

Developer Guide for Hackathon using Java Language - Sep 2023

Introduction:

This document explains step by step how to use Apprunner and other AWS Services during the Hackathon for CITI developers, Also this showing a Sample code using [Java Language](#) running on Apprunner .

Accessing AWS environment for the Hackathon

Navigate to the link provided to you by the instructor. If you do not have a specific URL for this event, you can navigate to <https://catalog.us-east-1.prod.workshops.aws/join> and you will be prompted for an event code later.

You will need to login to the environment using one of the two indicated methods below.

Workshop Studio > Sign in

Sign in
Choose a preferred sign-in method

Email one-time password (OTP)
Enter your personal or corporate email to receive a one-time password

Login with Amazon
Login with your Amazon.com retail account

Amazon employee
Login with your Amazon Corporate account. Only for Amazon Employees.

If you are prompted for an event code, enter the value supplied by the lab instructor.

Enter event access code

Event access code

Event access code

A 12 digit code that was given to you for this event

Cancel

Next

Finish the login process

You will be prompted to accept the terms and conditions of the event. Read the page, and then select the **I agree with the Terms and Conditions** checkbox. Choose **Join event**.

Review and join

Event details

Name	Start time	Duration	Level
AWS App Runner Workshop Event	10/11/2022 10:00 AM	12 hours	-

Description

AWS App Runner Workshop

Terms and Conditions

Read and accept before joining the event

1. By using AWS Workshop Studio for the relevant event, you agree to the AWS Event Terms and Conditions and the AWS Acceptable Use Policy. You acknowledge and agree that are using an AWS-owned account that you can only access for the duration of the relevant event. If you find residual resources or materials in the AWS-owned account, you will make us aware and cease use of the account. AWS reserves the right to terminate the account and delete the contents at any time.
2. You will not: (a) process or run any operation on any data other than test data sets or lab-approved materials by AWS, and (b) copy, import, export or otherwise create derivative works of materials provided by AWS, including but not limited to, data sets.
3. AWS is under no obligation to enable the transmission of your materials through Event Engine and may, in its discretion, edit, block, refuse to post, or remove your materials at any time.
4. Your use of AWS Workshop Studio will comply with these terms and all applicable laws, and your access to AWS Workshop Studio will immediately and automatically terminate if you do not comply with any of these terms or conditions.

☐ I agree with the Terms and Conditions

Cancel

Previous

Join event

The workshop dashboard will be shown. At any time, to log into the AWS console, choose the **Open AWS Console** link in the **AWS account access menu**.

← → ↻ catalog.us-east-1.prod.workshops.aws/event/dashboard/en-US

aws workshop studio

Citi-Hackathon ×

Event ends in 2 days 23 hours 49 minutes.

[Event dashboard](#) > AWS App Runner for .NET

Citi-Hackathon

Event information

Start time	Duration	Accessible regions
8/15/2023 11:09 PM	72 hours	us-east-1

Description
test for Citi Hackathon


Workshop

Title	Complexity level	AWS service
AWS App Runner Workshop	300	AWS App Runner

AWS App Runner for .NET

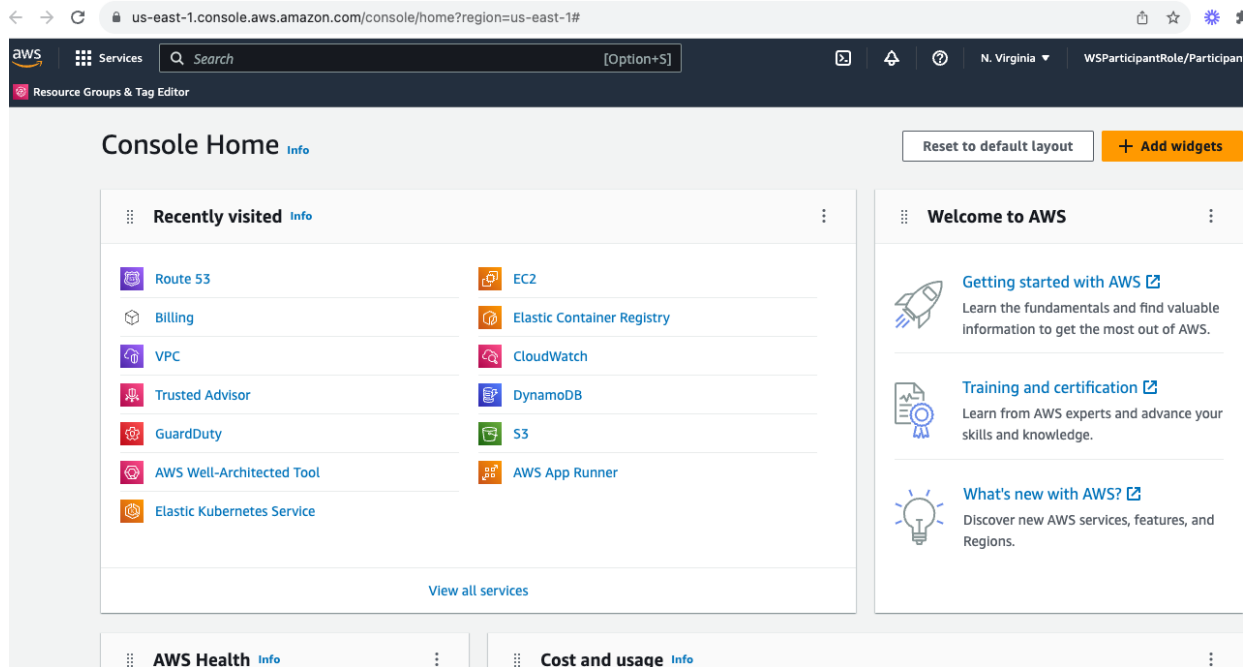
- ▶ Getting started
- ▶ Lab Option 1: Deploy from source code
- ▶ Lab Option 2: Deploy from a container image

▼ AWS account access

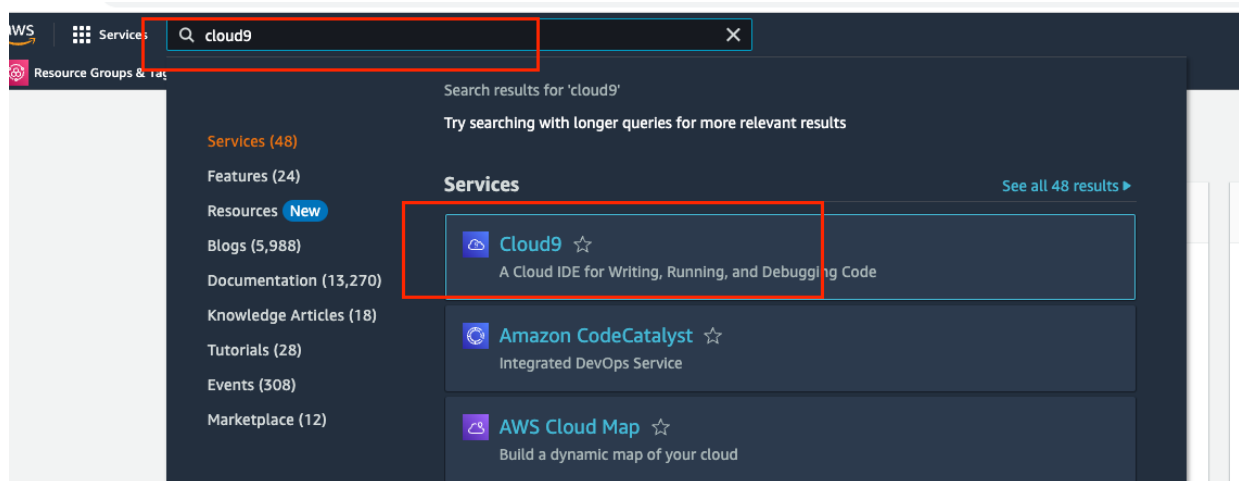
- [Open AWS console \(us-east-1\)](#) 
- [Get AWS CLI credentials](#)
- [Get EC2 SSH key](#)

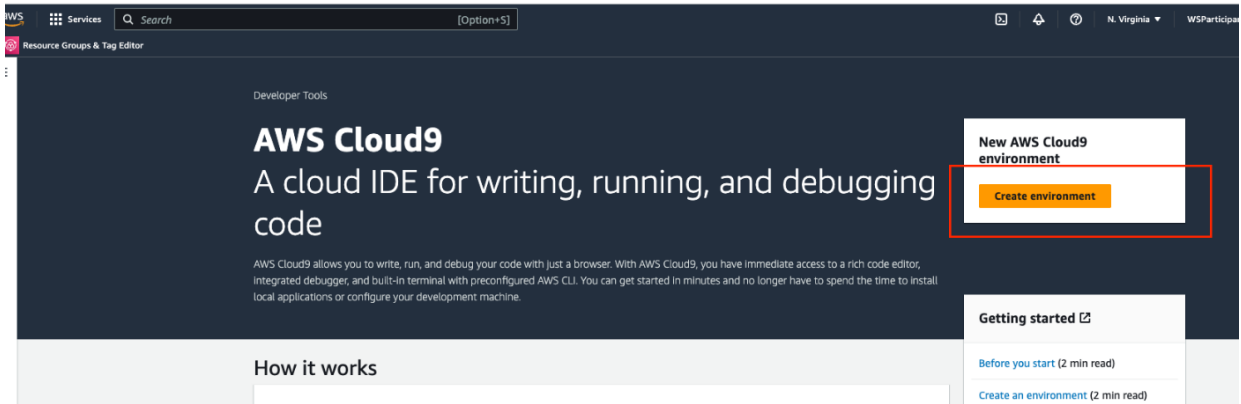
Exit event

Make sure you have access to AWS Services in the Console as shown below



Create a Cloud 9 Environment which will help on executing the AWS CLI commands and running big jobs (if needed) without disturbance





Give it a name and change the EC2 Instance size to be t3.small - Then Create the environment without changing any other configurations.

