeploy-agent
● codedeploy-agent.service - LSB: AWS CodeDeploy Host Agent
   Loaded: loaded (/etc/init.d/codedeploy-agent; generated)
   Active: active (running) since Tue 2024-12-31 10:14:48 UTC; 2min 4s ago
     Docs: man:systemd-sysv-generator(8)
   Process: 1959 ExecStart=/etc/init.d/codedeploy-agent start (code=exited, status=0/SUCCESS)
   Tasks: 3 (limit: 1130)
   Memory: 64.9M (peak: 67.5M)
      CPU: 887ms
      CGroup: /system.slice/codedeploy-agent.service
           └─1965 "codedeploy-agent: master 1965"
              ├─1966 "codedeploy-agent: InstanceAgent::Plugins::CodeDeployPlugin::CommandPoller of master 1965"
              └─1968 "codedeploy-agent: InstanceAgent::Plugins::CodeDeployPlugin::CommandPoller of master 1965"

Dec 31 10:14:47 ip-172-31-1-107 systemd[1]: Starting codedeploy-agent.service - LSB: AWS CodeDeploy Host Agent...
Dec 31 10:14:48 ip-172-31-1-107 codedeploy-agent[1959]: Starting codedeploy-agent:
Dec 31 10:14:48 ip-172-31-1-107 systemd[1]: Started codedeploy-agent.service - LSB: AWS CodeDeploy Host Agent.
root@ip-172-31-1-107:~/home/ubuntu#
```

--> next close putty

-->Search code deploy in services

The screenshot shows the AWS CodeDeploy service page. The left sidebar has a 'CodeDeploy' section under 'Developer Tools'. The main content area features a large heading 'AWS CodeDeploy' and the subtext 'Automate code deployments to maintain application uptime'. A call-to-action button 'Create application' is visible. The bottom of the screen shows a Windows taskbar with various icons.

--> go in applications--> create application--> give application name--> select EC2/On-premises under compute platform--> create application-->

The screenshot shows the 'Create application' wizard. Step 1 is 'Application configuration'. It includes fields for 'Application name' (set to 'app31'), 'Compute platform' (set to 'EC2/On-premises'), and 'Tags'. A 'Create application' button is at the bottom right. The bottom of the screen shows a Windows taskbar.

--> select create deployment group

The screenshot shows the AWS CodeDeploy application creation interface. On the left, a sidebar menu for 'CodeDeploy' includes options like Source, Artifacts, Build, Deploy, Pipeline, Application (selected), Settings, Deployment configurations, On-premises instances, and Pipeline. The main content area has a green header bar stating 'Application created' with a note to create a deployment group. It shows the application name 'app31' and compute platform 'EC2/On-premises'. Below this, tabs for 'Deployments', 'Deployment groups' (selected), and 'Revisions' are visible. A table titled 'Deployment groups' shows one entry: 'No deployment groups'. A message below the table says 'Before you can deploy your application using CodeDeploy, you must create a deployment group.' The browser address bar shows the URL: <https://ap-south-1.console.aws.amazon.com/codesuite/codedeploy/applications/app31/deployment-groups/new?region=ap-south-1>.

--> provide group name

The screenshot shows the 'Create deployment group' page. The sidebar on the left shows the application 'app31' and compute type 'EC2/On-premises'. The main form starts with a 'Deployment group name' field containing 'deploygrp12'. Below it is a 'Service role' section with a note about granting CodeDeploy access to target instances. The browser address bar shows the URL: <https://ap-south-1.console.aws.amazon.com/codesuite/codedeploy/applications/app31/deployment-groups/new?region=ap-south-1>.

--> under service role--> select codedeploy-to-ec2-31 under EC2

--> under deployment settings check whether all at once selected or not

The screenshot shows the 'Create deployment group' wizard on the AWS CodeDeploy console. The current step is 'Service role'. A search bar at the top contains the ARN: `arn:aws:iam::637423422597:role/deploy-ec2-31`. Below it, the 'Deployment type' section is visible, with the 'In-place' option selected. The 'Environment configuration' section is partially visible at the bottom.

--> Under environment configuration select amazon ec2 instance--> select name under key and instance name under value

The screenshot shows the 'Create deployment group' wizard on the AWS CodeDeploy console, specifically the 'Environment configuration' step. It lists 'Amazon EC2 instances' as selected. Below this, there's a section for adding tags, showing a 'Name' key with a value of 'myubuntu'. At the bottom, there are buttons for 'Add tag' and '+ Add tag group'.

