

Cloud Deployment with Automation

Introduction

This case study explores the concepts of AWS CodePipeline, EC2 and S3. In this we build a simple HTML WebApp using AWS CodeBuild, and automatically upload it to an S3 bucket to then automatically deploy it to an EC2 instance using CodeDeploy.

The tools and concepts used for this case study are :-

AWS CodeBuild

Key Features:

Build Automation: Fully managed service that compiles source code, runs tests, and produces software packages.

Custom Build Environments: Supports Docker images for custom build environments.

Scalability: Automatically scales to meet build demand.

Pay-as-you-go Pricing: You only pay for the compute resources you use.

Practical Uses:

Continuous Integration (CI) for automated testing and building of applications.

Integrating with other AWS services for a seamless development pipeline.

Building and packaging applications for deployment.

AWS CodePipeline

Key Features:

Continuous Delivery: Automates the software release process using defined workflows.

Integration with Other AWS Services: Works seamlessly with CodeBuild, CodeDeploy, and third-party tools.

Customizable Workflows: Easily define stages for building, testing, and deploying applications.

Practical Uses:

Automating the release process from code commit to deployment.

Creating pipelines for microservices or multi-environment setups.

Enabling rapid and reliable application delivery.

Amazon S3 (Simple Storage Service)

Key Features:

Scalable Storage: Virtually unlimited storage capacity.

Durability and Availability: Designed for 99.999999999% durability and high availability.

Security Features: Supports access control, encryption, and versioning.

Practical Uses:

Storing build artifacts and deployment packages.

Hosting static websites and serving assets for web applications.

Backup and archival storage.

Amazon EC2 (Elastic Compute Cloud)

Key Features:

Flexible Computing: Provides resizable compute capacity in the cloud.

Variety of Instance Types: Different instance types for various workloads.

Auto Scaling: Automatically adjusts capacity based on demand.

Practical Uses:

Hosting applications and services in a scalable manner.

Running batch processing and data analytics workloads.

Deploying web applications or back-end services.

AWS CodeDeploy

Key Features:

Automated Deployments: Automatically deploys applications to EC2, Lambda, or on-premises servers.

Blue/Green Deployments: Reduces downtime and risks during application updates.

Monitoring and Rollback: Monitors deployment status and can roll back if issues are detected.

Practical Uses:

Managing updates and deployments for applications running on EC2 instances.

Facilitating microservices deployments with minimal downtime.

Integrating with CI/CD pipelines to automate the deployment process.

Steps

Creating an S3 Bucket

General configuration

AWS Region
US East (N. Virginia) us-east-1

Bucket type [Info](#)

☒ **General purpose**
Recommended for most use cases and access patterns. General purpose buckets are the original S3 bucket type. They allow a mix of storage classes that redundantly store objects across multiple Availability Zones.

☐ **Directory**
Recommended for low-latency use cases. These buckets use only the S3 Express One Zone storage class, which provides faster processing of data within a single Availability Zone.

Bucket name [Info](#)

jai-61

Bucket name must be unique within the global namespace and follow the bucket naming rules. [See rules for bucket naming](#)

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

Choose bucket

Format: s3://bucket/prefix

Enabling static website hosting

Static website hosting

Use this bucket to host a website or redirect requests. [Learn more](#)

Edit

S3 static website hosting
Enabled

Hosting type
Bucket hosting

Bucket website endpoint

When you configure your bucket as a static website, the website is available at the AWS Region-specific website endpoint of the bucket. [Learn more](#)

<http://jai-61.s3-website-us-east-1.amazonaws.com>

Creating a Build project

Create build project

Project configuration

Project name

jai-61

A project name must be 2 to 255 characters. It can include the letters A-Z and a-z, the numbers 0-9, and the special characters - and _.

Public build access - optional
Public build access allows you to make the build results, including logs and artifacts, for this project available for the general public.

☒ Enable public build access

Public build access enabled
Your build results, including logs and artifacts, are accessible to the general public. Downloading logs and/or artifacts will increase your AWS costs. [Learn more](#)

Public build service role
The public build service role is used to provide read access to your logs and artifacts for public builds. You can let CodeBuild create a new role, or you can choose an existing role.