Create environment [info](#)

Details

Name

 Limit of 60 characters, alphanumeric, and unique per user.

Description - optional

 Limit 200 characters.

Environment type [info](#)
 Determines what the Cloud9 IDE will run on.

☒ **New EC2 instance**
 Cloud9 creates an EC2 instance in your account. The configuration of your EC2 instance cannot be changed by Cloud9 after creation.

☐ **Existing compute**
 You have an existing instance or server that you'd like to use.

New EC2 instance

Instance type [info](#)
 The memory and CPU of the EC2 instance that will be created for Cloud9 to run on.

☒ **t2.micro (1 GiB RAM + 1 vCPU)**
 Free-tier eligible. Ideal for educational users and exploration.

☐ **t3.small (2 GiB RAM + 2 vCPU)**
 Recommended for small web projects.

☐ **m5.large (8 GiB RAM + 2 vCPU)**
 Recommended for production and most general-purpose development.

☐ **Additional instance types**
 Explore additional instances to fit your need.

Platform [info](#)
 This will be installed on your EC2 instance. We recommend Amazon Linux 2.

Timeout
 How long Cloud9 can be inactive (no user input) before auto-hibernating. This helps prevent unnecessary charges.

Network settings [info](#)

Connection
 How your environment is accessed.

☒ **AWS Systems Manager (SSM)**
 Accesses environment via SSM without opening inbound ports (no ingress).

☐ **Secure Shell (SSH)**
 Accesses environment directly via SSH, opens inbound ports.

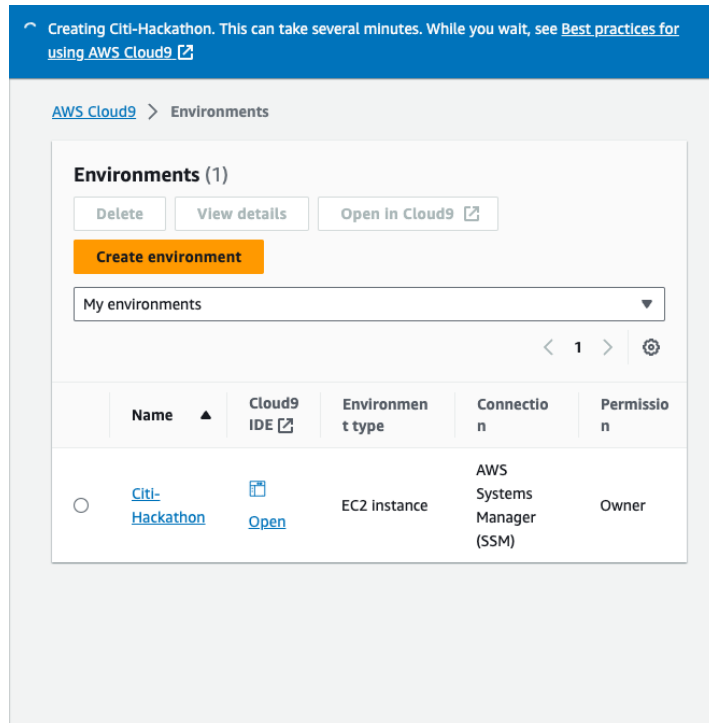
[VPC settings](#) [info](#)

Tags - optional [info](#)
 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.

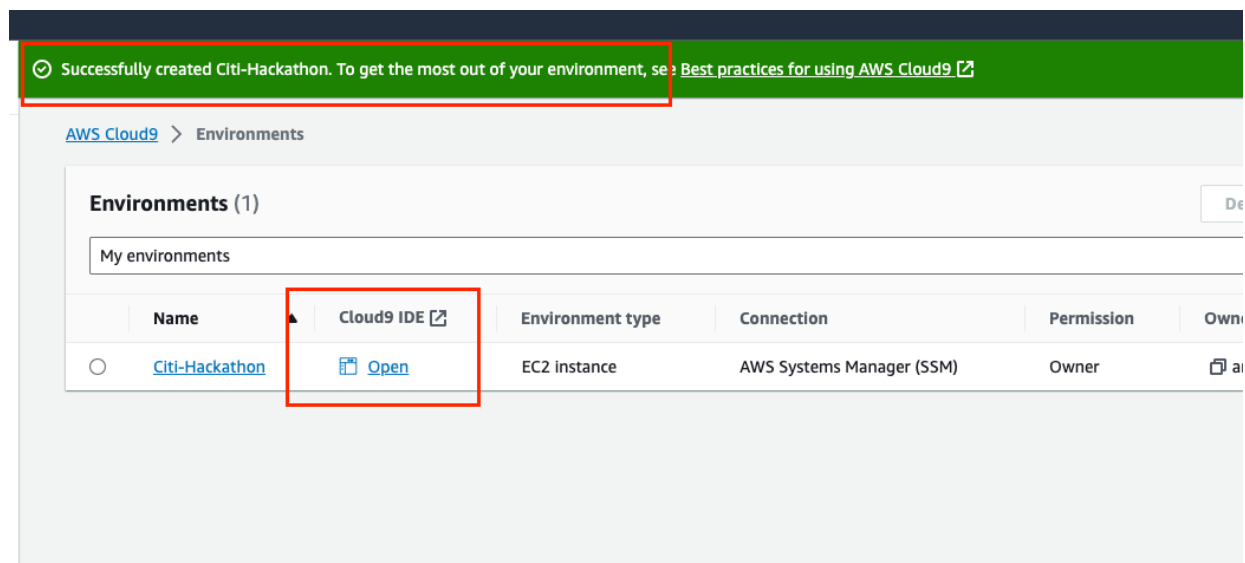
The following IAM resources will be created in your account

- AWSServiceRoleForAWSCloud9** - AWS Cloud9 creates a service-linked role for you. This allows AWS Cloud9 to call other AWS services on your behalf. You can delete the role from the AWS IAM console once you no longer have any AWS Cloud9 environments. [Learn more](#)
- AWSCloud9SSMAccessRole** and **AWSCloud9SSMInstanceProfile** - A service role and an instance profile are automatically created if Cloud9 accesses its EC2 instance through AWS Systems Manager. If your environments no longer require EC2 instances that block incoming traffic, you can delete these roles using the AWS IAM console. [Learn more](#)

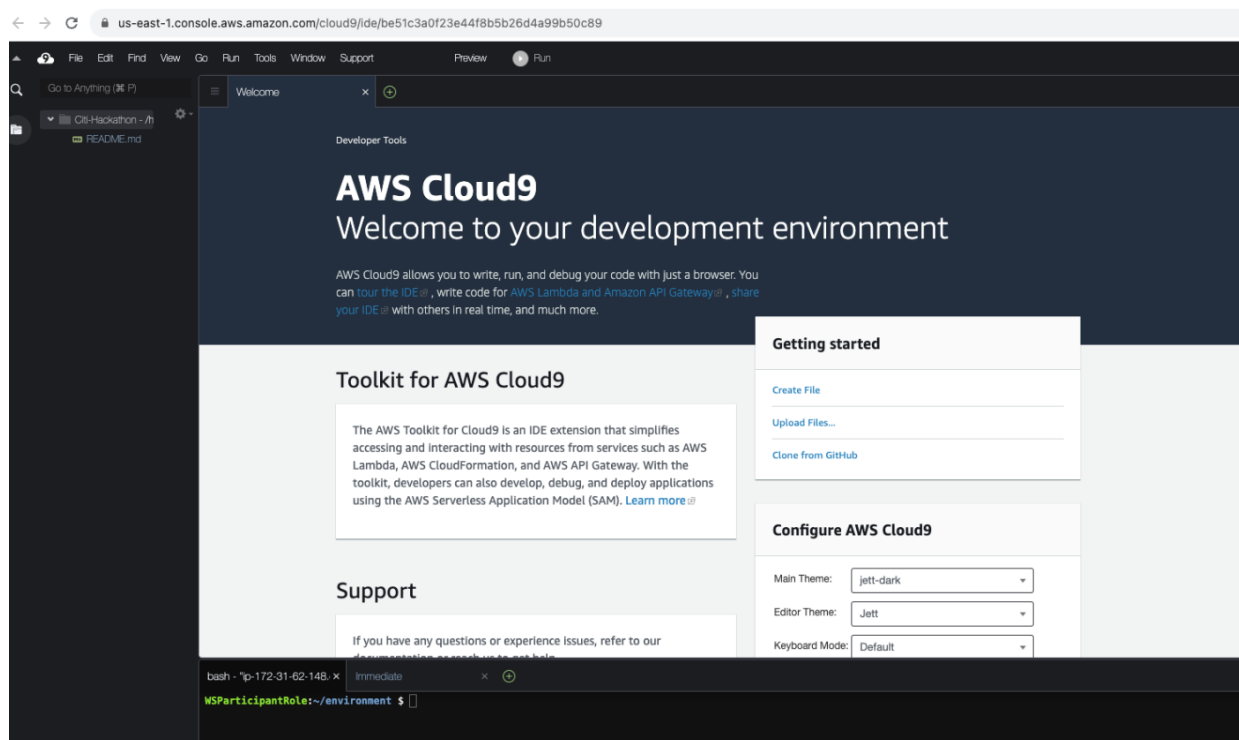
This can take couple of minutes to create your EC2 Instance for Cloud9



