

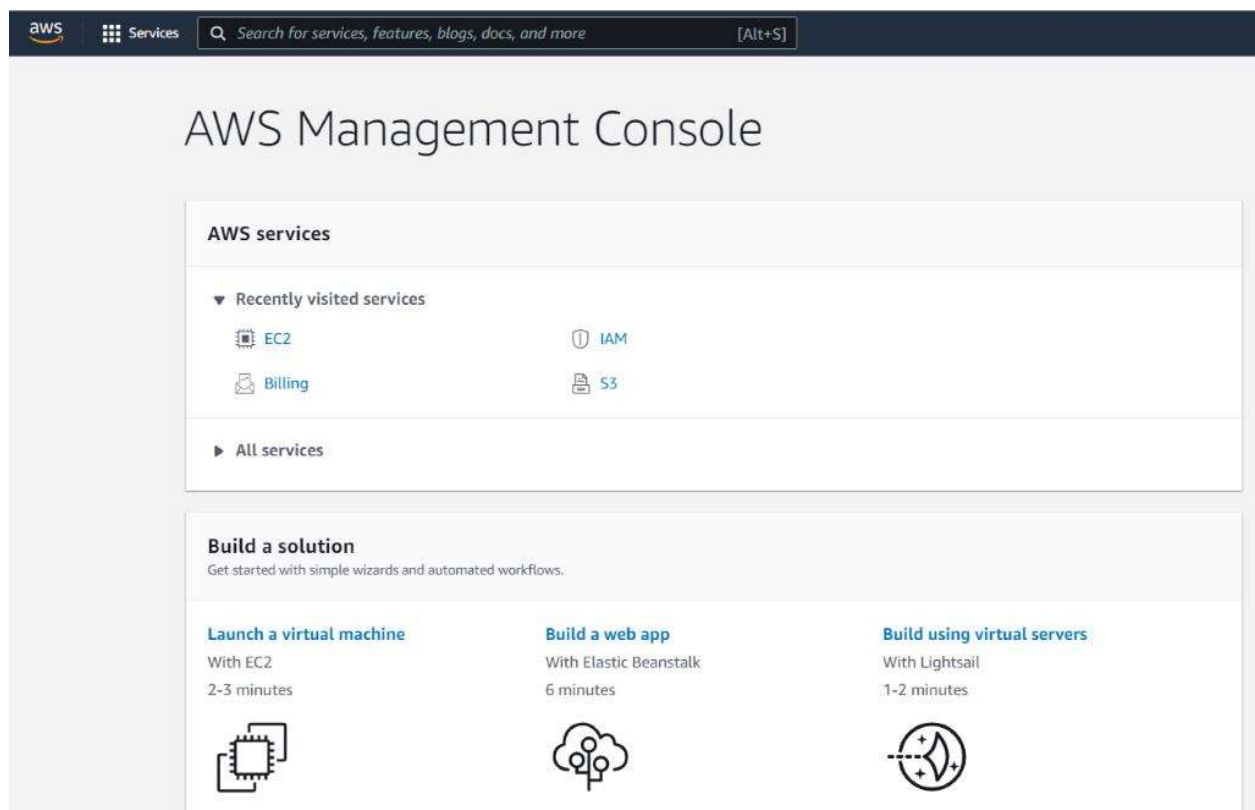
Deploy GitHub Project on Amazon EC2 Using GitHub Actions and AWS Code Deploy

Prerequisite

- Create a [GitHub](#) account
- Create an AWS Account

Create IAM Role for EC2 and Code Deploy

1. Open [AWS Management Console](#) and select IAM service



2. Select Roles from Access management

Identity and Access Management (IAM)

Search IAM

Dashboard

Access management

User groups

Users

Roles

Policies

Identity providers

Account settings

Access reports

Access analyzer

Archive rules

Analyzers

Settings

Credential report

Organization activity

Service control policies (SCPs)

Introducing the new IAM dashboard experience

We've redesigned the IAM dashboard experience to make it easier to use. [Let us know what you think](#)

IAM dashboard

Security recommendations 2

Add MFA for root user

Enable multi-factor authentication (MFA) for the root user to improve security for this account.

Add MFA

Deactivate or delete access keys for root user

Deactivate or delete the access keys for the root user. Instead, use access keys attached to an IAM user to improve security.

Manage access keys

IAM resources

User groups

Users

Roles

Policies

Identity providers

1

2

7

0

0

What's new

Updates for features in IAM

View all

- IAM Access Analyzer helps you generate fine-grained policies that specify the required actions for more than 50 services. 5 months ago
- IAM Access Analyzer helps you generate IAM policies based on access activity found in your organization trail. 5 months ago
- IAM Access Analyzer adds new policy checks to help validate conditions during IAM policy authoring. 7 months ago
- AWS Amplify announces support for IAM permissions boundaries on Amplify-generated IAM roles. 7 months ago

3. Select Create Role

AWS

Services

Search for services, features, blogs, docs, and more

[Alt+S]

Global

visual

Identity and Access Management (IAM)

Search IAM

Dashboard

Access management

User groups

Users

Roles

Policies

Identity providers

Account settings

Access reports

Access analyzer

Archive rules

Analyzers

Settings

Credential report

Organization activity

Service control policies (SCPs)

Introducing the new IAM roles experience

We've redesigned the IAM roles experience to make it easier to use. [Let us know what you think](#)

IAM > Roles

Roles (7) Info

An IAM role is an identity you can create that has specific permissions with credentials that are valid for short durations. Roles can be assumed by entities that you trust.

Search

< 1 >

Role name

Trusted entities

Last activity

☐ AWSServiceRoleForSupport

AWS Service: support (Service-Linked Role)

☐ AWSServiceRoleForTrustedAdvisor

AWS Service: trustedadvisor (Service-Linked Role)

☐ cdk-hnb659fts-ch-exec-role-537507758382-us-west-2

AWS Service: cloudformation

☐ cdk-hnb659fts-deploy-role-537507758382-us-west-2

Account: 537507758382

☐ cdk-hnb659fts-file-publishing-role-537507758382-us-west-2

Account: 537507758382

☐ cdk-hnb659fts-image-publishing-role-537507758382-us-west-2

Account: 537507758382

☐ cdk-hnb659fts-lookup-role-537507758382-us-west-2

Account: 537507758382

- Create a role for EC2 Instance. Select AWS Service as *trusted entity* and EC2 as *usecase*, click on *Next:Permissions*.

2

Create role 1 2 3 4

Select type of trusted entity

AWS service
EC2, Lambda and others

Another AWS account
Belonging to you or 3rd party

Web identity
Cognito or any OpenID provider

SAML 2.0 federation
Your corporate directory

Allows AWS services to perform actions on your behalf. [Learn more](#)

Choose a use case

Common use cases

EC2
Allows EC2 instances to call AWS services on your behalf

Lambda
Allows Lambda functions to call AWS services on your behalf

Or select a service to view its use cases

API Gateway	CloudWatch Events	EKS	IoT SiteWise	RDS
AWS Backup	CloudWatch Evidently	EMR	IoT Things Graph	Redshift
AWS Chatbot	CloudWatch RUM	EMR Containers	KMS	Rekognition
AWS Marketplace	CodeBuild	ElasticCache	Kinesis	RoboMaker
AWS Support	CodeDeploy	Elastic Beanstalk	Lake Formation	S3

* Required

[Cancel](#) [Next: Permissions](#)

- On the Permissions page, select **AmazonEC2RoleforAWSCodeDeploy** Policy and Click on **Next:Tags**

Create role 1 2 3 4

Attach permissions policies

Choose one or more policies to attach to your new role.

[Create policy](#)

[Filter policies](#) Showing 2 results

	Policy name	Used as
<input checked="" type="checkbox"/>	AmazonEC2RoleforAWSCodeDeploy	None
<input type="checkbox"/>	AmazonEC2RoleforAWSCodeDeployLimited	None

[Set permissions boundary](#)

* Required

[Cancel](#) [Previous](#) [Next: Tags](#)

- Ignore the tags and click **Next:Review**.

Create role

1 2 3 4

Add tags (optional)

IAM tags are key-value pairs you can add to your role. Tags can include user information, such as an email address, or can be descriptive, such as a job title. You can use the tags to organize, track, or control access for this role. [Learn more](#)

Key	Value (optional)	Remove
<input type="text" value="Add new key"/>	<input type="text"/>	

You can add 50 more tags.

Cancel Previous **Next: Review**

7. Provide the role name as `EC2_Role` on the review page and click create role.

Create role

1 2 3 4

Review

Provide the required information below and review this role before you create it.

Role name*
Use alphanumeric and "+-@_." characters. Maximum 64 characters.

Role description
Maximum 1000 characters. Use alphanumeric and "+-@_." characters.

Trusted entities AWS service: ec2.amazonaws.com

Policies AmazonEC2RoleforAWSCodeDeploy [?](#)

Permissions boundary Permissions boundary is not set

No tags were added.

* Required

Cancel Previous **Create role**

8. Open the `EC2_Role` and go to *Trust Relationships*.

New feature to generate a policy based on CloudTrail events.
AWS uses your CloudTrail events to identify the services and actions used and generate a least privileged policy that you can attach to this role.

Roles > EC2_Role

Summary

Role ARN	arn:aws:iam:537507758382:role/EC2_Role 🔗
Role description	Allows EC2 instances to call AWS services on your behalf. Edit
Instance Profile ARNs	arn:aws:iam:537507758382:instance-profile/EC2_Role 🔗
Path	/
Creation time	2022-01-10 10:51 UTC+0530
Last activity	Not accessed in the tracking period
Maximum session duration	1 hour Edit

Permissions

Trust relationships

Tags

Access Advisor

Revoke sessions

You can view the trusted entities that can assume the role and the access conditions for the role. [Show policy document](#)

Edit trust relationship

Trusted entities

The following trusted entities can assume this role.