☐ **New service role**
Create a service role in your account

☒ **Existing service role**
Choose an existing service role from your account

Configuring source for CodeBuild project as a GitHub repository

Source 1 - Primary

Source provider

GitHub

Credential

☐ Default source credential
Use your account's default source credential to apply to all projects

☒ Custom source credential
Use a custom source credential to override your account's default settings

Credential type

☒ GitHub App
Connect project to GitHub using an AWS managed GitHub App

☐ OAuth app
Connect project to GitHub using an OAuth app

☐ Personal access token
Connect project to GitHub using a personal access token

Connection

You can [create a new GitHub connection](#) by using an AWS managed GitHub App

arn:aws:codeconnections:us-east-1:311141519473:connection/

X

↺

Repository

☒ Repository in my GitHub account

☐ Public repository

☐ GitHub scoped webhook

Repository

https://github.com/D15A-Jai-61/ADO

X

↺

Build project created

Developer Tools > CodeBuild > Build projects > jai-61

jai-61

Actions

Create trigger

Edit

Clone

Debug build

Start build with overrides

Start build

Configuration

Source provider GitHub	Primary repository D15A-Jai-61/ADO	Artifacts upload location jai-61	Service role arn:aws:iam::311141519473:role/service-role/codebuild-jai-61-service-role
Public builds Enabled	Public project URL Go to public project		

Build process succeeded and finished

Developer Tools > CodeBuild > Build projects > jai-61 > jai-61:36120006-8118-4486-aed8-419ac0fbb27f

jai-61:36120006-8118-4486-aed8-419ac0fbb27f

Stop build

Retry build

Build status

Status Succeeded	Initiator root	Build ARN arn:aws:codebuild:us-east-1:311141519473:build/jai-61:36120006-8118-4486-aed8-419ac0fbb27f	Resolved source version 7fb191ad0723de0e6cda3e22017822e476c1f5b9
Start time Oct 23, 2024 10:00 PM (UTC+5:30)	End time Oct 23, 2024 10:01 PM (UTC+5:30)	Build number 1	

Creating a custom Pipeline to deploy the built project

Choose creation option [Info](#)

Step 1 of 6

Creation options

Choose one of the following options to create your pipeline.

☐ **Create pipeline from template**
Create a pipeline from a pre-built template for common scenarios.

☒ **Build custom pipeline**
Build a pipeline from scratch to meet your specific needs.

[Cancel](#) [Next](#)

Pipeline settings

Action name
Choose a name for your action

CodeConnections

No more than 100 characters

Action provider

GitHub (Version 2)

Connection
Choose an existing connection that you have already configured, or create a new one and then return to this task.

New GitHub version 2 (app-based) action
To add a GitHub version 2 action in CodePipeline, you create a connection, which uses GitHub Apps to access your repository. Use the options below to choose an existing connection or create a new one. [Learn more](#)

arn:aws:codeconnections:us-east-1:311141519473:connection/c7034fe8-bf27-449f-87fe-071b1f14aac5

×

 or

Connect to GitHub

Ready to connect
Your GitHub connection is ready for use.

Repository name
Choose a repository in your GitHub account.

D15A-Jai-61/ADO

×

You can type or paste the group path to any project that the provided credentials can access. Use the format 'group/subgroup/project'.

Branch name
Choose a branch of the repository.

main

×

Pipeline successfully created

jai-61-pipeline

Notify

Edit

Stop execution

Clone pipeline

Release change

Pipeline type: V2

Execution mode: SUPERSEDED

Source

Succeeded

Pipeline execution ID: 392bab84-7974-41f5-a533-c3095bcf951a

Source

GitHub (Version 2)

Succeeded - Just now

7fb191ad

View details

7fb191ad

Source: after_install.sh in scripts folder

Launching an EC2 instance to deploy to and host the built WebPage

EC2

Launch an instance

Launch an instance

Info

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags

Info

Name

jai-61

Add additional tags

Amazon Linux as the Linux distribution of the EC2 instance

Quick Start

Amazon Linux

aws

macOS

Mac

Ubuntu

ubuntu

Windows

Microsoft

Red Hat

Red Hat

SUSE Li

SUSE

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

Amazon Linux 2 AMI (HVM) - Kernel 5.10, SSD Volume Type

Free tier eligible

ami-0ddc798b3f1a5117e (64-bit (x86)) / ami-05f16f3539e999b77 (64-bit (Arm))

Virtualization: hvm ENA enabled: true Root device type: ebs

Jai Talreja 61 D15A

Key pair for EC2 instance

Create key pair

Key pair name

Key pairs allow you to connect to your instance securely.

jai-61

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type

☒ RSA
RSA encrypted private and public key pair

☐ ED25519
ED25519 encrypted private and public key pair

Private key file format

☒ .pem
For use with OpenSSH

☐ .ppk
For use with PuTTY

⚠ When prompted, store the private key in a secure and accessible location on your computer. **You will need it later to connect to your instance.** [Learn more](#)

Cancel

Create key pair

Instance created

<input type="checkbox"/>	Name	Instance ID	Instance state	Instance type
<input type="checkbox"/>	jai-61	i-Ofbc53fd7d2f2d188	Running	t2.micro

Installed HTTP Daemon

```
Installed:
  httpd.x86_64 0:2.4.62-1.amzn2.0.2

Dependency Installed:
  apr.x86_64 0:1.7.2-1.amzn2
  generic-logos-httpd.noarch 0:18.0.0-4.amzn2
  mailcap.noarch 0:2.1.41-2.amzn2

Complete!
[root@ip-172-31-95-227 ~]#
```