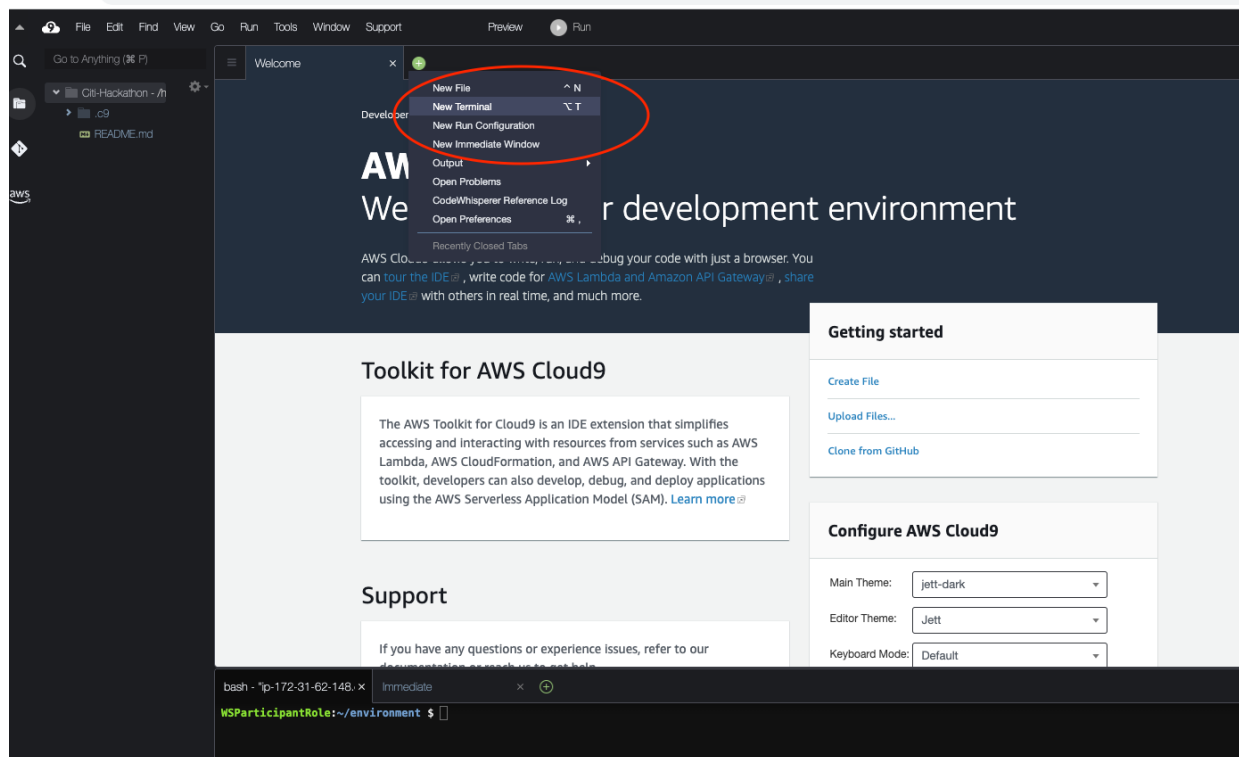
Once creation completed Successfully, Click Open :

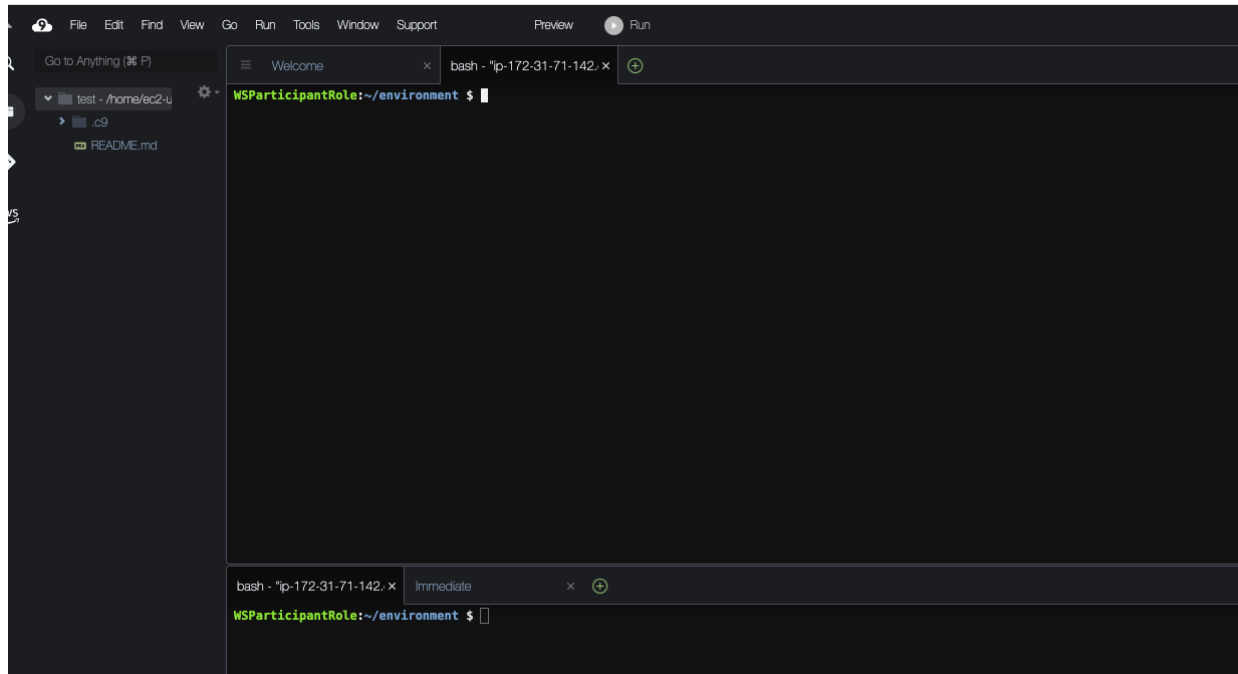


This is the Terminal of your Cloud9 Environment (Ec2 Instance)

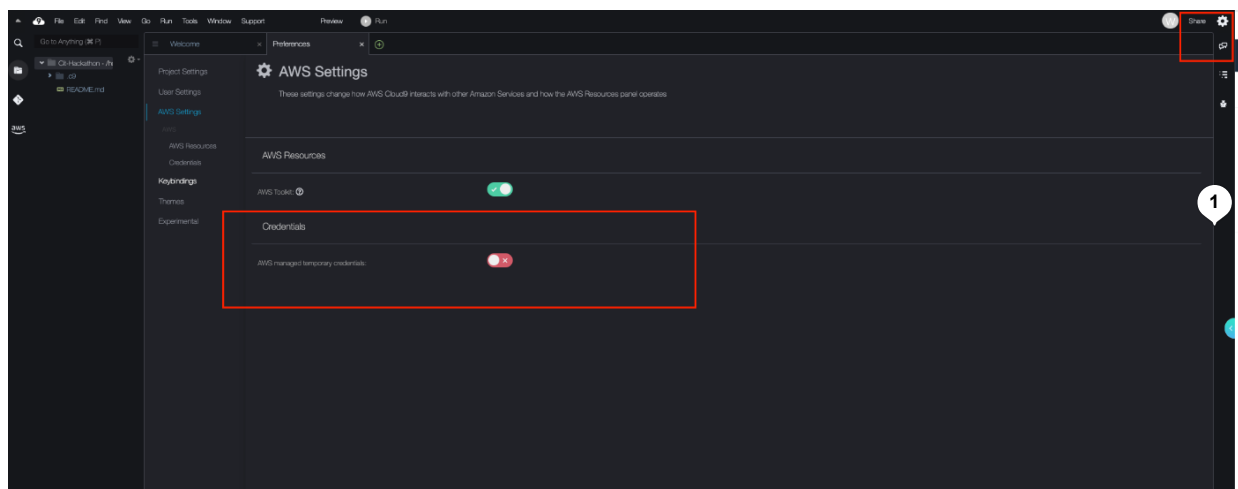


You can open a new terminal using the green "+" sign as shown below:





you need to disable the AWS Temporary credential from the Cloud9 machine from the machine Setting , you can find setting button on the up right corner then make sure AWS managed temporary credential is disabled as shown below:

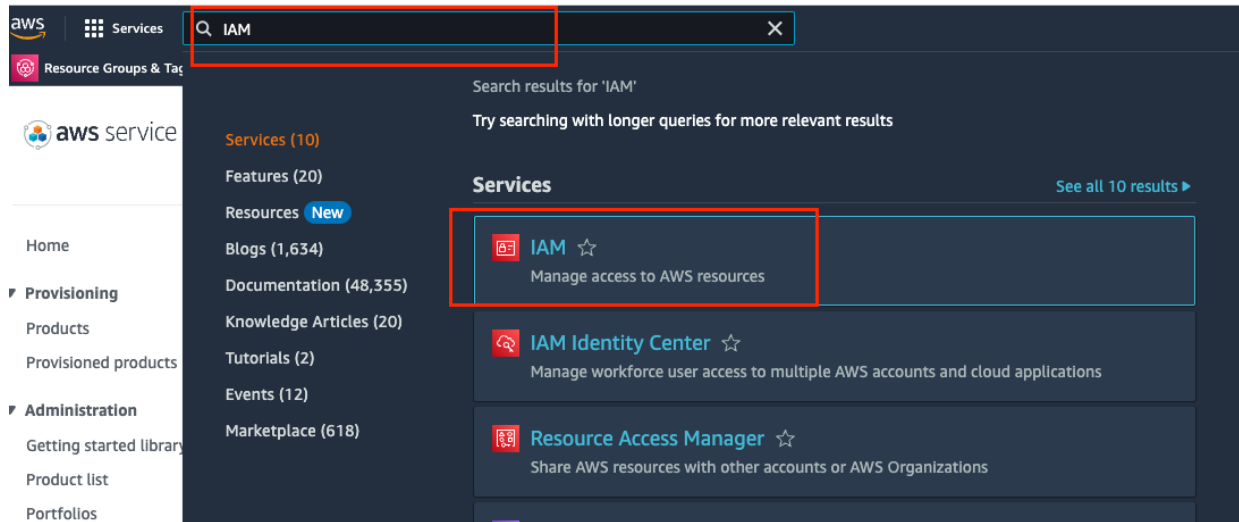


Now, We need to assign a proper IAM Permission to Cloud9 EC2 instance to be able to execute AWS CLI commands on other AWS Services like DynamoDB and S3

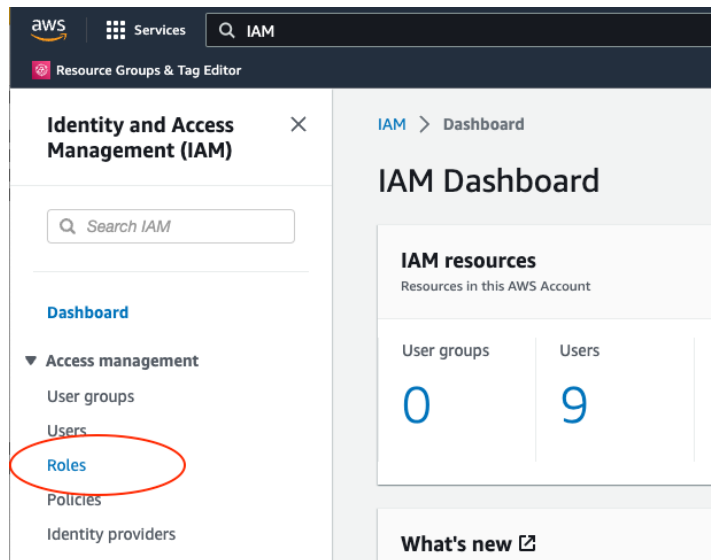
First you need to create Admin Role to attach it to Cloud9 :

Please Note: for the Demo purposes we choose the Full Access permissions for Cloud9 Ec2 instance but best practice here is to always give the least privilege permissions .

Go back to AWS console and Search for IAM on the Service search bar as below:



Go to Roles on the left side:



Then create a new role which will be attached to EC2 Instance:

Q IAM

Global Admin/mmasaud-isengard @ 6480-6410-7398

SS 4)

IAM > Roles

Roles (179) 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 2 3 4 5 6 7 8 9 > ⚙

<input type="checkbox"/>	Role name	Trusted entities
<input type="checkbox"/>	AccessAnalyzerMonitorServiceRole_OOPG9DNL5J	AWS Service: access-analyzer
<input type="checkbox"/>	AccessAnalyzerTrustedService	Account: 121925031476

Resource Groups & Tag Editor

IAM > Roles > Create role

Step 1: Select trusted entity

Step 2: Add permissions

Step 3: Name, review, and create

Select trusted entity

Trusted entity type

☒ AWS service
Allow AWS services like EC2, Lambda, or others to perform actions in this account.

☐ AWS account
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.

☐ Web identity
Allow users federated by the specified external web identity provider to assume this role to perform actions in this account.

☐ SAML 2.0 federation
Allow user federation with SAML 2.0 from a corporate directory to perform actions in this account.

☐ Custom trust policy
Create a custom trust policy to enable others to perform actions in this account.

Use case

Service or use case

EC2

Choose a use case for the specified service.

EC2

Allows EC2 instances to call AWS services on your behalf.

EC2 Role for AWS Systems Manager

Allows EC2 instances to call AWS services like CloudWatch and Systems Manager on your behalf.

EC2 Spot Fleet Role

Allows EC2 Spot Fleet to request and terminate Spot instances on your behalf.

EC2 - Spot Fleet Auto Scaling

Allows Auto Scaling to create and update EC2 spot fleets on your behalf.

EC2 - Spot Fleet Tagging

Allows EC2 to search spot instances and attach tags to the launched instances on your behalf.

EC2 - Spot Instances

Allows EC2 Spot instances to launch and manage spot instances on your behalf.

EC2 - Spot Fleet

Allows EC2 Spot Fleet to launch and manage spot fleet instances on your behalf.

EC2 - Scheduled Instances

Allows EC2 Scheduled instances to manage instances on your behalf.

Cancel

Next

Step 1: Select trusted entity

Step 2: Add permissions

Step 3: Name, review, and create

Add permissions

Permissions policies (1/177)

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

Q: Admin

Filter by Type: All types 38 matches

Policy name	Type	Description
<input checked="" type="checkbox"/> AdministratorAccess	AWS managed - job function	Provides full access to AWS services and resources.
<input type="checkbox"/> AdministratorAccess-AWSIoT	AWS managed	Grants account administrative permissions while explicitly allowing direct access to resou...
<input type="checkbox"/> AdministratorAccess-AWSIoTDevice	AWS managed	Grants account administrative permissions. Explicitly allows developers and administrato...
<input type="checkbox"/> AmazonAPIGatewayAdministrator	AWS managed	Provides full access to create/delete APIs in Amazon API Gateway via the AWS Man...
<input type="checkbox"/> AmazonAppStreamAdministrator	AWS managed	This policy grants access to Amazon AppStream resources associated with the studio a...
<input type="checkbox"/> AmazonSecurityInspectorServiceCatalogProductServicePolicy	AWS managed	Service role policy used by the AWS Service Catalog service to provision products from a...
<input type="checkbox"/> AmazonSecurityInspector	AWS managed	Provides full access to Amazon Security Lake and related services needed to administer S...
<input type="checkbox"/> AmazonWorkSpacesAdmin	AWS managed	Provides access to Amazon WorkSpaces administrative actions via AWS SDK and CLI.
<input type="checkbox"/> AmazonWorkSpacesAdminReadOnlyAccess	AWS managed	Provides administrator access for packaging an application in Amazon WorkSpaces Applic...
<input type="checkbox"/> AWSAppSyncAdministrator	AWS managed	Provides administrative access to the AppSync service, though not enough to access via t...
<input type="checkbox"/> AWSAuditManagerAdministratorAccess	AWS managed	Provides administrative access to enable or disable AWS Audit Manager, update settings, ...
<input type="checkbox"/> AWSBackupOpsAndMaintenanceAccess	AWS managed	This policy is for backup administrators who use cross-account backup management to m...
<input type="checkbox"/> AWSBackupActions_ReadOnlyForResourceAdministrationWithIAM	AWS managed	This policy gives permissions to control AWS resources. For example, to start and stop EC...
<input type="checkbox"/> AWSCloudFormation	AWS managed	Provides administrator access to AWS CloudFormation.
<input type="checkbox"/> AWSCodeBuildReadOnlyAccess	AWS managed	Provides full access to AWS CodeBuild via the AWS Management Console.
<input type="checkbox"/> AWSCodeBuildAdministratorAccess	AWS managed	Provides full access to AWS CodeBuild via the AWS Management Console. Also attach Am...
<input type="checkbox"/> AWSCodeGuruAdvisorReadOnlyAccess	AWS managed	DeepInspector admin access to all actions including toggling between multiuser and single u...
<input type="checkbox"/> AWSFMSReadOnlyAccess	AWS managed	Full access for AWS FMS Administrator
<input type="checkbox"/> AWSFMSReadOnlyAccess	AWS managed	Read only access for AWS FMS Administrator that allows monitoring AWS FMS operations
<input type="checkbox"/> AWSGlueReadOnlyAccess	AWS managed	Provides access within Amazon Glue to create and manage workspaces for the entire ...

Set permissions boundary - optional

Cancel Previous Next

give a Role Name (Please keep a record of that Role name as we need while assigning it to the Ec2 Instance for Cloud9 :

Name, review, and create

Role details

Role name
Enter a meaningful name to identify this role.

cloud9-admin-role

Maximum 64 characters. Use alphanumeric and "+,=,_,@,-" characters.

Description
Add a short explanation for this role.

Allows EC2 instances to call AWS services on your behalf.

Maximum 1000 characters. Use alphanumeric and "+,=,_,@,-" characters.

Step 1: Select trusted entities

Trust policy

```

1- {
2-   "Version": "2012-10-17",
3-   "Statement": [
4-     {
5-       "Effect": "Allow",
6-       "Action": [
7-         "sts:AssumeRole"
8-       ],
9-       "Principal": {
10-        "Service": [
11-          "ec2.amazonaws.com"
12-        ]
13-      }
14-    ]
15-  }
16- }
```

Step 2: Add permissions

Step 3: Name, review, and create

Role name
Enter a meaningful name to identify this role.
cloud9-admin-role

Description
Add a short explanation for this role.
Allow EC2 instances to call AWS services on your behalf.