Trusted entities

The identity provider(s) ec2.amazonaws.com

Conditions

The following conditions define how and when trusted entities can assume the role.

There are no conditions associated with this role.

9. Click on *Edit Trust Relationship* and paste below policy.

```

{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ec2.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}

```

Edit Trust Relationship

You can customize trust relationships by editing the following access control policy document.

Policy Document

```
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Principal": {
7         "Service": "ec2.amazonaws.com"
8       },
9       "Action": "sts:AssumeRole"
10    }
11  ]
12 }
```


Cancel Update Trust Policy

10. Now we will create a role for CodeDeploy. Open [AWS Management Console](#) and select IAM service. Select Roles from Access management and click on Create Role. Select AWS Service as trusted entity and EC2 as usecase, click on Next:Permissions.


Create role


1 2 3 4

Select type of trusted entity

**AWS service**
EC2, Lambda and others

**Another AWS account**
Belonging to you or 3rd party

**Web identity**
Cognito or any OpenID provider

**SAML 2.0 federation**
Your corporate directory

Allows AWS services to perform actions on your behalf. [Learn more](#)

Choose a use case

Common use cases

EC2

Allows EC2 instances to call AWS services on your behalf.

Lambda

Allows Lambda functions to call AWS services on your behalf.


Or select a service to view its use cases


API Gateway	CloudWatch Events	EKS	IoT SiteWise	RDS
AWS Backup	CloudWatch Evidently	EMR	IoT Things Graph	Redshift
AWS Chatbot	CloudWatch RUM	EMR Containers	KMS	Rekognition
AWS Marketplace	CodeBuild	ElastiCache	Kinesis	RoboMaker
AWS Support	CodeDeploy	Elastic Beanstalk	Lake Formation	S3


* Required

Cancel Next: Permissions

11. On the Permissions page, select the below policy and Click on *Next:Tags*.
AmazonEC2FullAccess, AWSCodeDeployFullAccess, AdministratorAccess,
AWSCodeDeployRole

Filter policies ▼ <input type="text" value="AWSCodeDeployFullAccess"/>		Showing 1 result
<input type="checkbox"/>	Policy name ▼	Used as
<input checked="" type="checkbox"/>	 AWSCodeDeployFullAccess	None

Filter policies ▼ <input type="text" value="AdministratorAccess"/>		Showing 4 results
<input type="checkbox"/>	Policy name ▼	Used as
<input checked="" type="checkbox"/>	 AdministratorAccess	Permissions policy (2)

Filter policies ▼ <input type="text" value="AWSCodeDeployRole"/>		Showing 6 results
<input type="checkbox"/>	Policy name ▼	Used as
<input checked="" type="checkbox"/>	 AWSCodeDeployRole	None

Filter policies ▼ <input type="text" value="AmazonEC2FullAccess"/>		Showing 1 result
<input type="checkbox"/>	Policy name ▼	Used as
<input checked="" type="checkbox"/>	 AmazonEC2FullAccess	None

12. Tags can be ignored, click on *Next:Review*.
13. Provide the role name as CodeDeploy_Role on the review page.

Create role

1 2 3 4

Review

Provide the required information below and review this role before you create it.

Role name*

Use alphanumeric and '+', '@', '-' characters. Maximum 64 characters.

Role description

Maximum 1000 characters. Use alphanumeric and '+', '@', '-' characters.

Trusted entities AWS service: ec2.amazonaws.com

Policies

-  AmazonEC2FullAccess [↗](#)
-  AWSCodeDeployFullAccess [↗](#)
-  AdministratorAccess [↗](#)
-  AWSCodeDeployRole [↗](#)

Permissions boundary Permissions boundary is not set

No tags were added.

* Required

[Cancel](#)

[Previous](#)

[Create role](#)

14. Once CodeDeploy Role is created, Open the CodeDeploy_Role.

New feature to generate a policy based on CloudTrail events.
AWS uses your CloudTrail events to identify the services and actions used and generate a least privileged policy that you can attach to this role.

Roles > CodeDeploy_Role

Summary [Delete role](#)

Role ARN	arn:aws:iam::537507758382:role/CodeDeploy_Role ↗
Role description	Allows EC2 instances to call AWS services on your behalf. Edit
Instance Profile ARNs	arn:aws:iam::537507758382:instance-profile/CodeDeploy_Role ↗
Path	/
Creation time	2022-01-10 11:08 UTC+0530
Last activity	Not accessed in the tracking period
Maximum session duration	1 hour Edit

Permissions

Trust relationships

Tags

Access Advisor

Revoke sessions

You can view the trusted entities that can assume the role and the access conditions for the role. [Show policy document](#)

[Edit trust relationship](#)

Trusted entities

The following trusted entities can assume this role.

Trusted entities

The identity provider(s) ec2.amazonaws.com

Conditions

The following conditions define how and when trusted entities can assume the role.

There are no conditions associated with this role.

15. Go to *Trust Relationships* then *Edit Trust Relationship* and use below policy

{

"Version": "2012-10-17",

"Statement": [

8


```

{
  "Effect": "Allow",
  "Principal": {
    "Service": "codedeploy.amazonaws.com"
  },
  "Action": "sts:AssumeRole"
}
]
}

```

Edit Trust Relationship

You can customize trust relationships by editing the following access control policy document.

Policy Document

```

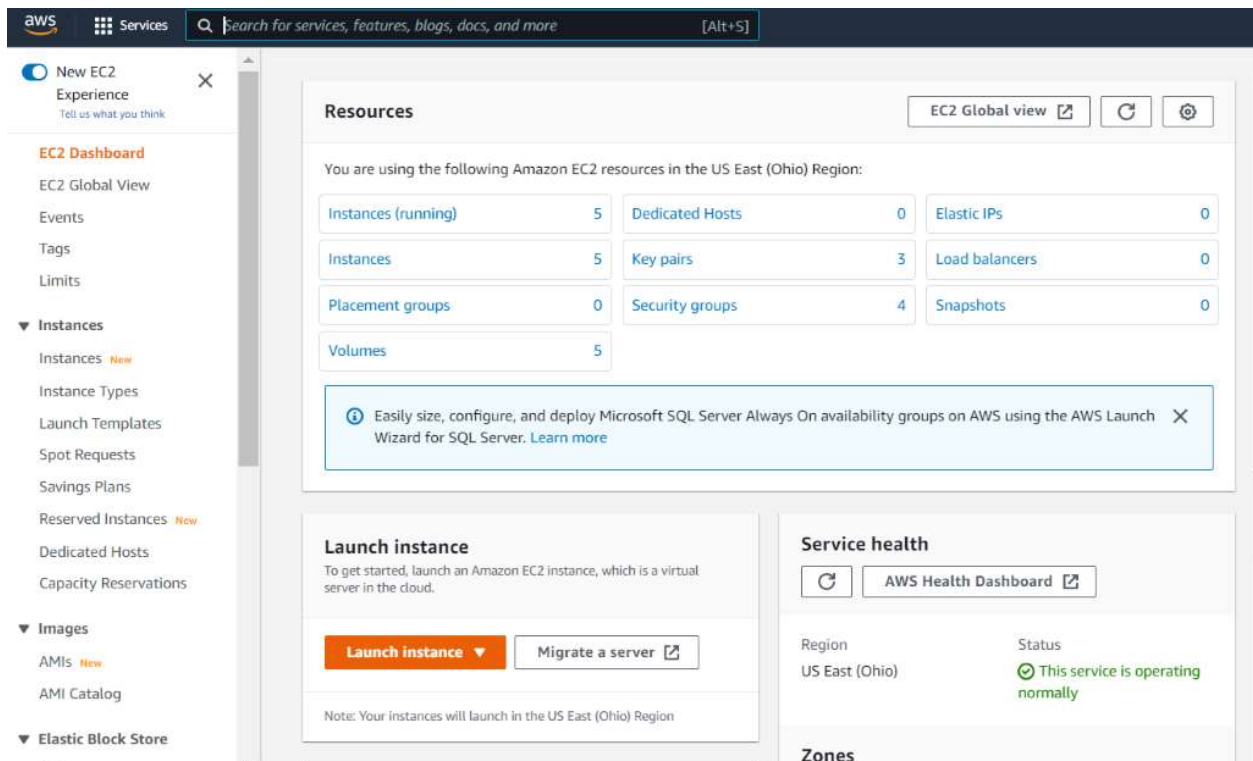
1 {
2   "Version": "2012-10-17",
3   "Statement": [
4     {
5       "Effect": "Allow",
6       "Principal": {
7         "Service": "codedeploy.amazonaws.com"
8       },
9       "Action": "sts:AssumeRole"
10    }
11  ]
12 }
13

```

Cancel [Update Trust Policy](#)

Create EC2 Instance

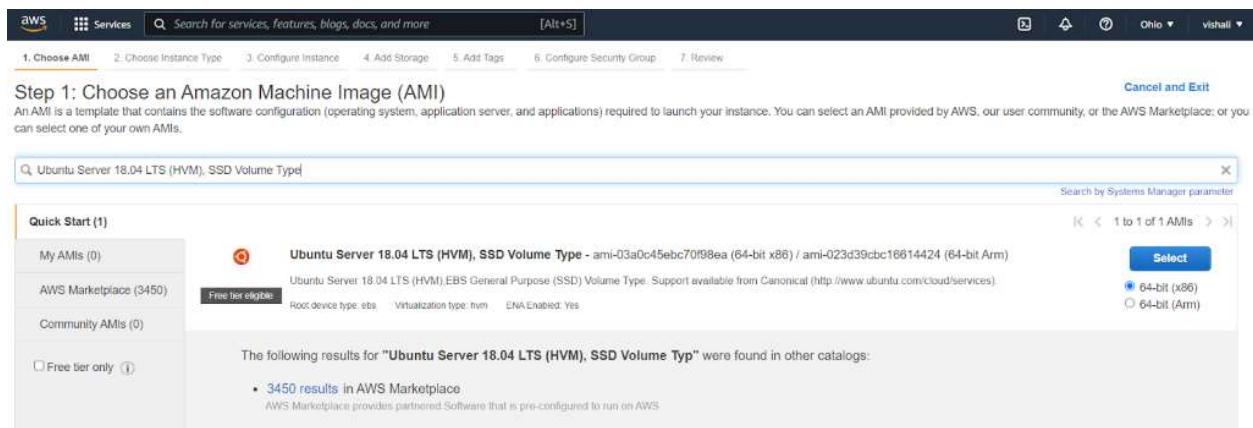
16. To create an EC2 instance, Open [AWS Management Console](#) and select EC2 service. Go to EC2 Dashboard and click on Launch Instance.