Installed Ruby

```
Installed:
  ruby.x86_64 0:2.0.0.648-36.amzn2.0.12

Dependency Installed:
  ruby-irb.noarch 0:2.0.0.648-36.amzn2.0.12
  rubygem-io-console.x86_64 0:0.4.2-36.amzn2.0.12
  rubygem-rdoc.noarch 0:4.0.0-36.amzn2.0.12

Complete!
[root@ip-172-31-95-227 ~]#
```

HTTP Daemon status

```
[root@ip-172-31-95-227 ~]# systemctl status httpd
● httpd.service - The Apache HTTP Server
   Loaded: loaded (/usr/lib/systemd/system/httpd.service; enabled; vendor preset: disabled)
   Active: active (running) since Thu 2024-10-24 13:09:50 UTC; 39s ago
     Docs: man:httpd.service(8)
  Main PID: 3825 (httpd)
   Status: "Total requests: 0; Idle/Busy workers 100/0; Requests/sec: 0; Bytes served/sec: 0 B"
    CGroup: /system.slice/httpd.service
            └─3825 /usr/sbin/httpd -DFOREGROUND
              └─3826 /usr/sbin/httpd -DFOREGROUND
                └─3827 /usr/sbin/httpd -DFOREGROUND
                  └─3828 /usr/sbin/httpd -DFOREGROUND
                    └─3829 /usr/sbin/httpd -DFOREGROUND
                      └─3830 /usr/sbin/httpd -DFOREGROUND

Oct 24 13:09:50 ip-172-31-95-227.ec2.internal systemd[1]: Starting The Apache HTTP Server...
Oct 24 13:09:50 ip-172-31-95-227.ec2.internal systemd[1]: Started The Apache HTTP Server.
[root@ip-172-31-95-227 ~]#
```

Installing AWS CLI on my personal machine running Fedora Linux

```
jai@fedora:~$ curl "https://awscli.amazonaws.com/awscli-exe-linux-x86_64.zip" -o "awscliv2.zip"
unzip awscliv2.zip
sudo ./aws/install
```

AWS CLI configured with Deploy command error that would be fixed later on

```
jai@fedora:~$ aws configure
AWS Access Key ID [None]: AKIAUQ4L22BYUTTUFM5S
AWS Secret Access Key [None]: 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
Default region name [None]: us-east-1
Default output format [None]: json
jai@fedora:~$ aws deploy create-application --application-name jai-61

An error occurred (AccessDeniedException) when calling the CreateApplication operation: User: arn:aws:iam::311141519473:user/jai-61 is not authorized to perform : codedeploy:CreateApplication on resource: arn:aws:codedeploy:us-east-1:311141519473:application:jai-61 because no identity-based policy allows the codedeploy:CreateApplication action
jai@fedora:~$
```


Deploy command now working

```
jai@fedora:~
role/jai-61-ec2-code-deploy

An error occurred (AccessDeniedException) when calling the CreateDeploymentGroup
operation: Cross-account pass role is not allowed.
jai@fedora:~$ aws deploy create-deployment-group --application-name jai2 --depl
yment-group-name my-webapp-group --service-role-arn arn:aws:iam::3111-4151-9473:
role/jai-61-ec2-code-deploy

An error occurred (AccessDeniedException) when calling the CreateDeploymentGroup
operation: Cross-account pass role is not allowed.
jai@fedora:~$ aws deploy create-deployment-group --application-name jai2 --depl
yment-group-name my-webapp-group --service-role-arn arn:aws:iam::311141519473:ro
le/jai-61-ec2-code-deploy

An error occurred (InvalidRoleException) when calling the CreateDeploymentGroup
operation: AWS CodeDeploy does not have the permissions required to assume the r
ole arn:aws:iam::311141519473:role/jai-61-ec2-code-deploy.
jai@fedora:~$ aws deploy create-deployment-group --application-name jai2 --depl
yment-group-name my-webapp-group --service-role-arn arn:aws:iam::311141519473:ro
le/jai-61-ec2-code-deploy
{
  "deploymentGroupId": "213471b0-7c57-4932-8151-9b01fa4d4a45"
}
jai@fedora:~$
```

Amazon Deploy agent restarted on EC2 to reflect new changes in settings and permissions

```
[ec2-user@ip-172-31-95-227 ~]$ sudo service codedeploy-agent restart
Restarting codedeploy-agent:[ec2-user@ip-172-31-95-227 ~]$ sudo service codedeploy-agent status
The AWS CodeDeploy agent is running as PID 4097
[ec2-user@ip-172-31-95-227 ~]$
```