Maximum 1000 characters. Use alphanumeric and "+, @, _" characters.

Step 1: Select trusted entities

Trust policy

```

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

```

Step 2: Add permissions

Permissions policy summary

Policy name	Type	Attached as
AdministratorAccess	AWS managed - job function	Permissions policy

Step 3: Add tags

Add tags - optional info
Tags are key-value pairs that you can add to AWS resources to help identify, organize, or search for resources.

No tags associated with the resource.

[Add new tag](#)
You can add up to 50 more tags.

[Cancel](#) [Previous](#) [Create role](#)

To validate role creation please type in role name created on Search and make sure it got created Successfully

Role cloud9-admin-role created.

[IAM](#) > Roles

Roles (30) [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.

1 match

<input type="checkbox"/>	Role name	Trusted entities	Last activity
<input type="checkbox"/>	cloud9-admin-role	AWS Service: ec2	-

Roles Anywhere [Info](#)

Authenticate your non AWS workloads and securely provide access to AWS services.

Access AWS from your non AWS workloads

Operate your non AWS workloads using the same authentication and authorization strategy that you use within AWS.

X.509 Standard

Use your own existing PKI infrastructure or use [AWS Certificate Manager Private Certificate Authority](#) to authenticate identities.

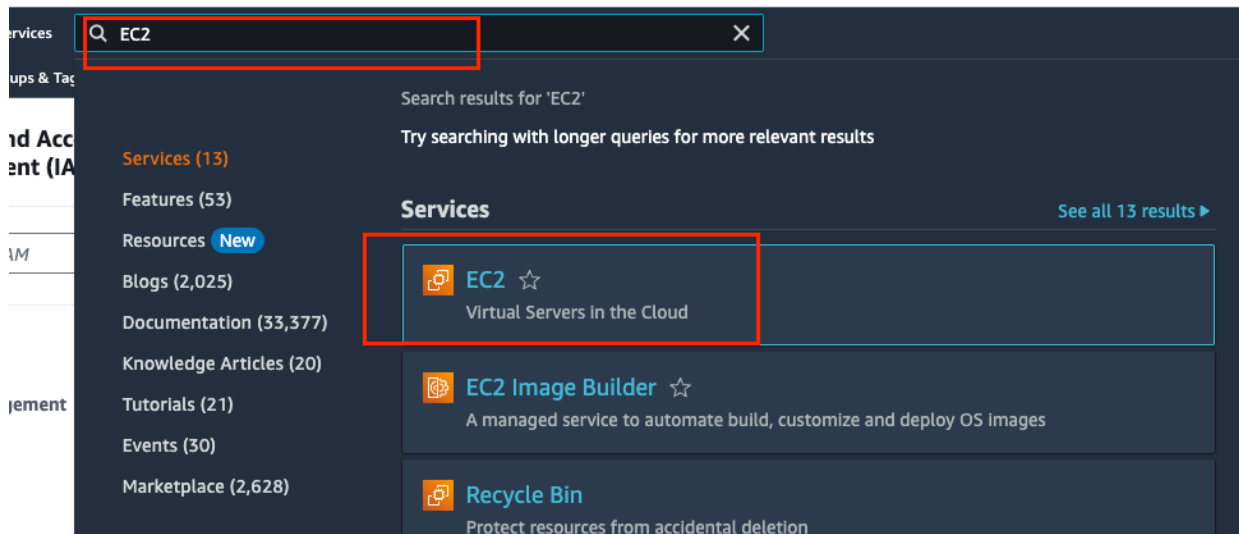
Temporary credentials

Use temporary credentials with ease and benefit from the enhanced security they provide.

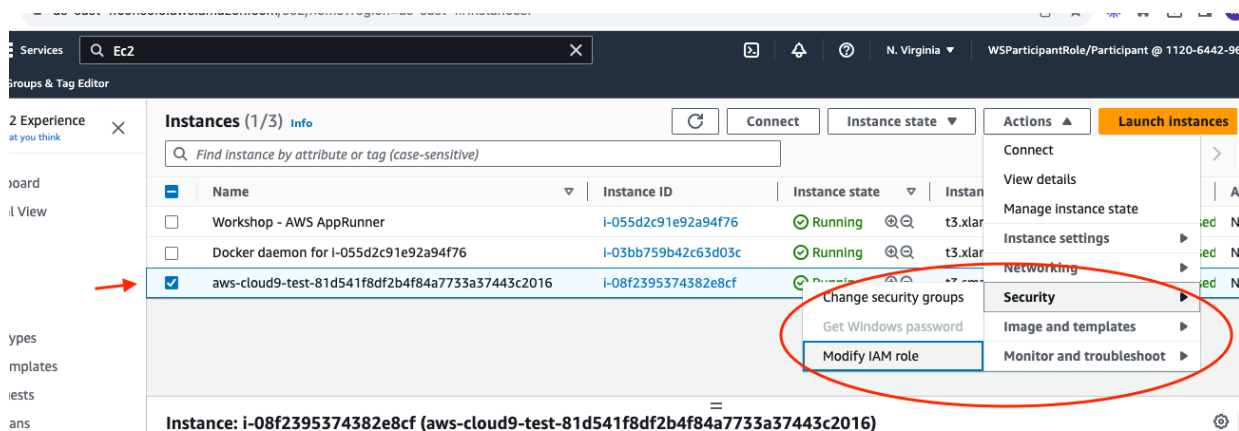
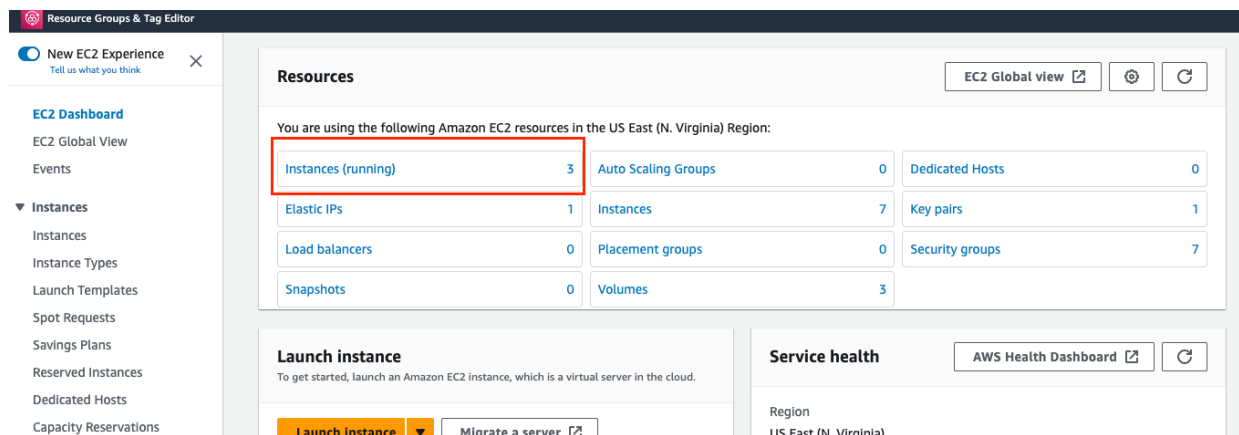
[Manage](#)

After creating the IAM Role you will need to attach it to the EC2 instance to enable Cloud9 to execute any AWS command needed based on the permissions you gave on that Role:

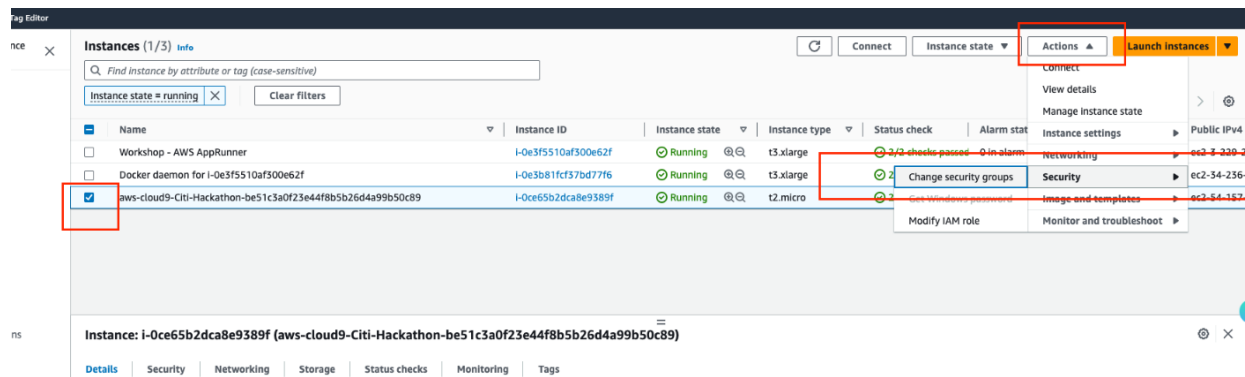
back to AWS Console and Look for EC2 as shown below :