17. Choose an Amazon Machine Image(AMI)

In the free tier option,

Ubuntu Server 18.04 LTS (HVM), SSD Volume Type — ami-0a313d6098716f372 (64-bit x86)



18. Select t2.micro in *Choose Instance Type* page and proceed to *Configure Instance* page.

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

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 (1 vCPUs, 2.5 GHz, 1 GiB memory, EBS only)

	Family	Type	vCPUs (1)	Memory (GiB)	Instance Storage (GB) (1)	EBS-Optimized Available (1)	Network Performance (1)	IPv6 Support (1)
<input type="checkbox"/>	t2	t2.nano	1	0.5	EBS only	-	Low to Moderate	Yes
<input checked="" type="checkbox"/>	t2	t2.micro	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

Cancel Previous **Review and Launch** Next: Configure Instance Details

19. To establish the connection between EC2 instance and codeDeploy, Select EC2_Role, which we created before.

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 3: Configure Instance Details

Network (1) vpc-01096deaf13538153 (default) Create new VPC

Subnet (1) No preference (default subnet in any Availability Zone) Create new subnet

Auto-assign Public IP (1) Use subnet setting (Enable) -

Hostname type (1) Use subnet setting (IP name) -

DNS Hostname (1)
 ☐ Enable IP name (A record) DNS requests
☒ Enable resource-based IPv4 (A record) DNS requests
☐ Enable resource-based IPv6 (AAAA record) DNS requests

Placement group (1) ☐ Add instance to placement group

Capacity Reservation (1) Open -

Domain join directory (1) No directory Create new directory

IAM role (1) None Create new IAM role

Shutdown behavior (1) None

Stop - Hibernate behavior (1) CodeDeploy_Role

Enable termination protection (1) EC2_Role

Monitoring (1) ☐ Enable CloudWatch detailed monitoring

Cancel Previous **Review and Launch** Next: Add Storage

20. On the *Tag* page, add a tag called development. The tag will require creating a *codeDeploy* service.

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 5: Add Tags

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver.

A copy of a tag can be applied to volumes, instances or both.

Tags will be applied to all instances and volumes. [Learn more](#) about tagging your Amazon EC2 resources.

Key (128 characters maximum)	Value (256 characters maximum)	Instances ①	Volumes ①	Network Interfaces ①
development		<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

[Add another tag](#) (Up to 50 tags maximum)

[Cancel](#) [Previous](#) [Review and Launch](#) [Next: Configure Security Group](#)

21. In the *Configure Security Group* page *Add Rule* called *All traffic*, select *source* called *anywhere*. This rule will enable you to connect the Instance from anywhere. NOTE - This is not advisable in the Production environment. Add the following rules:

Type	Port Range	Description.
SSH	22	Port for SSH'ing into your server
HTTP	80	Port for HTTP requests to your web server
HTTPS	443	Port for HTTPS requests to your web server
Custom TCP	8000	Port which Django will run
Custom TCP	5432	Port at which to connect to PostgreSQL

Details
Security
Networking
Storage
Status checks
Monitoring
Tags

▼ Security details

IAM Role
-

Owner ID
537507758382

Launch time
Wed Jan 05 2022 13:51:20 GMT+0530 (India Standard Time)

Security groups
sg-0d5a6d29e1175af6a (launch-wizard-3)

▼ Inbound rules

Filter rules

Security group rule ID	Port range	Protocol	Source	Security groups
sg-07d5c202bb479ccb4	22	TCP	0.0.0.0/0	launch-wizard-3
sg-0d22b1799f1e24dc3	8000	TCP	::/0	launch-wizard-3
sg-0b967b78937e1d99e	22	TCP	::/0	launch-wizard-3
sg-0c8f928b13f9dd694	8000	TCP	0.0.0.0/0	launch-wizard-3

▼ Outbound rules

22. Select the review page, then Launch the Instance. Wait for a few minutes to start the EC2 Instance.

1. Choose AMI 2. Choose Instance Type 3. Configure Instance 4. Add Storage 5. Add Tags 6. Configure Security Group 7. Review

Step 7: Review Instance Launch

Please review your instance launch details. You can go back to edit changes for each section. Click **Launch** to assign a key pair to your instance and complete the launch process.

⚠ Improve your instances' security. Your security group, launch-wizard-4, is open to the world.

Your instances may be accessible from any IP address. We recommend that you update your security group rules to allow access from known IP addresses only. You can also open additional ports in your security group to facilitate access to the application or service you're running, e.g., HTTP (80) for web servers. [Edit security groups](#)

▼ AMI Details [Edit AMI](#)

Free tier eligible **Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type - ami-002068ed284fb165b**

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 2.25, Butils 2.26.1, and the latest software packages through extras. This AMI is the successor of the Amazon Linux AMI that is n...

Root Device Type: xfs Virtualization type: hvm

▼ Instance Type [Edit instance type](#)

Instance Type	ECUs	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
t2.micro	-	1	1	EBS only	-	Low to Moderate

▼ Security Groups [Edit security groups](#)

Security group name	
launch-wizard-4	
Description	launch-wizard-4 created 2022-01-10T11:25:50.002+05:30

[Cancel](#) [Previous](#) [Launch](#)

23. If you want to access the Instance (ssh) from your local system, create a new Key Pair and download the key.

Select an existing key pair or create a new key pair



A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance. Amazon EC2 supports ED25519 and RSA key pair types.

Note: The selected key pair will be added to the set of keys authorized for this instance. Learn more about [removing existing key pairs from a public AMI](#).

Create a new key pair 

Key pair type

☒ RSA ☐ ED25519

Key pair name

new_key_pair

Download Key Pair



You have to download the **private key file** (*.pem file) before you can continue. **Store it in a secure and accessible location.** You will not be able to download the file again after it's created.