Deployment (final stage of pipeline) is successful

deployment
Succeeded

[Start rollback](#)

Pipeline execution ID: [9ac496af-e2f8-4bdd-9a4e-c4bd00a759a8](#)

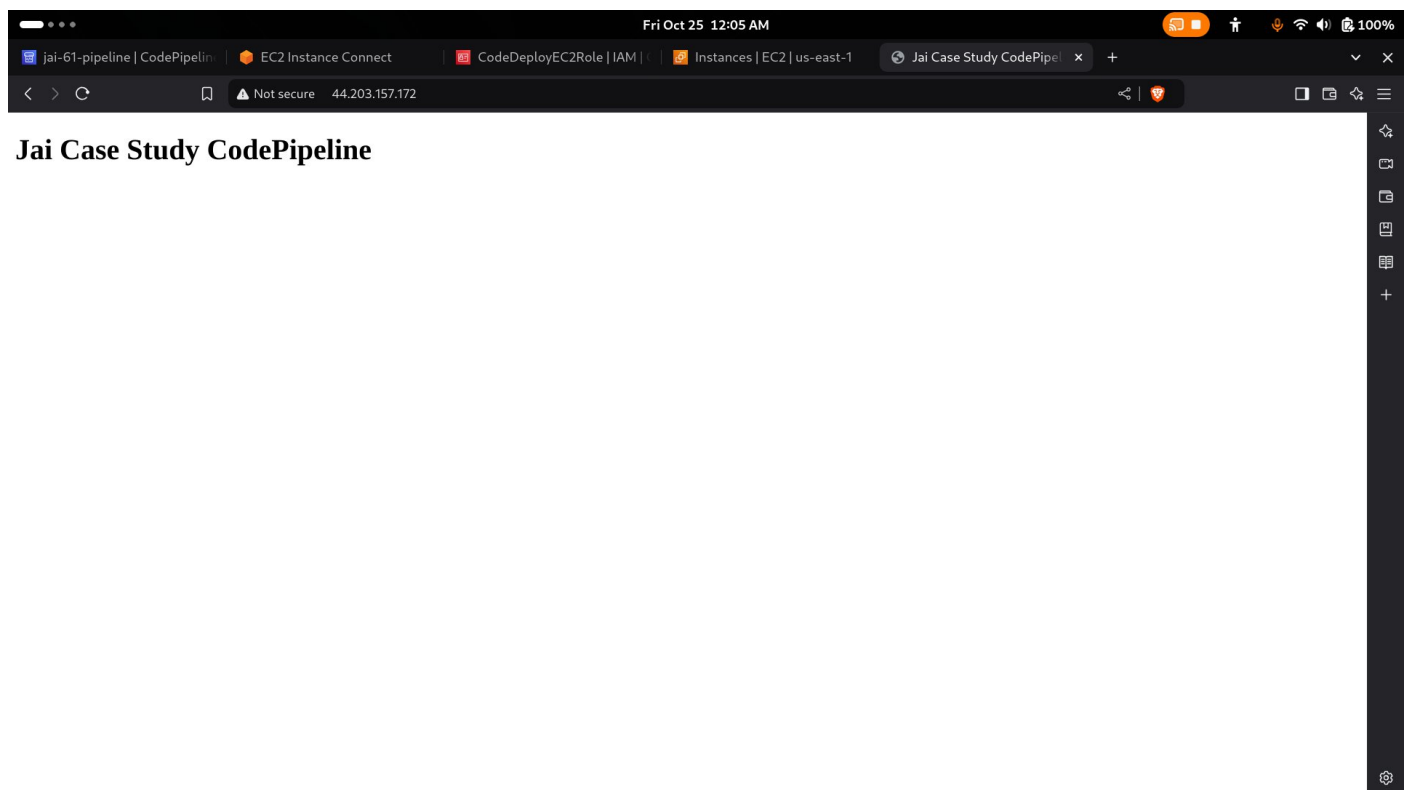
Deploy-to-AWS-Code-...
 [AWS CodeDeploy](#)

Succeeded - 1 minute ago
 [View details](#)

[7fb191ad](#)

 Source: after_install.sh in scripts folder

Connecting to the Public IP address of the EC2 instance via HTTP to see the result of the deployed WebApp



Learning

One of the things I have learned and has stuck with me is that the number of attached services besides permission policies in IAM roles does not refer to or define the number of services that can access it or are actually using it.

Other than that, this case study played an important role in widening the understanding of how Amazon Web Services works.

Conclusion

This case study was lengthy and tedious, some of the problems were caused due to the incomplete understanding of Amazon Web Services and how it works, with many possible variations in configurations for each and every step.

However, the case study was completed successfully, the issues faced were fixed in time, and the entire process is automated from start to end, starting from source file gathering to building, to deployment, and finally ending at hosting, everything is achieved without further user input.