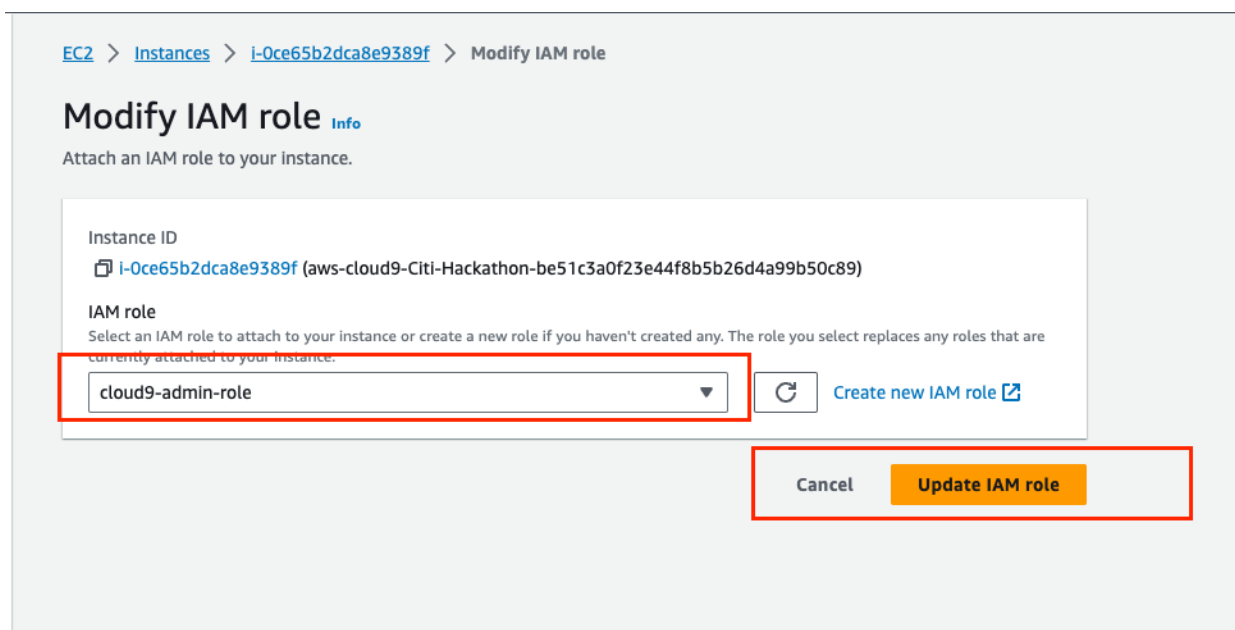
Click on the " Instance Running"



check the box for the EC2 instance related to Cloud9 then go to Security > Modify IAM Policy :



Select the newly created admin role and click Update IAM Role then Click on Update IAM Role :



After assigning the Proper Permissions to the Cloud9 machine, you can now you can execute AWS commands from Cloud9 environment as shown below :

1- Create S3 Bucket

give example from random number to actual number , open up word or notepad and past that there to copy it on AWS CLI on Cloud9

S3 bucket name must be Globally unique, so please make sure to add random number or date to make it unique:

```
aws s3api create-bucket --bucket bucket-hackathon-test-<random number>
```

```
WSParticipantRole:~/environment $  
WSParticipantRole:~/environment $  
WSParticipantRole:~/environment $ aws s3api create-bucket --bucket bucket-hackathon-test-20230913  
{  
  "Location": "/bucket-hackathon-test-20230913"  
}  
WSParticipantRole:~/environment $  
WSParticipantRole:~/environment $
```

2- Create dynamoDB table

For my Sample Code I'm using DynamoDB for my no-SQL Database, here is the command to create the table :

Note:

Please make sure to use the right region name while excusing that command below, for example I'm using us-east-1 so the command should be like below :

```
aws dynamodb create-table \  
  --table-name dynamo-table-demo \  
  --attribute-definitions \  
    AttributeName=Team, AttributeType=S \  
    AttributeName=YOE, AttributeType=S \  
  --key-schema \  
    AttributeName=Team, KeyType=HASH \  
    AttributeName=YOE, KeyType=RANGE \  
  --provisioned-throughput \  
    ReadCapacityUnits=5, WriteCapacityUnits=5 --region us-east-1
```

```
WSParticipantRole:~/environment $ aws dynamodb create-table \
> --table-name dynamo-table-demo \
> --attribute-definitions \
> AttributeName=Team,AttributeType=S \
> AttributeName=YOE,AttributeType=S \
> --key-schema \
> AttributeName=Team,KeyType=HASH \
> AttributeName=YOE,KeyType=RANGE \
> --provisioned-throughput \
> ReadCapacityUnits=5,WriteCapacityUnits=5 --region us-east-1

{
  "TableDescription": {
    "AttributeDefinitions": [
      {
        "AttributeName": "Team",
        "AttributeType": "S"
      },
      {
        "AttributeName": "YOE",
        "AttributeType": "S"
      }
    ],
    "TableName": "dynamo-table-demo",
    "KeySchema": [
      {
        "AttributeName": "Team",
        "KeyType": "HASH"
      },
      {
        "AttributeName": "YOE",
        "KeyType": "RANGE"
      }
    ],
    "TableStatus": "CREATING",
    "CreationDateTime": "2023-09-15T11:38:09.688000+00:00",
    "ProvisionedThroughput": {
      "NumberOfDecreasesToday": 0,
      "ReadCapacityUnits": 5,
      "WriteCapacityUnits": 5
    }
  }
}
```

3- Create IAM Role for app Runner:

Please Note: for the Demo purposes we choose the Full Access permissions to DynamoDB , best practices is to give the least privilege permissions for AppRunner which enable it to execute the code on the targeted services.

Finally, We need to assign IAM Role for Apprunner Service to enable the service communicate with S3 and DynamoDB while running the code, please follow the steps below so Apprunner can communicate with your Github account and with other AWS Service :

```
cat << EOF > apprunner-role-policy.json
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Action": "sts:AssumeRole",
      "Principal": {
        "Service": "tasks.apprunner.amazonaws.com"
      },
      "Effect": "Allow",
      "Sid": ""
    }
  ]
}
```


EOF

```
WSParticipantRole:~/environment $ ls
README.md
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $ cat << EOF > apprunner-role-policy.json
> {
>   "Version": "2012-10-17",
>   "Statement": [
>     {
>       "Action": "sts:AssumeRole",
>       "Principal": {
>         "Service": "tasks.apprunner.amazonaws.com"
>       },
>       "Effect": "Allow",
>       "Sid": ""
>     }
>   ]
> }
> EOF
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $ ls
apprunner-role-policy.json  README.md
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $
```

Create IAM Assume Role for apprunner to enable it communicate with Github and other API as needed :

```
aws iam create-role --role-name apprunner-role --assume-role-policy-document file://ap
```

```
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $ aws iam create-role --role-name apprunner-admin-role --assume-role-policy-document file://apprunner-role-policy.json
{
  "Role": {
    "Path": "/",
    "RoleName": "apprunner-admin-role",
    "RoleId": "AROAUQZTAWWPBFSPDY2",
    "Arn": "arn:aws:iam::282447775113:role/apprunner-admin-role",
    "CreateDate": "2023-09-15T11:45:12+00:00",
    "AssumeRolePolicyDocument": {
      "Version": "2012-10-17",
      "Statement": [
        {
          "Action": "sts:AssumeRole",
          "Principal": {
            "Service": "tasks.apprunner.amazonaws.com"
          },
          "Effect": "Allow",
          "Sid": ""
        }
      ]
    }
  }
}
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $
```

Give also permissions to APprunner to communicate with DynamoDB and S3 as We need it to run the code there

```
aws iam attach-role-policy --role-name apprunner-role --policy-arn arn:aws:iam::aws:po
aws iam attach-role-policy --role-name apprunner-role --policy-arn arn:aws:iam::aws:po
```

```

WSParticipantRole:~/environment $
WSParticipantRole:~/environment $ aws iam attach-role-policy --role-name apprunner-admin-role --policy-arn arn:aws:iam::aws:policy/AmazonDynamoDBFullAccess
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $ aws iam attach-role-policy --role-name apprunner-admin-role --policy-arn arn:aws:iam::aws:policy/AmazonS3FullAccess
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $

```

Optional Step You can also download all those commands executed previously from Github using the command below that might easier for copy/paste commands from there directly:

```

WSParticipantRole:~/environment $
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $ wget https://github.com/mmasaud/Hackathon/blob/main/Citi%20Hackathon%20Scripts.txt
--2023-09-15 12:05:38-- https://github.com/mmasaud/Hackathon/blob/main/Citi%20Hackathon%20Scripts.txt
Resolving github.com (github.com)... 140.82.112.3
Connecting to github.com (github.com)|140.82.112.3|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 5630 (5.5K) [text/plain]
Saving to: 'Citi Hackathon Scripts.txt'

100%[=====] 5,630 --K/s in 0s

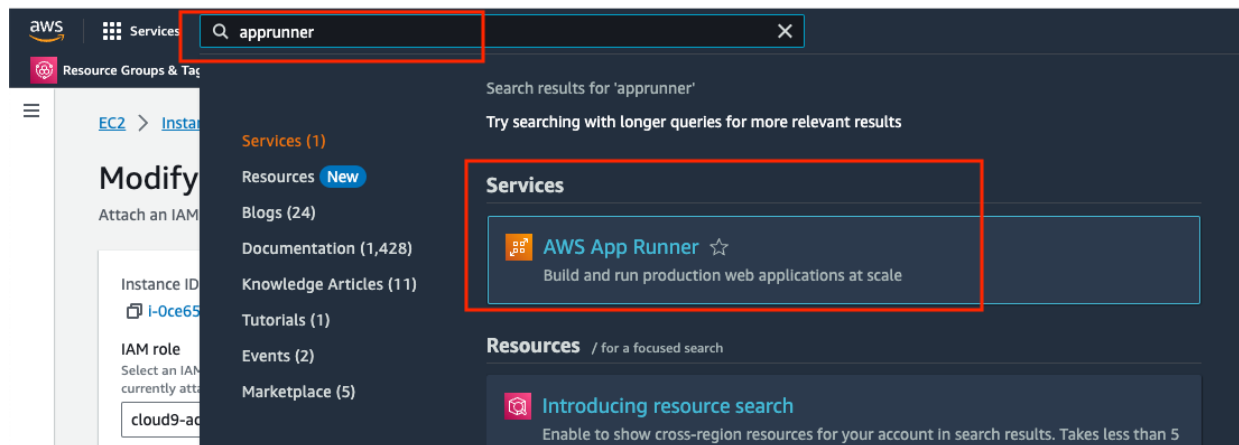
2023-09-15 12:05:38 (54.0 MB/s) - 'Citi Hackathon Scripts.txt' saved [5630/5630]

WSParticipantRole:~/environment $
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $ ls -ltrh
total 12K
-rw-r--r-- 1 ec2-user ec2-user 569 Aug 29 02:19 README.md
-rw-r--r-- 1 ec2-user ec2-user 5.5K Sep 15 12:05 Citi Hackathon Scripts.txt
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $
WSParticipantRole:~/environment $

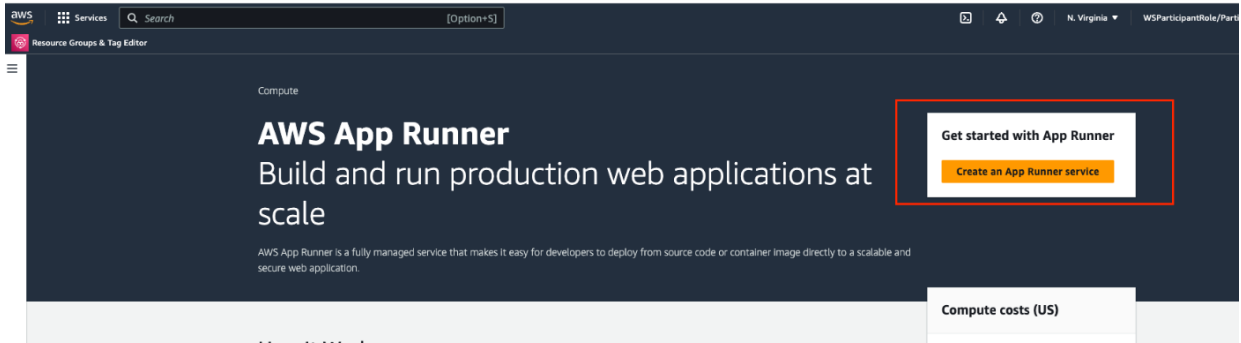
```

AppRunner Code execution :

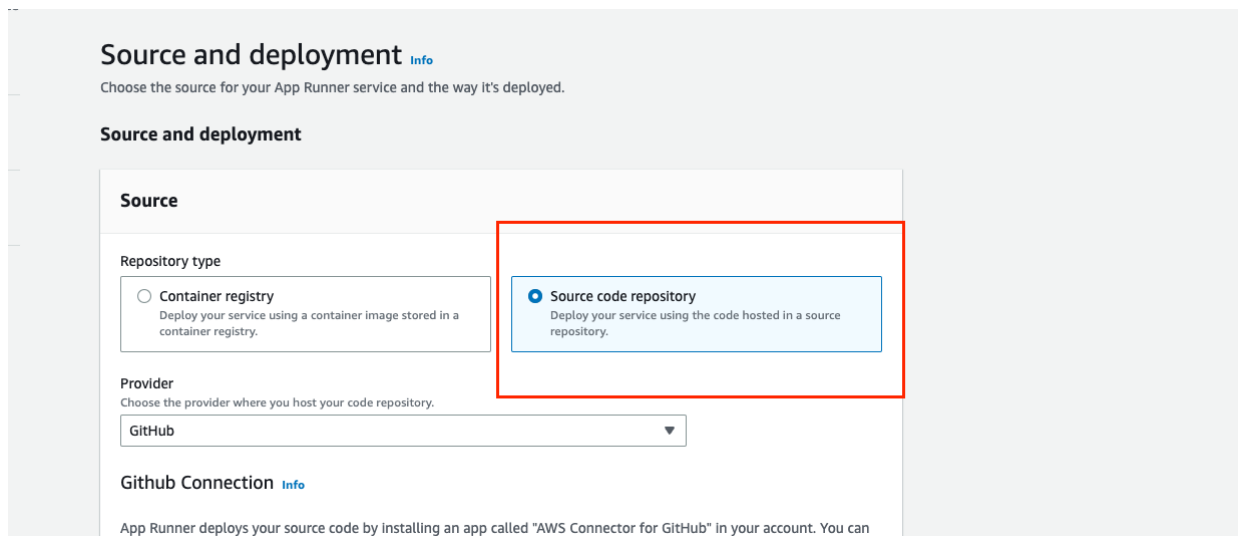
Back to AWS Console and look for Apprunner Service on the search bar :



Click on create a new Service :



Choose your repository type either you can use the source code directly from your Github account or you can use it from a container image which you uploaded already to Amazon ECR, for the sample code I'm using Source Code repository directly from Github as shown below



Then you need to give your Github credentials to Apprunner so it can access your repositories from Github as shown below:

Source and deployment

Source

Repository type

☐ Container registry
Deploy your service using a container image stored in a container registry.

☒ Source code repository
Deploy your service using the code hosted in a source repository.

Provider
Choose the provider where you host your code repository.

GitHub

Github Connection [Info](#)

App Runner deploys your source code by installing an app called "AWS Connector for GitHub" in your account. You can [install this app in your main](#) GitHub account or in a GitHub organization.


Add new

Repository


Branch

Confirm access

github.com/settings/installations/22266654



Confirm access

 Signed in as @m

Password

[Forgot password?](#)

Confirm

Tip: You are entering **sudo mode**. After you've performed a sudo-protected action, you'll only be asked to re-authenticate again after a few hours of inactivity.

Create a new connection
Create a new connection to deploy your service by using a GitHub or AWS Code Repository repository and using a GitHub or AWS Code Repository account. This is the first connection for future App Runner services.

Connection name
Name your connection something easy for you to reference if you want to use it again.

Provider
Choose from providers where a GitHub or AWS Code Repository is installed as a default or in a GitHub or AWS Code Repository account.

GitHub or Install another

Cancel Next

For this Sample application repository, please use the Github link on the reference section and fork the repository into your personal account so you can create the Apprunner Service using that forked Repo.

After Choosing the required repository and the right branch, you would need to choose the Deployment Setting as shown below, you have two option:

1- Manual:

Which will not reflect any changes happening on your repository unless you explicitly restart your service after you commit your code changes on Github.

2- Automatic:

This will instantly restart your running service once any changes committed on Github repository without any user interaction.

For more information please check out Deployment methods details. here:

<https://docs.aws.amazon.com/apprunner/latest/dg/manage-deploy.html#:~:text=Deployment-,methods,-App%20Runner%20provides>

Step 4
 Review and create

Repository type

☐ Container registry
Deploy your service using a container image stored in a container registry.

☒ Source code repository
Deploy your service using the code hosted in a source repository.

Provider
Choose the provider where you host your code repository.

GitHub

Github Connection [Info](#)

App Runner deploys your source code by installing an app called "AWS Connector for GitHub" in your account. You can install this app in your main GitHub account or in a GitHub organization.

test Add new

Repository
cmanikandan-apprunner-java-helloworld Refresh

Branch
main Refresh

Deployment settings

Deployment trigger

☒ Manual
Start each deployment yourself using the App Runner console or AWS CLI.

☐ Automatic
Every push to this branch that affects files in the specified Source directory deploys a new version of your service.

Cancel Next

Next step is to choose your Runtime configuration, Build Command, Start Command and port number based on the language

used in your code, I'm using Java for that Sample code so my configurations like below:

App Runner > Create service

Step 1
Source and deployment

Step 2
Configure build

Step 3
Configure service

Step 4
Review and create

Configure build info

Configure build

Build settings

Configuration file

☒ **Configure all settings here**
Specify all settings for your service here in the App Runner console.

☐ **Use a configuration file**
Let App Runner read your configuration from the `apprunner.yaml` file in the source directory of your code repository. App Runner defaults to the root directory if a Source directory wasn't specified in the previous step.

Runtime

Choose an App Runner runtime for your service.

Corretto 11

Build command

This command runs in the source directory of your repository when a new code version is deployed. Use it to install dependencies or compile your code. App Runner defaults to the root directory if a Source directory wasn't specified in the previous step.

mvn clean package

Start command

This command runs in the source directory of your service to start the service processes. Use this command to start a webserver for your service. The command can access environment variables that App Runner and you defined. App Runner defaults to the root directory if a Source directory wasn't specified in the previous step.

java -jar target/demo-0.0.1-SNAPSHOT.jar

Port

Your service uses this TCP port.

8080

Cancel Previous **Next**

Give a service name and make sure it would have enough CPU and Memory to start your application, one of the main features of using Apprunner that it takes care of compute resources scalability without any manual interaction from the user so even if you have spikes happening apprunner will scale the resources accordingly based on the number of HTTP/HTTPS request received

Resource Groups & Tag Editor

App Runner > Create service

Step 1
Source and deployment

Step 2
Configure build

Step 3
Configure service

Step 4
Review and create

Configure service Info

Configure service

Service settings

Service name

Hello-Java-App

Virtual CPU

1 vCPU

Virtual memory

2 GB

Environment variables — optional Info

Add environment variables in plain text or reference them from [Secrets Manager](#) and [SSM Parameter Store](#). Update IAM Policies using the IAM Policy template given below to securely reference secrets and configurations as environment variables.