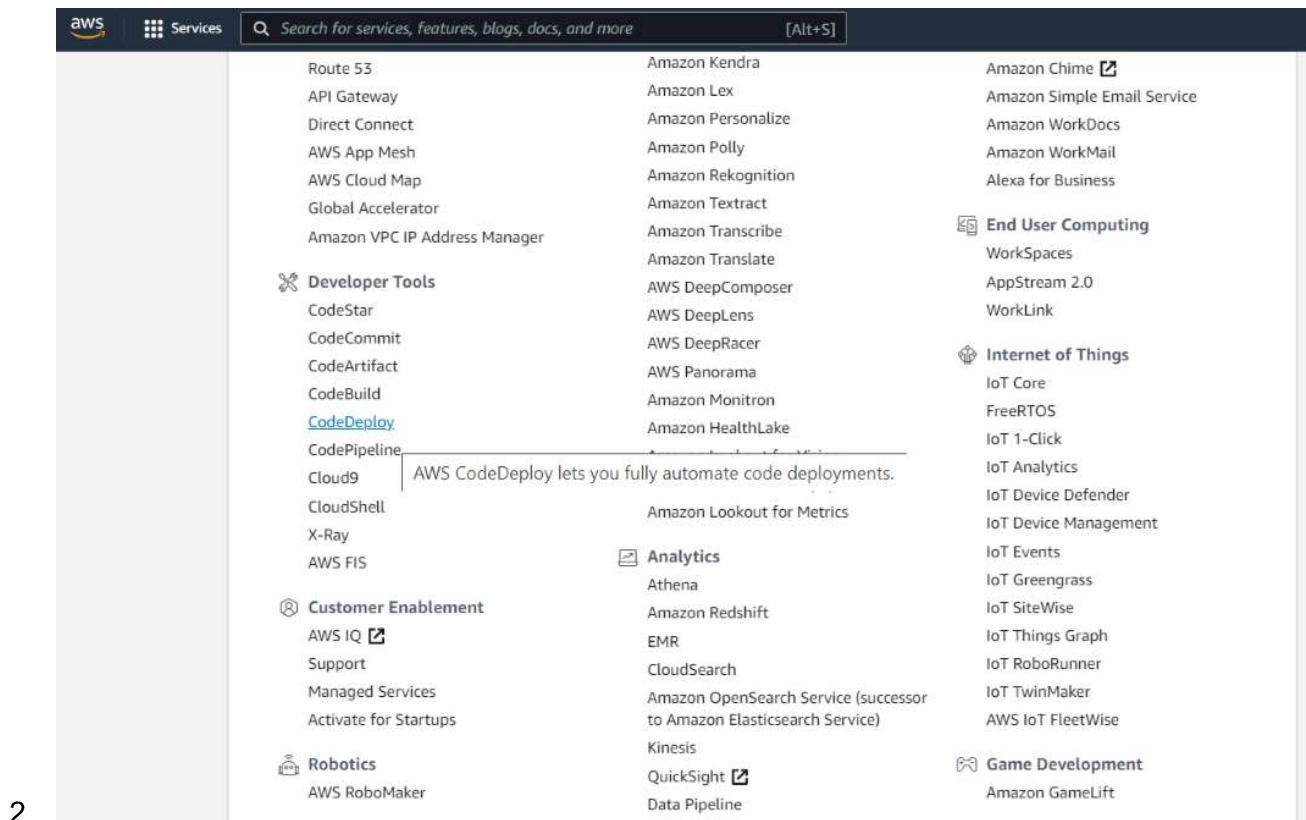
Cancel

Launch Instances

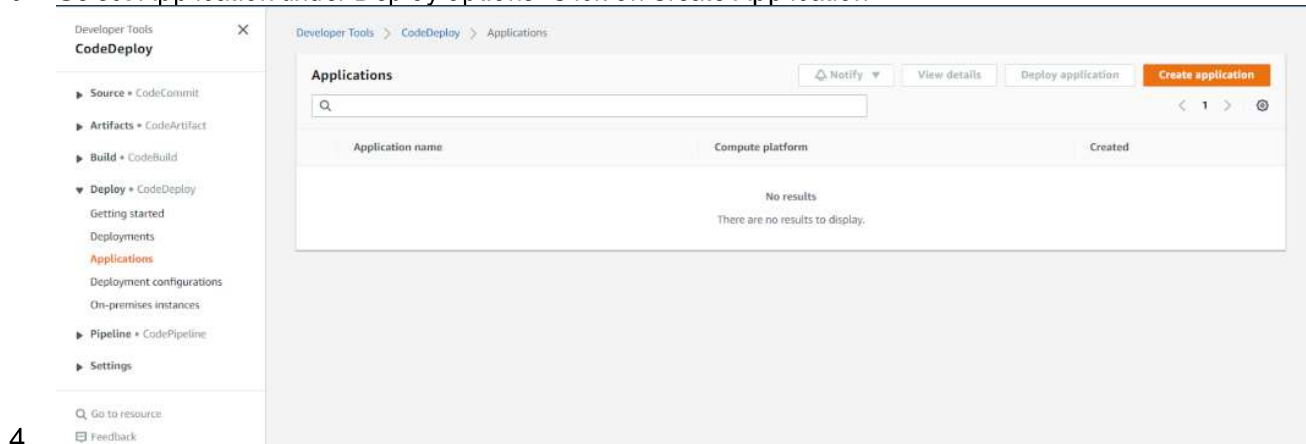
CodeDeploy Service Configuration

AWS [CodeDeploy Service](#) will automate the GitHub application deployment to EC2.

1. Open AWS Management Console and select CodeDeploy under Developer tools.



3. Select Application under Deploy options. Click on Create Application.



5. Create an Application name called Django_Application with *compute platform* EC2/On-premises. [GitHub Action](#) will use the application name.

Developer Tools > CodeDeploy > Applications > Create application

Create application

Application configuration

Application name

Enter an application name

100 character limit

Compute platform

Choose a compute platform

Cancel **Create application**

6.

7. Once Application Created, Create a *Deployment Group* and name development_gropup. Get the *Role ARN* from [CodeDeploy_Role](#), which we created before and put in the service role. [GitHub Action](#) will use the deployment Group name.

Developer Tools > CodeDeploy > Applications > Django_Application

Django_Application

Notify Delete application

Application details

Name	Compute platform
Django_Application	EC2/On-premises

Deployments **Deployment groups** Revisions

Deployment groups

View details Edit **Create deployment group**

8.

Application

Application
Git_Application
Compute type
EC2/On-premises

Deployment group name

Enter a deployment group name

development_gropup

100 character limit

Service role

Enter a service role
Enter a service role with CodeDeploy permissions that grants AWS CodeDeploy access to your target instances.

Q |

CodeDeploy_Role
arn:aws:iam::537507758382:role/CodeDeploy_Role

Deployment type

CodeDeploy Role

9.

- Choose In-place Deployment type. Select Amazon Ec2 Instances environment configuration and Tag key development to create AWS EC2 instance.

Deployment type

Choose how to deploy your application

☒ In-place

Updates the instances in the deployment group with the latest application revisions. During a deployment, each instance will be briefly taken offline for its update.

☐ Blue/green

Replaces the instances in the deployment group with new instances and deploys the latest application revision to them. After instances in the replacement environment are registered with a load balancer, instances from the original environment are deregistered and can be terminated.

Environment configuration

Select any combination of Amazon EC2 Auto Scaling groups, Amazon EC2 instances, and on-premises instances to add to this deployment

☐ Amazon EC2 Auto Scaling groups

☒ Amazon EC2 instances

1 unique matched instance. [Click here for details](#)

You can add up to three groups of tags for EC2 instances to this deployment group.

One tag group: Any instance identified by the tag group will be deployed to.

Multiple tag groups: Only instances identified by all the tag groups will be deployed to.

Tag group 1

Key

Value - optional

Remove tag

11.

12. Select a schedule manager to install the CodeDeploy agent. Set *OneAtATime* deployment setting and Create Deployment Group without a load balancer.

Agent configuration with AWS Systems Manager [Info](#)

AWS Systems Manager will install the CodeDeploy Agent on all instances and update it based on the configured frequency.

Complete the required prerequisites before AWS Systems Manager can install the CodeDeploy Agent.
 Make sure the AWS Systems Manager Agent is installed on all instances and attach the required IAM policies to them. [Learn more](#)

Install AWS CodeDeploy Agent

☐ Never
☐ Only once
☒ Now and schedule updates

Days

Deployment settings

Deployment configuration

Choose from a list of default and custom deployment configurations. A deployment configuration is a set of rules that determines how fast an application is deployed and the success or failure conditions for a deployment.

or

Load balancer

Select a load balancer to manage incoming traffic during the deployment process. The load balancer blocks traffic from each instance while it's being deployed to and allows traffic to it again after the deployment succeeds.

☐ Enable load balancing

► Advanced - optional

13.

14. Once the Deployment Group is created, test the deployment by creating a Deployment with any name. Click on Create deployment.

✓ Success
✕

Deployment group created

Developer Tools > CodeDeploy > Applications > Git_Application > development_gropup

development_gropup

Deployment group details

Deployment group name	Application name	Compute platform
development_gropup	Git_Application	EC2/On-premises
Deployment type	Service role ARN	Deployment configuration
In-place	arn:aws:iam::537507758382:role/CodeDeployRole	CodeDeployDefault.OneAtATime
Rollback enabled	Agent update scheduler	

15.

16. You will see the create deployment settings.

Create deployment

Deployment settings

Application

Git_Application

Deployment group

development_gropup

Compute platform

EC2/On-premises

Deployment type

In-place

17.
18.

19. Go to the [GitHub settings](#) page. Under Developer settings, click on Generate new token

[Settings](#) / Developer settings

GitHub Apps

OAuth Apps

Personal access tokens

Personal access tokens

Generate new token

Revoke all

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

Personal access token for simple-python-app repo — *admin:enterprise, admin:gpg_key, admin:org, Never used, admin:org_hook, admin:public_key, admin:repo_hook, delete:packages, delete_repo, gist, notifications, repo, user, workflow, write:discussion, write:packages*

Delete

[Regenerate](#) this token to take advantage of the [new token formats](#)

[⚠ This token has no expiration date.](#)

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

20.

New personal access token

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

Note

Used for deployment purpose in EC2 instance

What's this token for?

Expiration *

30 days

The token will expire on Wed, Feb 9 2022

Select scopes

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

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> repo:invite	Access repository invitations
<input checked="" type="checkbox"/> security_events	Read and write security events
<input checked="" type="checkbox"/> workflow	Update GitHub Action workflows
<input checked="" type="checkbox"/> write:packages	Upload packages to GitHub Package Registry
<input checked="" type="checkbox"/> read:packages	Download packages from GitHub Package Registry
<input checked="" type="checkbox"/> delete:packages	Delete packages from GitHub Package Registry
<input checked="" type="checkbox"/> admin:org	Full control of orgs and teams, read and write org projects

21.

<input checked="" type="checkbox"/> notifications	Access notifications
<input checked="" type="checkbox"/> user	Update ALL user data
<input checked="" type="checkbox"/> read:user	Read ALL user profile data
<input checked="" type="checkbox"/> user:email	Access user email addresses (read-only)
<input checked="" type="checkbox"/> user:follow	Follow and unfollow users
<input checked="" type="checkbox"/> delete_repo	Delete repositories
<input checked="" type="checkbox"/> write:discussion	Read and write team discussions
<input checked="" type="checkbox"/> read:discussion	Read team discussions
<input checked="" type="checkbox"/> admin:enterprise	Full control of enterprises
<input checked="" type="checkbox"/> manage_runners:enterprise	Manage enterprise runners and runner-groups
<input checked="" type="checkbox"/> manage_billing:enterprise	Read and write enterprise billing data
<input checked="" type="checkbox"/> read:enterprise	Read enterprise profile data
<input checked="" type="checkbox"/> admin:pgp_key	Full control of public user GPG keys (Developer Preview)
<input checked="" type="checkbox"/> write:pgp_key	Write public user GPG keys
<input checked="" type="checkbox"/> read:pgp_key	Read public user GPG keys

Generate token

Cancel

22.

Personal access tokens

Generate new token

Revoke all

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


Make sure to copy your personal access token now. You won't be able to see it again!

✓ ghp_IKx9ze8PCKLQ0NLQvTJ40P9XpWqRfL26LrF4 

Delete

Personal access token for simple-python-app repo — *admin:enterprise, admin:pgp_key, admin:org, Never used*
admin:org_hook, admin:public_key, admin:repo_hook, delete:packages, delete_repo, gist, notifications, repo, user,
workflow, write:discussion, write:packages

Delete

 [Regenerate](#) this token to take advantage of the [new token formats](#)

 This token has no expiration date.

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

23.

24. **GitHub Token:** ghp_IKx9ze8PCKLQ0NLQvTJ40P9XpWqRfL26LrF4

25. Select Revision Type My application is stored in GitHub, and select Connect to GitHub by providing the [GitHub token](#).