--> untick the unable load balancing check box under load balancer--> create deployment group-->

The screenshot shows the 'Create deployment group' wizard in the AWS CodeDeploy console. The 'Deployment Settings' step is selected. It includes a dropdown for 'Deployment configuration' set to 'CodeDeployDefault.AllAtOnce' and a 'Create deployment configuration' button. Below this, the 'Load balancer' section is shown, with a note about managing incoming traffic and a checkbox for 'Enable load balancing'. A 'Create deployment group' button is at the bottom right.

--> go in pipelines under code pipeline from left pane--> create pipeline

The screenshot shows the AWS CodePipeline console. The left sidebar is titled 'CodePipeline' and lists 'Source • CodeCommit', 'Artifacts • CodeArtifact', 'Build • CodeBuild', 'Deploy • CodeDeploy', and 'Pipeline • CodePipeline'. Under 'Pipeline', there is a 'Getting started' link and a 'Pipelines' link which is currently selected. The main area shows the 'Pipelines' page with a search bar and a table. The table has columns: Name, Latest execution status, Latest source revisions, Latest execution started, and Most recent executions. A message below the table says 'No results' and 'There are no results to display.'

--> provide name as pipeline23

Choose creation option

Step 2 of 6

Pipeline settings

Pipeline name
Enter the pipeline name. You cannot edit the pipeline name after it is created.
pipeline23
No more than 100 characters

Pipeline type
You can no longer create V1 pipelines through the console. We recommend you use the V2 pipeline type with improved release safety, pipeline triggers, parameterized pipelines, and a new billing model.

Execution mode
Choose the execution mode for your pipeline. This determines how the pipeline is run.
 Superseded
A more recent execution can overtake an older one. This is the default.
 Queued (Pipeline type V2 required)
Executions are processed one by one in the order that they are queued.
 Parallel (Pipeline type V2 required)

CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 15:52 31-12-2024

--> NEXT--> select source as AWS CodeCommit

--> select repo name as repo(our repo name)--> Branch name Main--> Next

Choose pipeline settings

Step 3 of 6

Source

Source provider
This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.
GitHub (via GitHub App)

Connection
Choose an existing connection that you have already configured, or create a new one and then return to this task.
arn:aws:codeconnections:ap-south-1:637423422597:connection/01 or **Connect to GitHub**

Repository name
Choose a repository in your GitHub account.
Pratiksha57/Python_Project

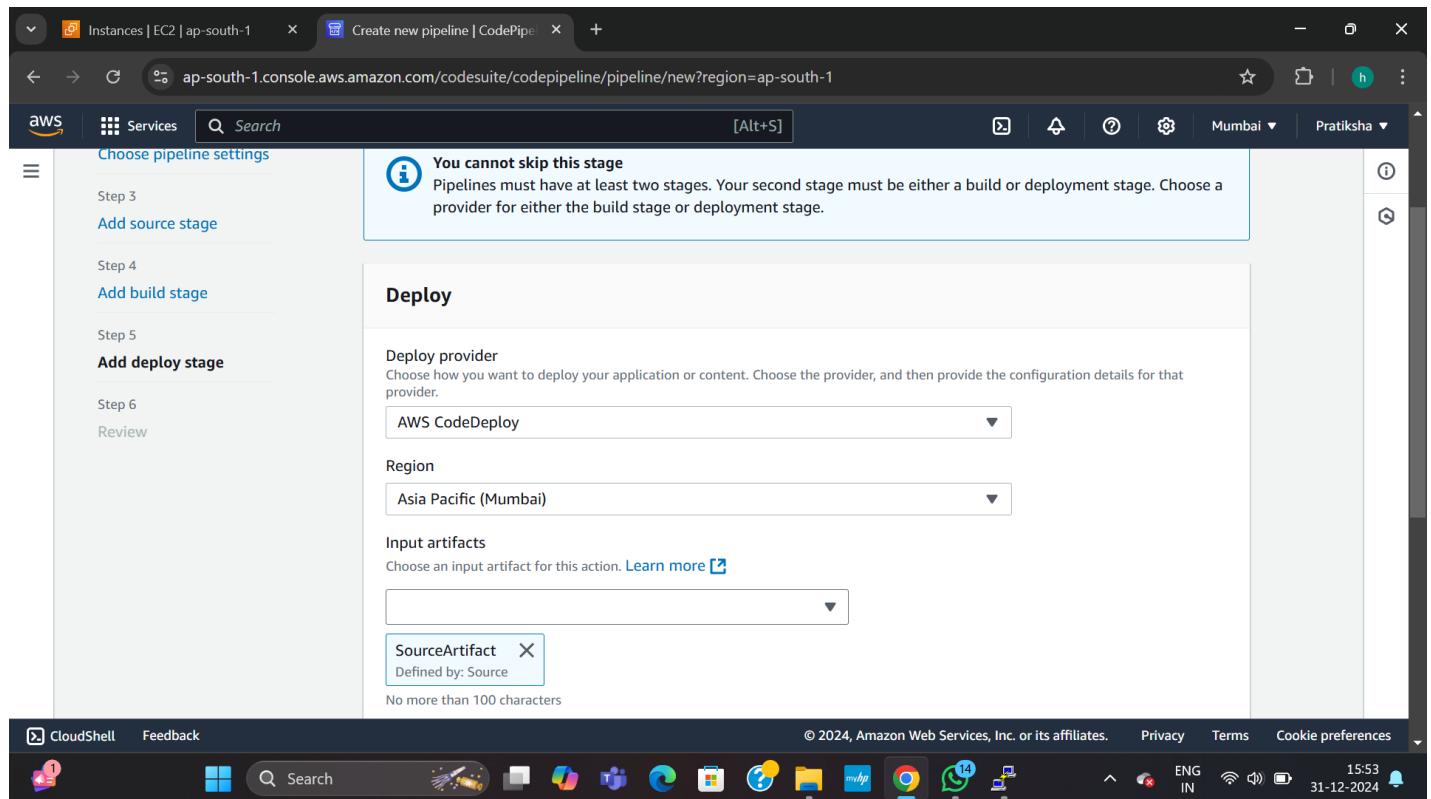
Default branch
Default branch will be used only when pipeline execution starts from a different source or manually started.
main

Output artifact format

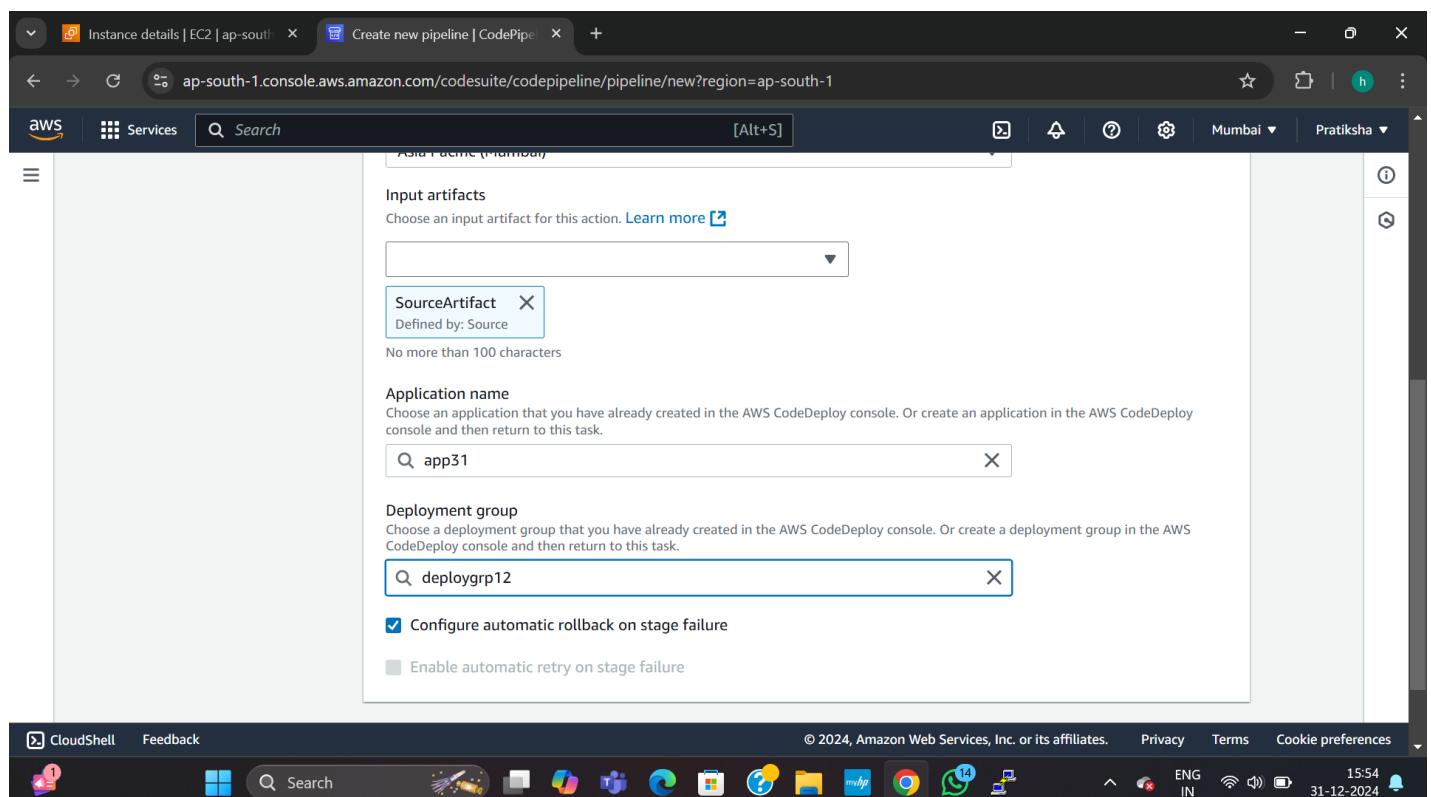
CloudShell Feedback © 2024, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences ENG IN 15:53 31-12-2024

--> Skip build stage

--> Select AWS code deploy



--> select application name and deployment group



--> NEXT-->create pipeline

The screenshot shows the AWS CodePipeline 'Create new pipeline' wizard at Step 5: Add deploy stage. The page title is 'Step 5: Add deploy stage'. A large text input field contains the configuration for a 'Deploy action provider' using AWS CodeDeploy. The configuration includes:

- ApplicationName: app31
- DeploymentGroupName: deploygrp12
- Configure automatic rollback on stage failure: Enabled
- Enable automatic retry on stage failure: Disabled

At the bottom right are 'Cancel', 'Previous', and 'Create pipeline' buttons. The browser's address bar shows the URL: ap-south-1.console.aws.amazon.com/codesuite/codepipeline/pipeline/new?region=ap-south-1.

We have completed setting up and creating CI-CD pipeline

The screenshot shows the AWS CodePipeline pipeline details page for 'pipeline23'. The pipeline type is V2 and the execution mode is QUEUED. The pipeline structure is as follows:

- Source**: GitHub (via GitHub App) - Succeeded (Auto retry attempt, View retry metadata). Pipeline execution ID: 72763156-ed83-44d7-90ac-e8c6833e6f4e.
- Build**: CodeBuild - Succeeded (2 minutes ago, View details). Build ID: dcc0da0b.
- Deploy**: CodeDeploy - Succeeded (2 minutes ago, View details). Deployment ID: dcc0da0b.

On the left, the navigation menu is expanded to show 'CodePipeline' and its sub-options: Source, Artifacts, Build, Deploy, Pipeline (selected), History, Settings, and Settings. The browser's address bar shows the URL: ap-south-1.console.aws.amazon.com/codesuite/codepipeline/pipelines/pipeline23/view?region=ap-south-1.

The screenshot shows the AWS CodePipeline console. On the left, the navigation pane is open with sections like 'Source', 'Artifacts', 'Build', 'Deploy', and 'Pipeline'. The 'Pipeline' section is expanded, showing 'Getting started', 'Pipelines', and 'Pipeline'. Under 'Pipeline', 'Pipeline' is selected, and its details are shown on the right. The pipeline has one stage named 'Deploy' which has succeeded. The pipeline execution ID is 22763156-ed83-44d7-90ac-e8c6833e6f4e. There are two green checkmarks on the right side of the pipeline stage.

--> Copy the public IP of instance and paste it in another tab

The screenshot shows the AWS EC2 Instances page. The left sidebar shows 'Instances' and 'Images' sections. The main area displays the 'Instance summary for i-03c9ae2c74ffb41ff (myubuntu)'. The summary includes the instance ID (i-03c9ae2c74ffb41ff), Public IPv4 address (15.207.248.22), Instance state (Running), Hostname type (IP name: ip-172-31-1-107.ap-south-1.compute.internal), Answer private resource DNS name (IPv4 (A)), Auto-assigned IP address, Private IP DNS name (IPv4 only) (ip-172-31-1-107.ap-south-1.compute.internal), Instance type (t2.micro), VPC ID, and Elastic IP addresses. A tooltip 'Public IPv4 address copied' is shown over the Public IPv4 address field. The bottom status bar shows the date and time as 31-12-2024 15:58.

You should see the Django application is running

