



Build a CI/CD Pipeline with AWS



shilpa kale

Success
Congratulations! The pipeline nextwork-devops-cicd has been created.

Developer Tools > CodePipeline > Pipelines > nextwork-devops-cicd

nextwork-devops-cicd

Edit Stop execution Create trigger Clone pipeline Release change

Pipeline Executions Triggers Settings Tags Stage

Source Build Deploy

d575e141-184e-4ebb-8f66-781339565733 All actions succeeded.
GitHub (via GitHub App) 6 minutes ago
2fb84f05 Source: added code

d675e141-184e-4ebb-8f66-781339565733 All actions succeeded.
AWS CodeBuild 4 minutes ago
2fb84f05 Source: added code

d675e141-184e-4ebb-8f66-781339565733 All actions succeeded.
AWS CodeDeploy 3 minutes ago
2fb84f05 Source: added code

The screenshot shows the AWS CodePipeline interface with a green success banner at the top. Below it, the pipeline name 'nextwork-devops-cicd' is displayed along with standard navigation buttons like Edit, Stop execution, Create trigger, Clone pipeline, and Release change. The main area shows three stages: Source, Build, and Deploy, each with its own action and status. The Source stage shows a GitHub action that succeeded 6 minutes ago. The Build stage shows an AWS CodeBuild action that succeeded 4 minutes ago. The Deploy stage shows an AWS CodeDeploy action that succeeded 3 minutes ago. Each stage has a 'Details' button and a 'Logs' button.

Introducing Today's Project!

In this project, we will demonstrate how to use CodePipeline to set up a CI/CD pipeline! CodePipeline will automate the flow from GitHub all the way to CodeDeploy. We'll be able to push a change to code & see the updated webapp without going to CB,CD

Key tools and concepts

Services we used were CodePipeline, CodeDeploy, CodeBuild, CodeConnection, CodeArtifact, GitHub, S3, EC2, IAM, CloudFormation, VS Code. Key concepts we learnt include the different stages in CI/CD pipelines and webhooks.

Project reflection

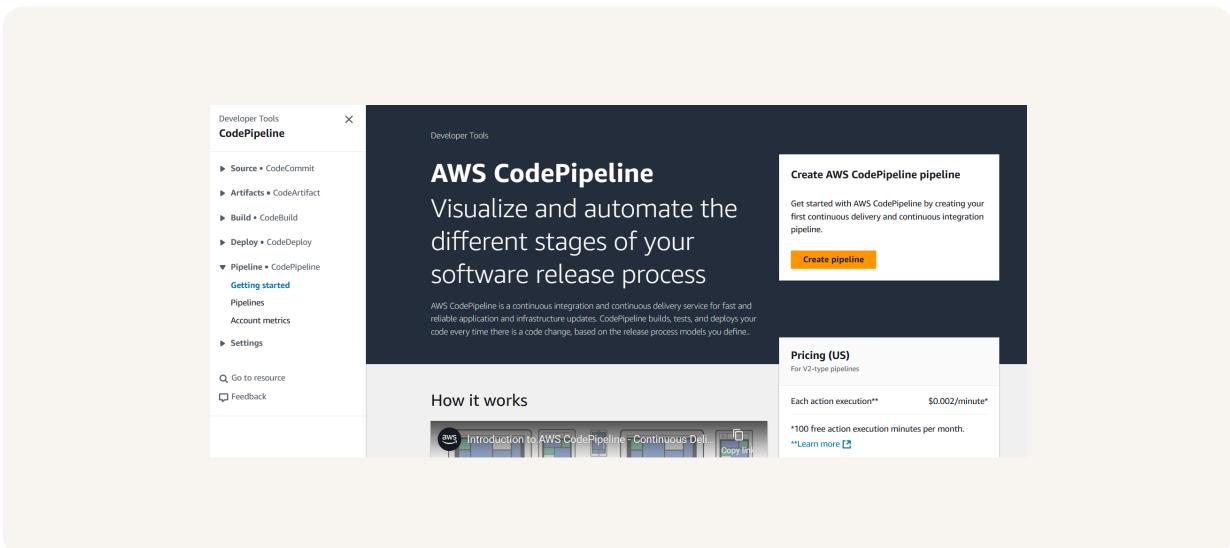
This project took us approx 3 hrs including documentation and troubleshooting time. The most challenging part was to push our code changes and resolve permissions errors. It was most rewarding to see our deployed web app update without doing ourselves

Starting a CI/CD Pipeline

AWS CodePipeline is AWS DevOps tool that helps us create a workflow that automatically moves code from GitHub all the way to CodeDeploy. This helps to make sure deployments are consistent and reliable.

CodePipeline offers different execution modes based on how we treat multiple run of the same pipeline. We chose 'Superseded' which means we will focus on running the latest push that we did and cancel older runs. Other options are Queued & Parallel.