Revision type

☐ My application is stored in Amazon S3

☒ My application is stored in GitHub

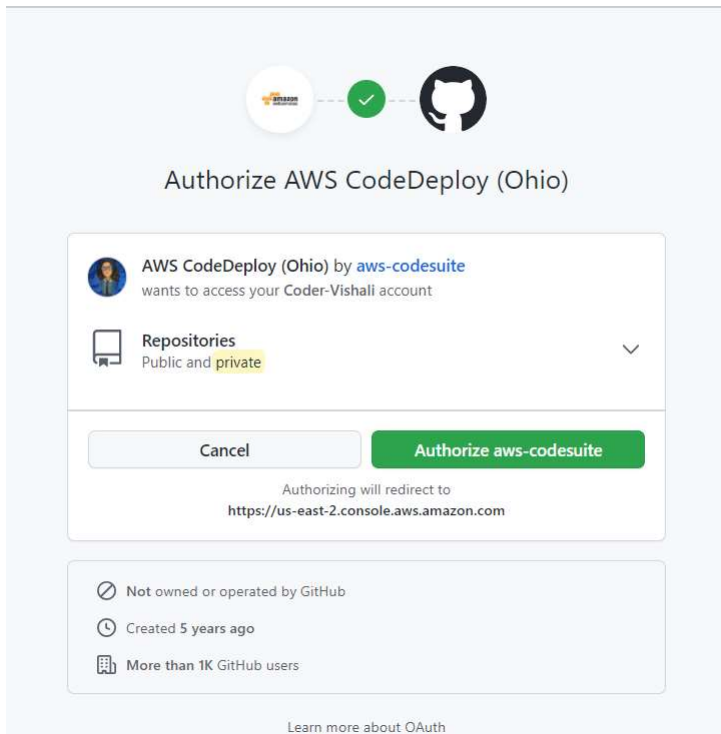
GitHub token name

Select the name of the token associated to an account you have already connected, or grant AWS CodeDeploy permission to access a different account. To connect to a GitHub account for the first time, type an alias for the account, and then choose Connect to GitHub

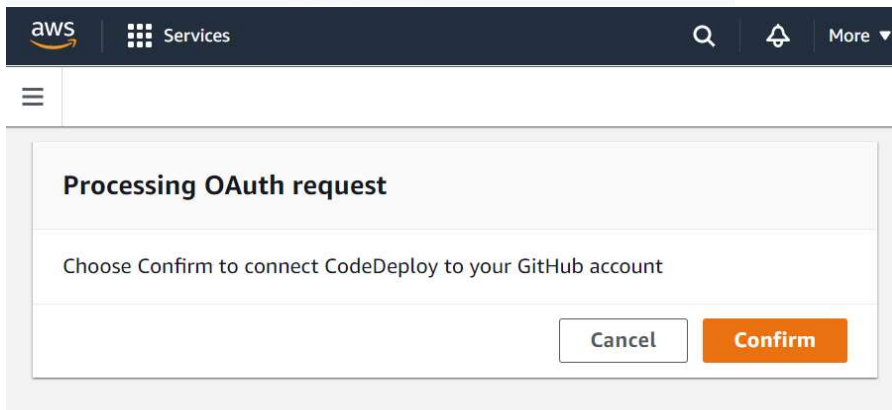
ghp_IKx9ze8PCKLQ0NLQvTJ40P9XpWqRfL26LrF4

Connecting

26.



27.



28.

GitHub token name

Select the name of the token associated to an account you have already connected, or grant AWS CodeDeploy permission to access a different account. To connect to a GitHub account for the first time, type an alias for the account, and then choose Connect to GitHub

Q ghp_IKx9ze8PCKLQ0NLQvTJ40P9XpWqRfL26LrF4



Connected



Application Git_Application successfully bound to ghp_IKx9ze8PCKLQ0NLQvTJ40P9XpWqRfL26LrF4 GitHub token



29.

30. Once connected to GitHub, Provide the [repository name](#) and last *Commit ID*. Select *Overwrite the content* and Create Deployment.

Repository name

Coder-Vishali/spring-boot-mongo

Commit ID

990cb615755ec9e2bca9f9ec3bcd881be4bc3503

31.

Additional deployment behavior settings

ApplicationStop lifecycle event failure - *optional*

Type a deployment group name

☐ Don't fail the deployment to an instance if this lifecycle event on the instance fails

Content options - *optional*

Choose what to do during a deployment when a file on a target instance has the same name as a file in the application revision

☐ Fail the deployment
An error is reported and the deployment status is changed to Failed.

☒ Overwrite the content
The file in the application revision is copied to the target location on the instance, replacing the previous file.

☐ Retain the content
The file in the application revision is not copied to the instance. The existing file is kept at the target location and treated as part of the new deployment.

► Deployment group overrides

► Rollback configuration overrides

Cancel

Create deployment

32.

33. Successfully created deployment

GitHub Project

To push the files into GitHub:

git init

Open the Git bash:

git add .

git commit -m "first commit"

git branch -M main

git remote add origin https://github.com/Coder-Vishali/django_app.git

git push -u origin main

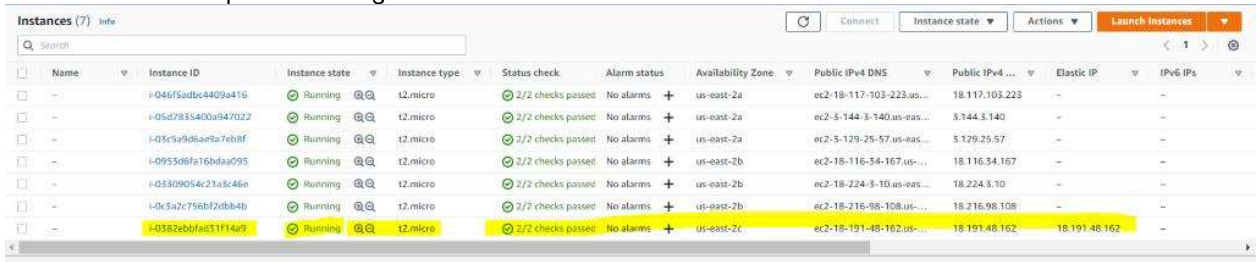
To push local files into ec2 instance:

pscp -i <.pkk file> <filename> <ec2 user>@<ip address>:/home/ec2-user

pscp -i C:\Users\VISHALI\Downloads\new_key_pair.ppk C:\Users\VISHALI\Django_app ec2-user@18.224.3.10:/home/ec2-user

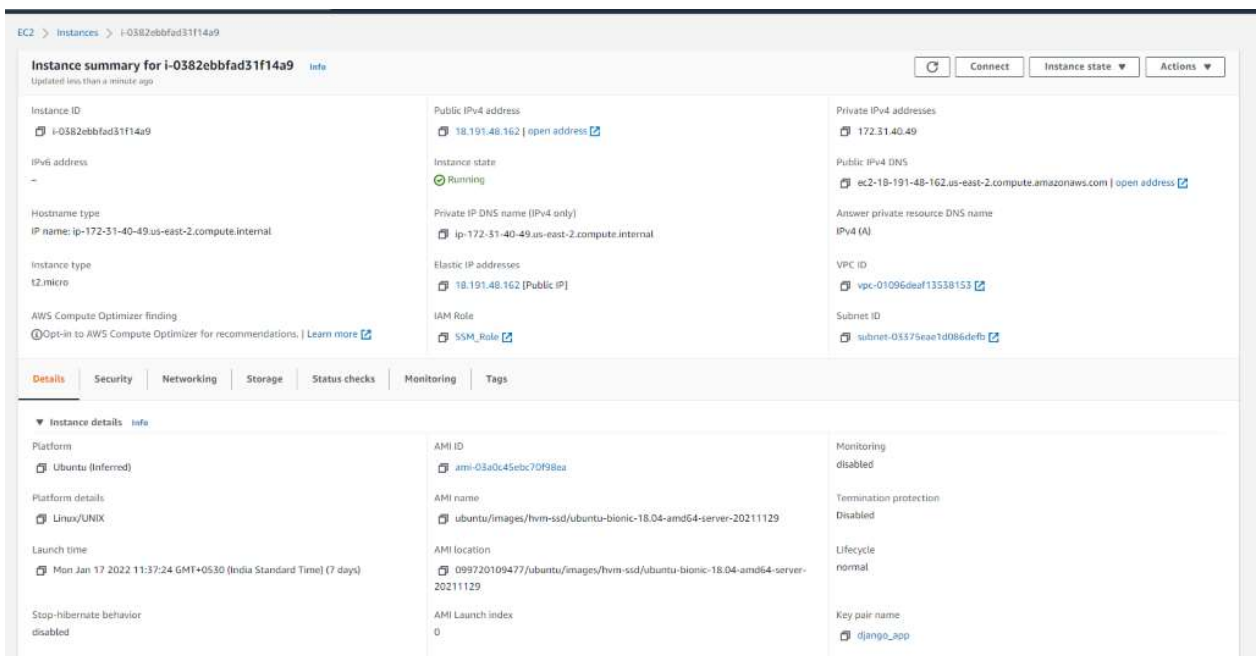
Launch EC2 Instance

1. Once Instance is up and running.



Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP	IPv6 IPs
-	i-046f5adbc4409a416	Running	t2.micro	2/2 checks passed	No alarms	us-east-2a	ec2-18-117-103-223.us...	18.117.103.223	-	-
-	i-05d7835400a947022	Running	t2.micro	2/2 checks passed	No alarms	us-east-2a	ec2-3-144-3-140.us-eas...	3.144.3.140	-	-
-	i-04c5a9d6ae9a7eb8f	Running	t2.micro	2/2 checks passed	No alarms	us-east-2a	ec2-3-129-25-57.us-eas...	3.129.25.57	-	-
-	i-0953d6fa16bdaa095	Running	t2.micro	2/2 checks passed	No alarms	us-east-2b	ec2-18-116-34-167.us-...	18.116.34.167	-	-
-	i-03309054c21a8c46e	Running	t2.micro	2/2 checks passed	No alarms	us-east-2b	ec2-18-224-3-10.us-eas...	18.224.3.10	-	-
-	i-0c5a2c756bf7d8bb4b	Running	t2.micro	2/2 checks passed	No alarms	us-east-2b	ec2-18-216-98-108.us-...	18.216.98.108	-	-
-	i-0382ebbfad31f14a9	Running	t2.micro	2/2 checks passed	No alarms	us-east-2c	ec2-18-191-48-162.us-...	18.191.48.162	18.191.48.162	-

- 2.