No environment variables have been configured.

Add environment variable

You can add up to 50 more items.

► IAM policy templates

Scroll down on the same page to the Security Section and make sure to choose the IAM Security Role we created in Cloud9 command line for Apprunner to give it access to S3 and DynamoDB

▼ Security Info

Specify an Instance role and an AWS KMS encryption key

Permissions

Select an IAM role with permissions to AWS actions that your service code calls. To create a custom role, use the [IAM console](#).

Instance role

An instance role is auto-generated for every IAM role that is created for Amazon EC2 using the AWS Management Console. Choose an Instance role to apply the required IAM role to your application code. This grants access permissions to call AWS services.

apprunner-admin-role

↻

AWS KMS key

This key is used to encrypt the stored copies of your data.

☒ Use an AWS-owned key
A key that AWS owns and manages for you.

☐ Choose a different AWS KMS key
A key that you own or have permission to use.

Web Application Firewall Info

Activate WAF to define Web access control list (ACL) to protect against web exploits and bots. Learn more about [WAF and pricing](#).

☐ Activate

Finally, we will have to review the create the service if everything looks good as explained previously :

Services

Search

[Option+5]

Resource Groups & Tag Editor

Step 4

Review and create

Hello-Java-App

Virtual CPU

1 vCPU

Virtual memory

2 GB

Environment variables — optional

Add environment variables in plain text or reference them from [Secrets Manager](#) and [SSM Parameter Store](#). Update IAM Policies using the IAM Policy template given below to securely reference secrets and configurations as environment variables.

No environment variables have been configured.

Add environment variable

You can add up to 50 more items.

► IAM policy templates

► Auto scaling

Configure automatic scaling behavior.

► Health check

Configure load balancer health checks.

▼ Security

Specify an instance role and an AWS KMS encryption key

Permissions

Select an IAM role with permissions to AWS actions that your service code calls. To create a custom role, use the [IAM console](#).

Instance role

An instance role is auto-generated for every IAM role that is created for Amazon EC2 using the AWS Management Console. Choose an instance role to apply the required IAM role to your application code. This grants access permissions to call AWS services.

apprunner-admin-role

AWS KMS key

☒ Use an AWS-owned key

A key that AWS owns and manages for you.

☐ Choose a different AWS KMS key

A key that you own or have permission to use.

Web Application Firewall

Activate WAF to define Web access control list (ACL) to protect against web exploits and bots. Learn more about [WAF and pricing](#).

☐ Activate

► Networking

Configure the way your service communicates with other applications, services, and resources.

► Observability

Configure observability testing.

▼ Tags

Use tags to search and filter your resources, track your AWS costs, and control access permissions.

Tags — optional

A tag is a key-value pair that you assign to an AWS resource.

No tags associated with the resource.

Add new tag

You can add 50 more tags.

Cancel

Previous

Next

▼ Auto scaling

Concurrency

100

Minimum size

1

Maximum size

25

▼ Health check

Protocol

TCP

Path

Timeout

5 seconds

Interval

10 seconds

Unhealthy threshold

5 requests

Health threshold

1 requests

▼ Security

Permissions

Instance role

arn:aws:iam::282447775113:role/apprunner-admin-role

Data encryption

AWS KMS encryption key

AWS managed

▼ Networking

Outgoing

Outgoing network traffic

Public access

Incoming

Incoming network traffic

Public endpoint

▼ Observability

Observability

off

▼ Tags

Name

Value

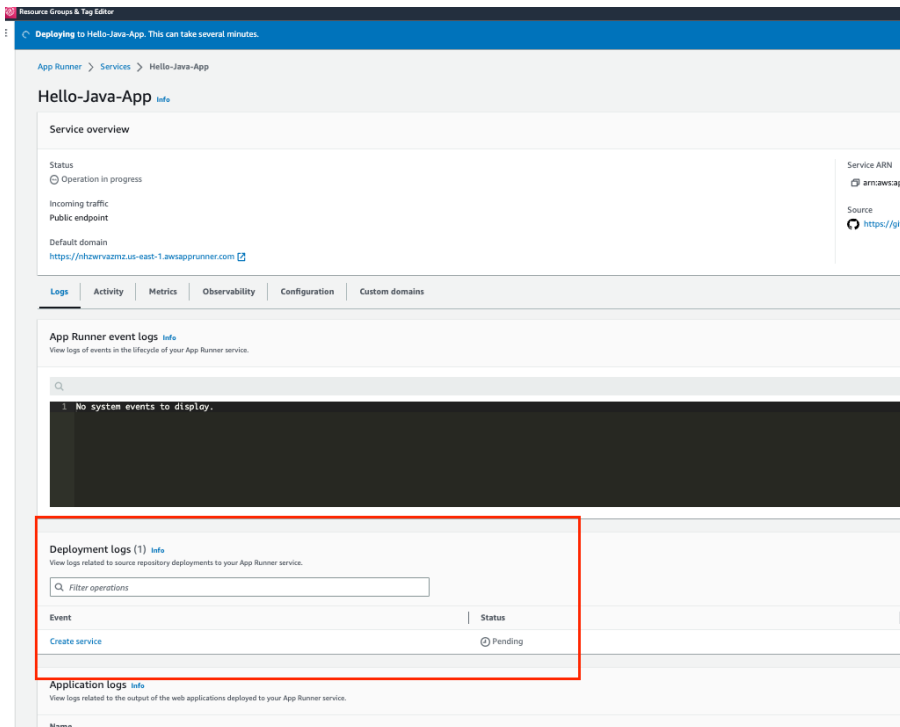
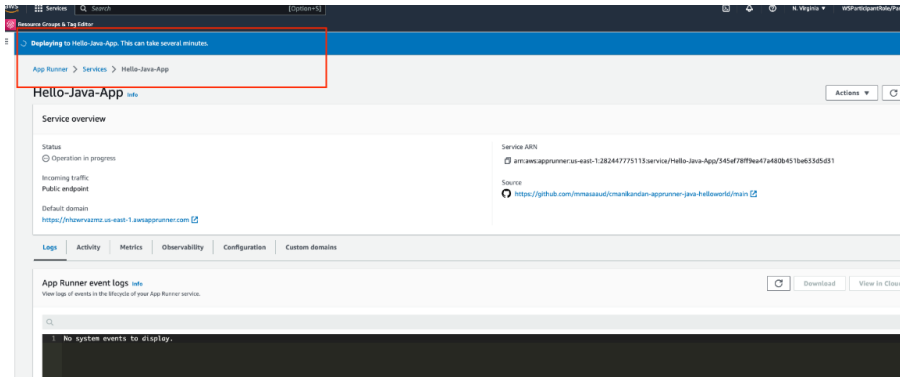
No tags have been configured.

Cancel

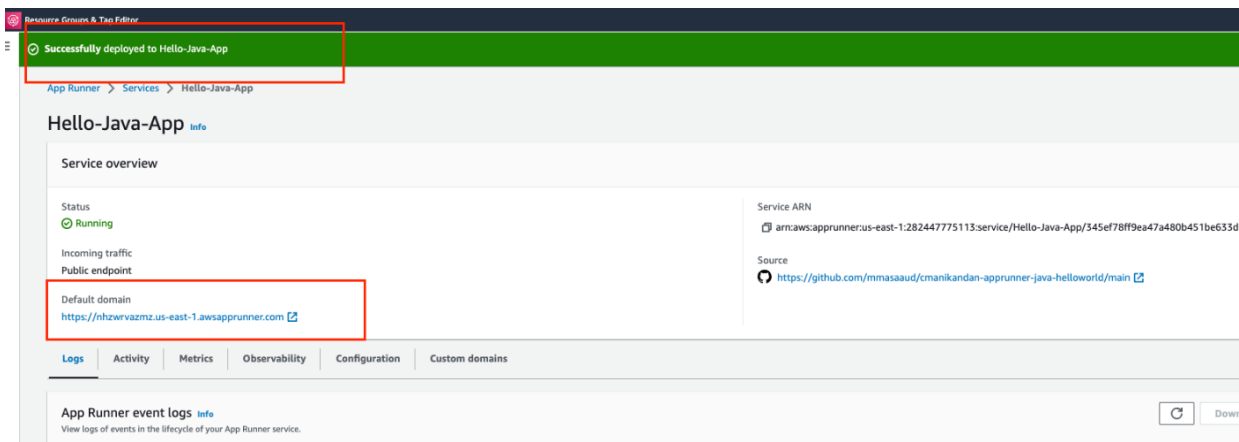
Previous

Create & deploy

This will take between 5-7 minute to create the service, you can also check the deployment log while deploying is taking place :



After few minutes, service created Successfully and you can access it from the default domain link, which will be created automatically as a part of service creation :



← → ↻ nhzwrvmz.us-east-1.awsapprunner.com

Hello Citi Team from AWS App Runner. The JDK version is 11.0.20.1

Important Links :

1- AWS SDK for Java :

<https://aws.amazon.com/sdk-for-java/>

2- Github link for Sample Java app :

<https://github.com/mmasaad/apprunner-java-helloworld>

3- DynamoDB document :

<https://us-east-2.console.aws.amazon.com/dynamodbv2/home?region=us-east-2#service>

4- S3 document :

<https://docs.aws.amazon.com/AmazonS3/latest/userguide/Welcome.html>