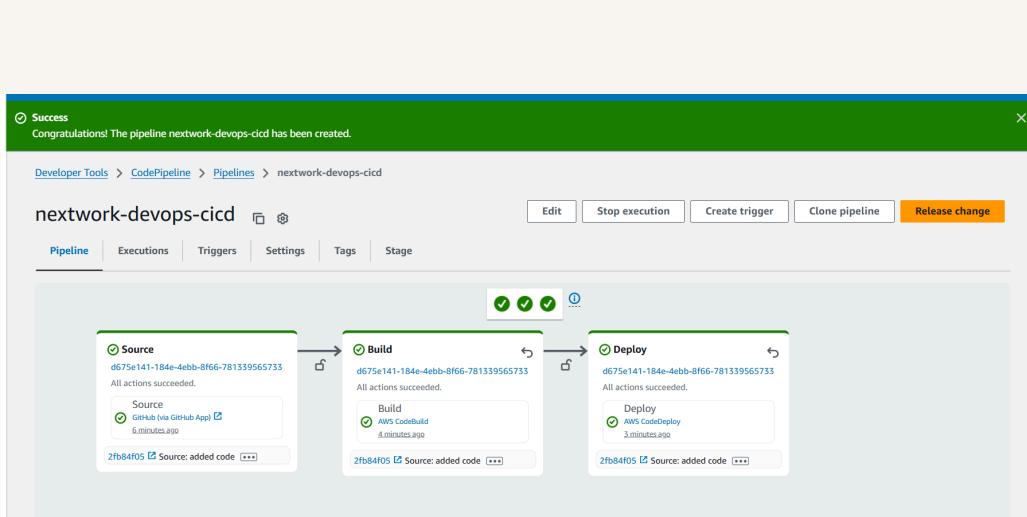
A service role gets created automatically during setup so that CodePipeline has access to the resources it needs to coordinate the diff stages i.e access to CodeConnection to connect to GitHub, access to CodeBuild and S3 to store artifacts.



CI/CD Stages

The three stages we've set up in my CI/CD pipeline are Source(source code which is GitHub), Build(building the web app using CodeBuild), Deploy(deploying changes of our web app using CodeDeploy).

CodePipeline organizes the three stages into a single diagram that showcases the flow of changes from source to deploy. In each stage, you can see more details on the pipeline execution that it belongs to & shortcuts to connectd servic



Source Stage

In the Source stage, the default branch tells CodePipeline exactly which version of our code we want to use in this workflow. In prod env. you can have many diff branches to represent many diff versions of your code.

The source stage is also where you enable webhook events, which are like notifications. Whenever you make change to the code, webhook events will detect the change and alert CodePipeline to trigger a new run.

The screenshot shows the 'Add source stage' configuration screen in the AWS CodePipeline console. The left sidebar lists steps: Step 3 (Add source stage), Step 4 (Add build stage), Step 5 (Add test stage), Step 6 (Add deploy stage), Step 7 (Review). The main area is titled 'Source provider' with the sub-instruction: 'This is where you stored your input artifacts for your pipeline. Choose the provider and then provide the connection details.' A dropdown menu is open, showing 'GitHub (via GitHub App)' as the selected option. Below it, there's a 'Connection' section with a search bar containing 'arn:aws:codeconnections:eu-central-1:619071342657:connection/6i' and a 'Connect to GitHub' button. The 'Repository name' field contains 'KaleShilpa/nextwork-web-project'. A note below says, 'You can type or paste the group path to any project that the provided credentials can access. Use the format "group/subgroup/project".' The 'Default branch' field contains 'master'. The 'Output artifact format' section has two options: 'CodePipeline default' (selected) and 'Full clone'. A note for 'Full clone' states: 'AWS CodePipeline passes metadata about the repository that allows subsequent actions to do a full Git clone. Only supported for AWS CodeBuild actions. Learn more' with a link icon. At the bottom, there's a checked checkbox for 'Enable automatic retry on stage failure'.

Build Stage

The Build stage sets up how we build our web app and make it ready for deployment. We configured CodeBuild to be our build provider and to use the input artifacts that was outputted by source stage. The input artifact is the zip file.

The screenshot shows the 'Add build stage' configuration interface. On the left, a sidebar lists steps: Step 5 (Add test stage), Step 6 (Add deploy stage), Step 7 (Review). The main area has tabs for 'Commands' (selected) and 'Other build providers'. A dropdown shows 'AWS CodeBuild' selected. The 'Project name' field contains 'nextwork-devops-cicd'. Below it is a checkbox for 'Define buildspec override - optional' which is unchecked. The 'Environment variables - optional' section is collapsed. The 'Build type' section shows 'Single build' (selected) and 'Batch build' (disabled). The 'Region' dropdown is set to 'Europe (Frankfurt)'. The 'Input artifacts' section is collapsed.