Instance summary for i-0382ebbfad31f14a9		
Instance ID i-0382ebbfad31f14a9	Public IPv4 address 18.191.48.162 open address	Private IPv4 addresses 172.31.40.49
IPv6 address -	Instance state Running	Public IPv4 DNS ec2-18-191-48-162.us-east-2.compute.amazonaws.com open address
Hostname type IP name: ip-172-31-40-49.us-east-2.compute.internal	Private IP DNS name (IPv4 only) ip-172-31-40-49.us-east-2.compute.internal	Answer private resource DNS name IPv4 (A)
Instance type t2.micro	Elastic IP addresses 18.191.48.162 [Public IP]	VPC ID vpc-01096deaf13538153
AWS Compute Optimizer finding Opt-in to AWS Compute Optimizer for recommendations. Learn more	IAM Role SSM_Role	Subnet ID subnet-03375eae1d086defb
Details Security Networking Storage Status checks Monitoring Tags		
Instance details		
Platform Ubuntu (Inferred)	AMI ID ami-03a0c45ebc70f98ea	Monitoring disabled
Platform details Linux/UNIX	AMI name ubuntu/images/hvm-ssd/ubuntu-bionic-18.04-amd64-server-20211129	Termination protection Disabled
Launch time Mon Jan 17 2022 11:37:24 GMT+0530 (India Standard Time) (7 days)	AMI location 099720109477/ubuntu/images/hvm-ssd/ubuntu-bionic-18.04-amd64-server-20211129	Lifecycle normal
Stop-hibernate behavior disabled	AMI Launch index 0	Key pair name django_app

- 3.

4. You can add elastic IPS by following the below instructions. You have to go under network and security. Select Elastic IPs.

▼ Network & Security

Security Groups

Elastic IPs

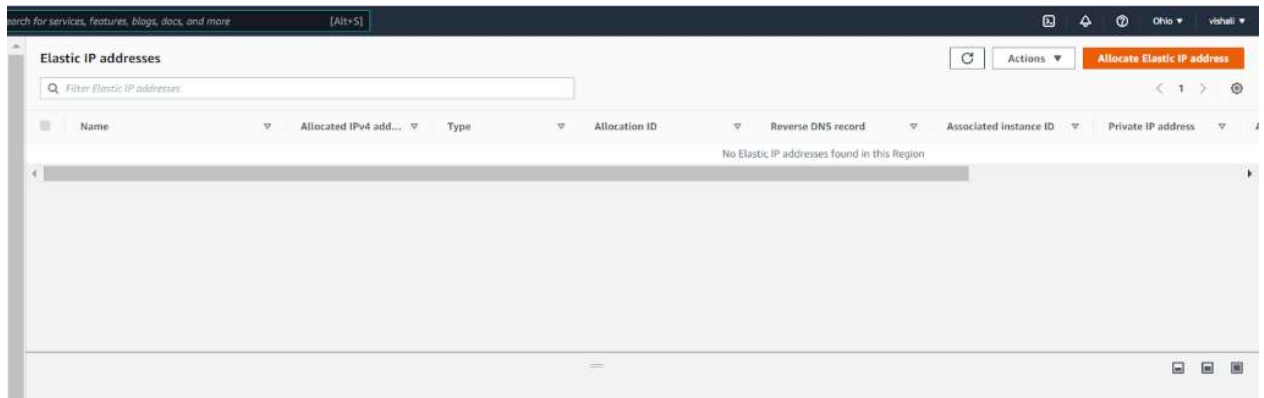
Placement Groups

Key Pairs

Network Interfaces

5.

6. Select allocate Elastic IP address



7.

8. Press Allocate

EC2 > Elastic IP addresses > Allocate Elastic IP address

Allocate Elastic IP address [Info](#)

Elastic IP address settings [Info](#)

Public IPv4 address pool

- ☒ Amazon's pool of IPv4 addresses
- ☐ Public IPv4 address that you bring to your AWS account (option disabled because no pools found) [Learn more](#)
- ☐ Customer owned pool of IPv4 addresses (option disabled because no customer owned pools found) [Learn more](#)

Global static IP addresses

AWS Global Accelerator can provide global static IP addresses that are announced worldwide using anycast from AWS edge locations. This can help improve the availability and latency for your user traffic by using the Amazon global network. [Learn more](#)

[Create accelerator](#)

Tags - optional

A tag is a label that you assign to an AWS resource. Each tag consists of a key and an optional value. You can use tags to search and filter your resources or track your AWS costs.

No tags associated with the resource.

[Add new tag](#)

You can add up to 50 more tag

[Cancel](#)
[Allocate](#)

9.

Elastic IP address allocated successfully.
Elastic IP address 18.191.48.162

Associate this Elastic IP address

Elastic IP addresses (1/1)

Public IPv4 address: 18.191.48.162

Clear filters

Actions

Allocate Elastic IP address

< 1 >

<input checked="" type="checkbox"/>	Name	Allocated IPv4 add...	Type	Allocation ID	Reverse DNS record	Associated instance ID	Private IP address
<input checked="" type="checkbox"/>	-	18.191.48.162	Public IP	eipalloc-07500950f19a223ff	-	-	-

10.

11. Go to actions -> Associate Elastic IP Address:



12.

13. Provide the IP Address of EC2 instance and press associate:

EC2 > Elastic IP addresses > Associate Elastic IP address

Associate Elastic IP address

Choose the instance or network interface to associate to this Elastic IP address (18.191.48.162)

Elastic IP address: 18.191.48.162

Resource type
Choose the type of resource with which to associate the Elastic IP address.

☒ Instance

☐ Network interface

Warning: If you associate an Elastic IP address to an instance that already has an Elastic IP address associated, this previously associated Elastic IP address will be disassociated but still allocated to your account. [Learn more](#)

Instance

Private IP address
The private IP address with which to associate the Elastic IP address.

Reassociation
Specify whether the Elastic IP address can be reassociated with a different resource if it already associated with a resource.

☐ Allow this Elastic IP address to be reassociated

Cancel Associate

14.

15. You can use Putty terminal to connect to EC2 instance. Provide username as ubuntu.

Install CodeDeploy Agent on EC2 Instance

To deploy the git repo by using CodeDeploy Service, codeDeploy-agent must install in the EC2 instance. Use the below commands to install codedeploy-agent.

Use the below commands to install codedeploy-agent.

```
sudo yum update
```

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

```
sudo yum install wget
```

```
wget https://bucket-name.s3.region-identifier.amazonaws.com/latest/install
```

bucket-name is the Amazon S3 bucket containing the CodeDeploy Resource Kit files for your region.
region-identifier is the identifier for your region.

You can find the list of bucket names and region identifiers [here](#).

For example:

```
wget https://aws-codedeploy-us-east-2.s3.us-east-2.amazonaws.com/latest/install
```

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

```
sudo ./install auto
```

```
sudo service codedeploy-agent start
```

```
starting codedeploy-agent: [ec2-user@ip-172-31-29-19 ~]$
```

```
ctrl + c
```

Firewall Setup:

Enable Firewall and allow OpenSSH

```
sudo ufw enable
```

```
sudo ufw allow OpenSSH
```

```
sudo ufw allow 'Nginx Full'
```

Check to make sure we are allowing OpenSSH:

sudo ufw status

Expected output:



Install the python nginx and git:

sudo apt-get update

sudo apt-get install python-pip python-dev nginx git

sudo pip install virtualenv

git clone https://github.com/Coder-Vishali/django_app.git

cd django_app

Setup the virtual environment & install necessary packages

python -m venv myenv

source myenv/bin/activate

pip install -r requirements.txt

pip install django bcrypt django-extensions

pip install gunicorn

Add the EC2 instance IP in the Allowed host of django settings.py file:

cd django_app

sudo vim settings.py

Inside settings.py modify these lines allowed host public IP address

ALLOWED_HOSTS = ['13.59.206.93']

add the line below to the bottom of the file

STATIC_ROOT = os.path.join(BASE_DIR, "static/")

Save your changes and quit. ESC :wq

cd ..

Bind the django with gunicorn:

python manage.py collectstatic

```
gunicorn --bind 0.0.0.0:8000 django_app.wsgi:application  
ctrl+c
```

Edit the gunicorn service:

```
sudo vim /etc/systemd/system/gunicorn.service
```

```
[Unit]
```

```
Description=gunicorn daemon
```

```
After=network.target
```

```
[Service]
```

```
User=ubuntu
```

```
Group=www-data
```

```
WorkingDirectory=/home/ubuntu/django_app
```

```
ExecStart=/home/ubuntu/django_app/venv/bin/gunicorn --workers 3 --bind  
unix:/home/ubuntu/django_app/django_app.sock django_app.wsgi:application
```

```
[Install]
```

```
WantedBy=multi-user.target
```

ESC :wq

Start and Enable the gunicorn service:

```
sudo chown ubuntu:www-data /home/ubuntu/django_app/
```

```
pkill gunicorn
```

```
sudo systemctl daemon-reload
```

```
sudo systemctl start gunicorn
```

```
sudo systemctl enable gunicorn
```

```
sudo systemctl status gunicorn.service
```

Edit the nginx configurations file:

```
sudo vim /etc/nginx/sites-available/django_app
```

```
server {
```

```
    listen 80;
```

```
    server_name 18.191.48.162;
```



```

location = /favicon.ico { access_log off; log_not_found off; }

location /static/ {

    root /home/ubuntu/django_app;

}

location /media/ {

    root /home/ubuntu/django_app;

}

location / {

    include proxy_params;

    proxy_pass http://unix:/home/ubuntu/django\_app/django\_app.sock;;

}
}

```

ESC :wq

Start the nginx service:

```
sudo ln -sf /etc/nginx/sites-available/django_app /etc/nginx/sites-enabled
```

```
sudo nginx -t
```

```
sudo rm /etc/nginx/sites-enabled/default
```

```
sudo service nginx restart
```

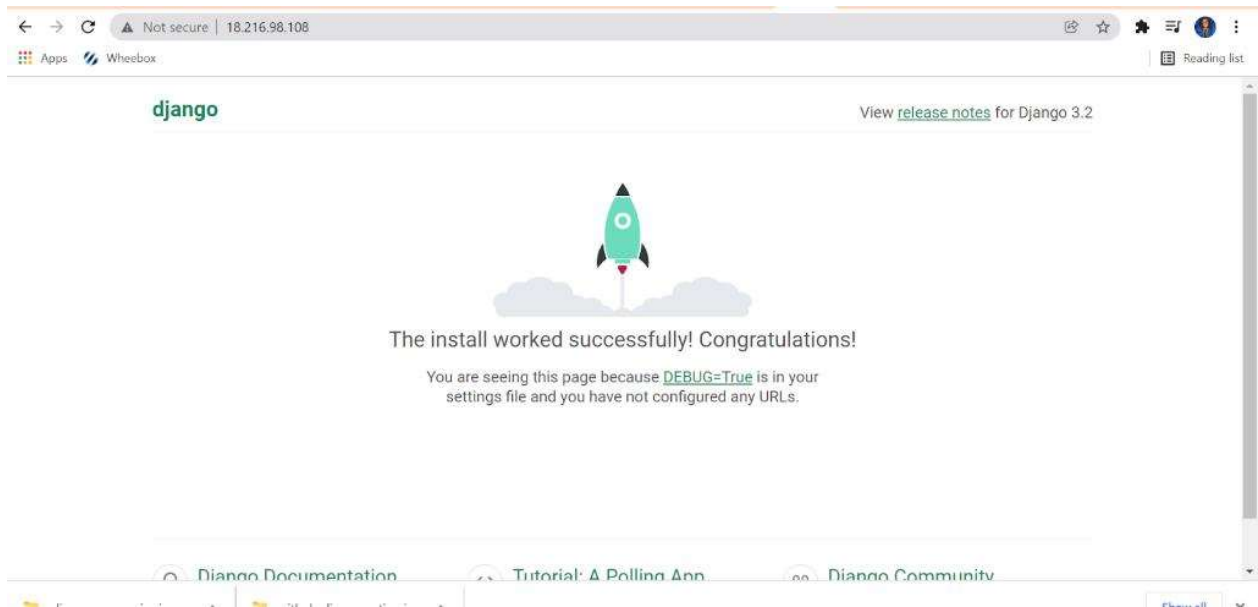
```
sudo systemctl status nginx
```

```

(venv) ubuntu@ip-172-31-24-93:~/django_app$ sudo chown ubuntu:www-data /home/ubuntu/django_app/
(venv) ubuntu@ip-172-31-24-93:~/django_app$ pkill gunicorn
(venv) ubuntu@ip-172-31-24-93:~/django_app$ sudo systemctl daemon-reload
(venv) ubuntu@ip-172-31-24-93:~/django_app$ sudo systemctl start gunicorn
(venv) ubuntu@ip-172-31-24-93:~/django_app$ sudo systemctl enable gunicorn
(venv) ubuntu@ip-172-31-24-93:~/django_app$ sudo systemctl status gunicorn.service
● gunicorn.service - gunicorn daemon
   Loaded: loaded (/etc/systemd/system/gunicorn.service; enabled; vendor preset: enabled)
   Active: active (running) since Mon 2022-01-17 10:51:32 UTC; 34s ago
 Main PID: 27978 (gunicorn)
    Tasks: 4 (limit: 1140)
   CGroup: /system.slice/gunicorn.service
           └─27978 /home/ubuntu/django_app/venv/bin/python /home/ubuntu/django_app/venv/bin/gunicorn --access-logfile - --workers 3 --bind unix:/home/ubuntu/
             └─28000 /home/ubuntu/django_app/venv/bin/python /home/ubuntu/django_app/venv/bin/gunicorn --access-logfile - --workers 3 --bind unix:/home/ubuntu/
             └─28004 /home/ubuntu/django_app/venv/bin/python /home/ubuntu/django_app/venv/bin/gunicorn --access-logfile - --workers 3 --bind unix:/home/ubuntu/
             └─28005 /home/ubuntu/django_app/venv/bin/python /home/ubuntu/django_app/venv/bin/gunicorn --access-logfile - --workers 3 --bind unix:/home/ubuntu/

Jan 17 10:51:32 ip-172-31-24-93 systemd[1]: Started gunicorn daemon.
Jan 17 10:51:32 ip-172-31-24-93 gunicorn[27978]: [2022-01-17 10:51:32 +0000] [27978] [INFO] Starting gunicorn 20.1.0
Jan 17 10:51:32 ip-172-31-24-93 gunicorn[27978]: [2022-01-17 10:51:32 +0000] [27978] [INFO] Listening at: unix:/home/ubuntu/django_app/django_app.sock (27978)
Jan 17 10:51:32 ip-172-31-24-93 gunicorn[27978]: [2022-01-17 10:51:32 +0000] [27978] [INFO] Using worker: sync
Jan 17 10:51:32 ip-172-31-24-93 gunicorn[27978]: [2022-01-17 10:51:32 +0000] [28000] [INFO] Booting worker with pid: 28000
Jan 17 10:51:32 ip-172-31-24-93 gunicorn[27978]: [2022-01-17 10:51:32 +0000] [28004] [INFO] Booting worker with pid: 28004
Jan 17 10:51:32 ip-172-31-24-93 gunicorn[27978]: [2022-01-17 10:51:32 +0000] [28005] [INFO] Booting worker with pid: 28005
lines 1-10/10 (END)

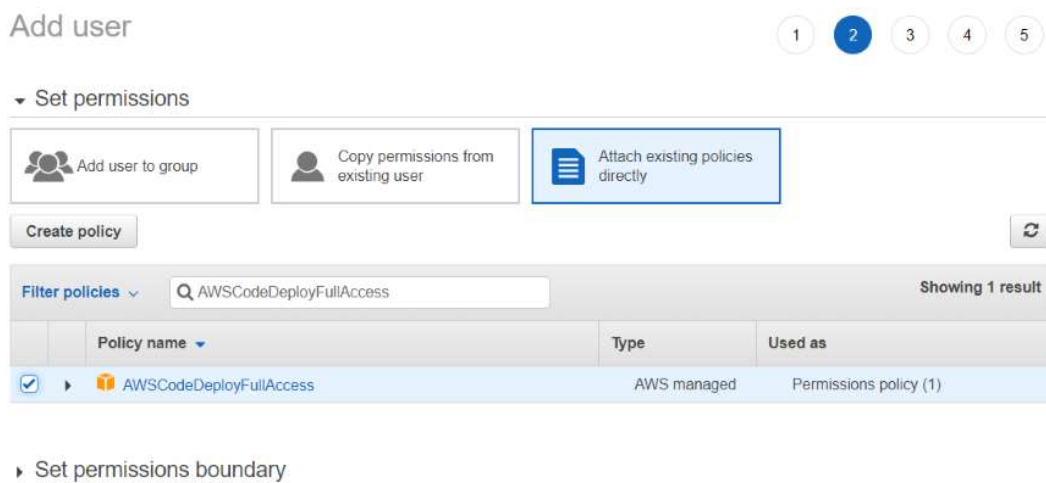
```



GitHub Action

First, create an [IAM user](#) with full `AWSCodeDeployFullAccess` policy and generate an [access key and secret access](#) for the user to configure GitHub Action.

Before configuring Action, set the environment in the GitHub repository.



Add user

1 2 3 4 5

Review

Review your choices. After you create the user, you can view and download the autogenerated password and access key.

User details

User name	admin
AWS access type	AWS Management Console access - with a password
Console password type	Custom
Require password reset	Yes
Permissions boundary	Permissions boundary is not set

Permissions summary

The following policies will be attached to the user shown above.

Type	Name
Managed policy	AWSCodeDeployFullAccess
Managed policy	AdministratorAccess
Managed policy	IAMUserChangePassword

Tags

Cancel

Previous

Create user

Click on Create user. IAM User will be created.

To generate the access key, click on **Create access key**:

Summary

Delete user ⓘ

User ARNiam:aws:iam::537507758362:user/admin ⓘ

Path /

Creation time2022-01-10 14:57 UTC+0530

Permissions

Groups

Tags

Security credentials

Access Advisor

Sign-in credentials

Summary

• Console sign-in link: <https://537507758362.signin.aws.amazon.com/console> ⓘ

Console passwordEnabled (never signed in) | [Manage](#)

Assigned MFA deviceNot assigned | [Manage](#)

Signing certificatesNone ⓘ

Access keys

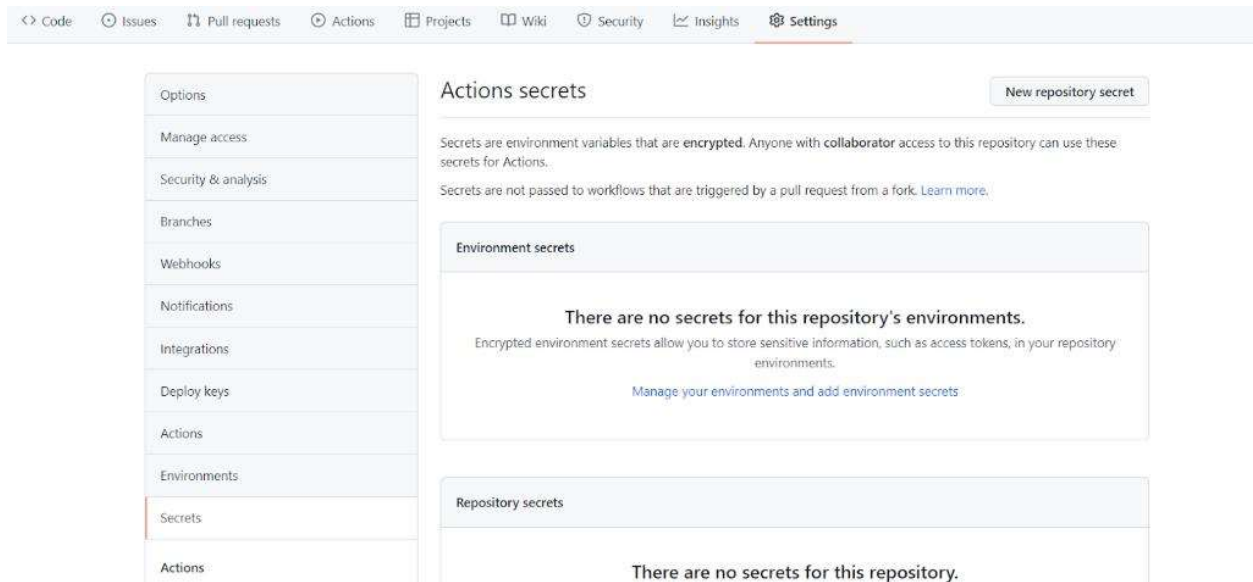
Use access keys to make programmatic calls to AWS from the AWS CLI, Tools for PowerShell, AWS SDKs, or direct AWS API calls. You can have a maximum of two access keys (active or inactive) at a time.