Deploy Stage

The Deploy stage is where we setup CodeDeploy to be our deployment provider. It takes the BuildArtifact from CodeBuild, application and deployment settings that we set in our deployment group.

The screenshot shows the AWS CodePipeline console interface during the configuration of a Deploy stage. On the left, a vertical sidebar lists steps: Step 3 (Add source stage), Step 4 (Add build stage), Step 5 (Add test stage), Step 6 (Add deploy stage), and Step 7 (Review). Step 6 is currently selected. The main panel displays the Deploy stage configuration:

- Deploy provider:** AWS CodeDeploy
- Region:** Europe (Frankfurt)
- Input artifacts:** BuildArtifact (Defined by: Build)
- Application name:** nextwork-devops-cicd
- Deployment group:** nextwork-devops-cicd-deployment-group
- Configure automatic rollback on stage failure:**
- Enable automatic retry on stage failure:**

Success!

Since our CI/CD pipeline gets triggered by code changes and webhooks, we tested our pipeline by updating our code. We added a new line to index.jsp and pushed those changes too.

The moment we pushed the code change, CodePipeline responded immediately by triggering a new build and deploy stage. The commit message under each stage reflects the latest code change that they are using.

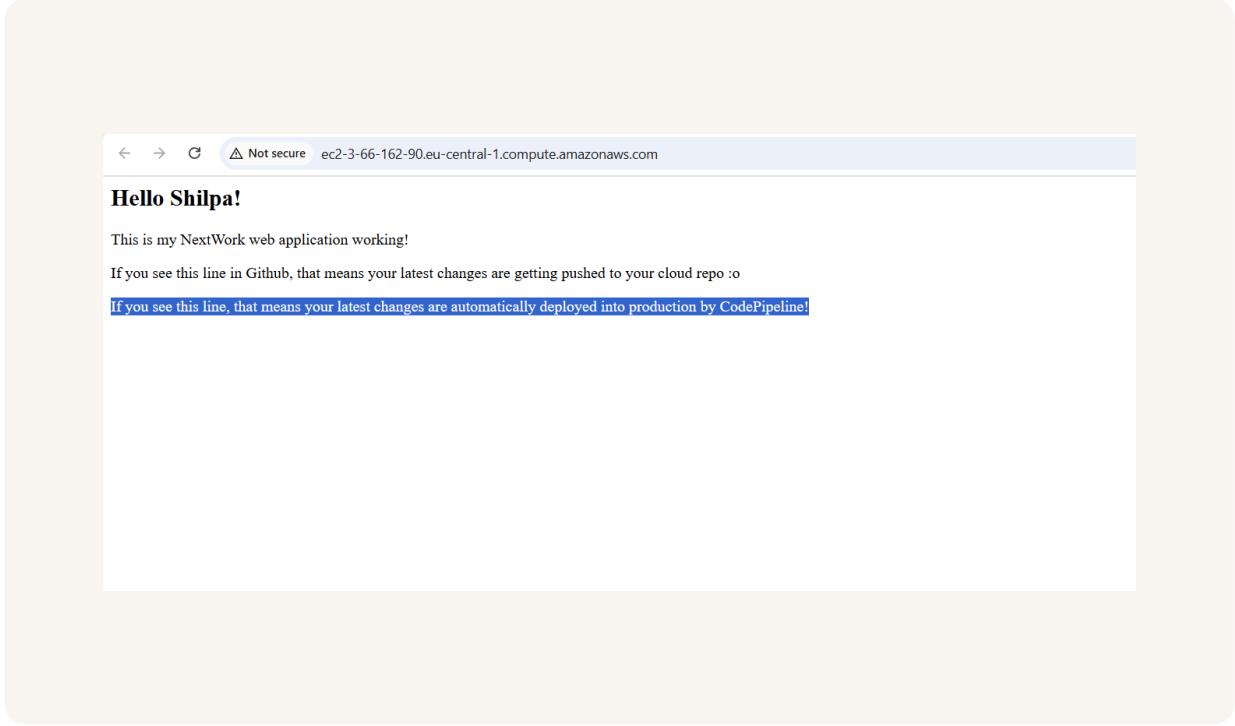
Once our pipeline executed successfully, we checked our live webapp and can confirm that it updated without us having to manually rebuild the project in CodeBuild and redeploy the code in CodeDeploy. The changes went live straight away to production.



S

shilpa kale
NextWork Student

nextwork.org



Not secure ec2-3-66-162-90.eu-central-1.compute.amazonaws.com

Hello Shilpa!

This is my NextWork web application working!

If you see this line in Github, that means your latest changes are getting pushed to your cloud repo :o

If you see this line, that means your latest changes are automatically deployed into production by CodePipeline!



nextwork.org

The place to learn & showcase your skills

Check out nextwork.org for more projects