For your protection, you should never share your secret keys with anyone. As a best practice, we recommend frequent key rotation. [If you lose or forget your secret key, you cannot retrieve it. Instead, create a new access key and make the old key inactive. Learn more](#)

Create access key

Access key ID	Created	Last used	Status
---------------	---------	-----------	--------

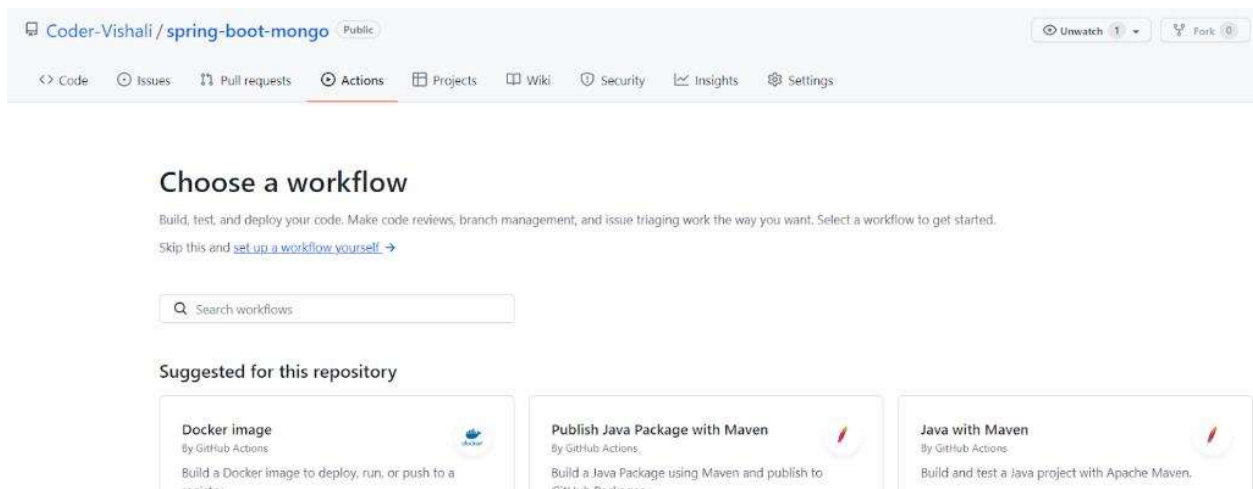
Before configuring Action, set the environment in the GitHub repository.



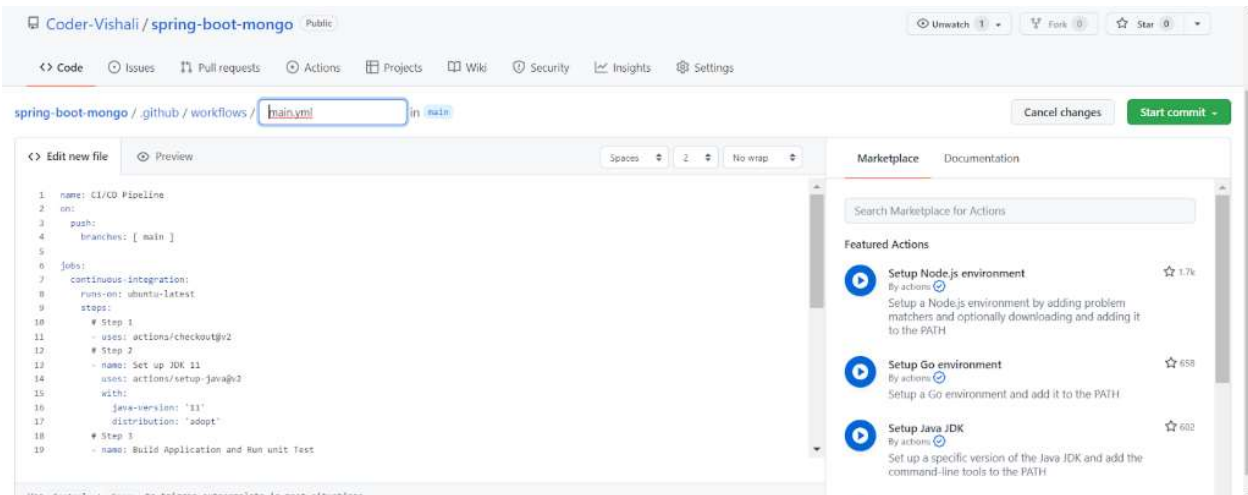
Click on New repository secret and add the following secrets:



GitHub repository changes will trigger [GitHub Action](#),



Here, you can configure your actions:

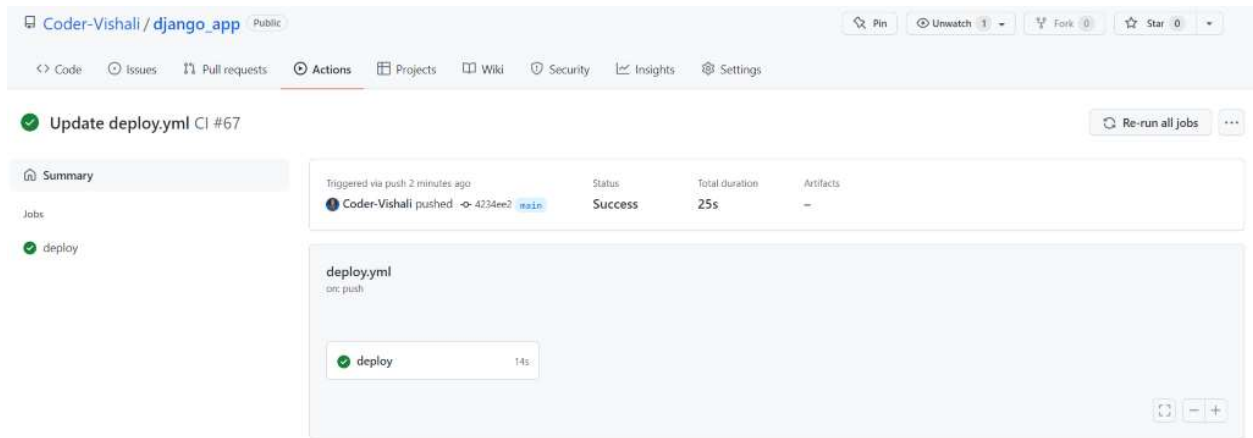


Now make a change to your repository. Your changes should automatically deploy to your EC2 server. Access the application from a web browser or postman.

curl --location --request GET 'http://{{ec2_public_ip}}:8000'



Look into GitHub Actions:



Deployment status:

LifecycleEvent - ApplicationStart

Script - scripts/start_server

```
[stderr][2022-01-20 08:09:22 +0000] [16810] [INFO] Starting gunicorn 20.1.0
[stderr][2022-01-20 08:09:22 +0000] [16810] [INFO] Listening at: http://0.0.0.0:8000 (16810)
[stderr][2022-01-20 08:09:22 +0000] [16810] [INFO] Using worker: sync
```

```
[stderr][2022-01-20 08:09:22 +0000] [16813] [INFO] Booting worker with pid: 16813
[stderr]
```

```
[stderr]Session terminated, terminating shell...[2022-01-20 08:14:21 +0000] [16813] [INFO] Worker exiting
(pid: 16813)
```

```
[stderr] ...terminated.
```

Event Logs

Event details

Error code

ScriptTimedOut

Script name

scripts/start_server

Message

Script at specified location: scripts/start_server failed to complete in 300 seconds

Logs

```
LifecycleEvent - ApplicationStart
Script - scripts/start_server
[stderr][2022-01-20 08:09:22 +0000] [16810] [INFO] Starting gunicorn 20.1.0
[stderr][2022-01-20 08:09:22 +0000] [16810] [INFO] Listening at: http://0.0.0.0:8000 (16810)
[stderr][2022-01-20 08:09:22 +0000] [16810] [INFO] Using worker: sync
[stderr][2022-01-20 08:09:22 +0000] [16813] [INFO] Booting worker with pid: 16813
[stderr]
[stderr]Session terminated, terminating shell...[2022-01-20 08:14:21 +0000] [16813] [INFO] Worker exiting (pid: 16813)
[stderr] ...terminated.
```

Reference:

- Django app on aws: <https://dev.to/rmiyazaki6499/deploying-a-production-ready-django-app-on-aws-1pk3>
- Guicorn and nginx setup: <https://www.youtube.com/watch?v=QjrfUO91wfc>
- Github code: https://github.com/Coder-Vishali/django_app