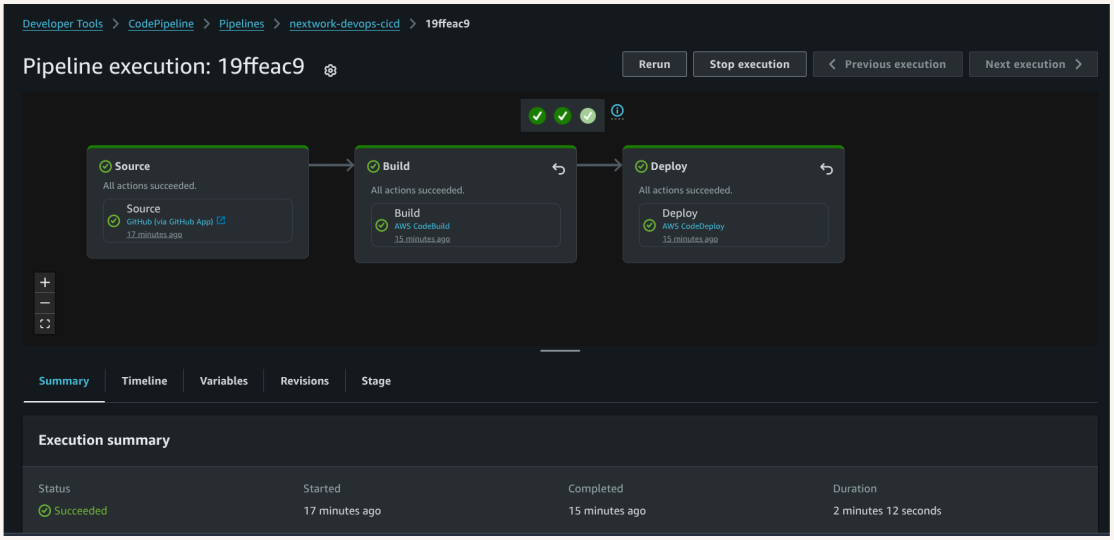
nextwork.org

Build a CI/CD Pipeline with AWS

S Shadrack Kalukwo



S
Shadrack Kalukwo
NextWork Student

Introducing Today's Project!

In this project, I will demonstrate how to use CodePipeline to setup a CICD pipeline, CodePipeline will automate the flow from github to CodeDeploy. By the end of this project we'll be able to push a change and see the updated live in production.

Key tools and concepts

Services I used were CodeDeploy, CodeBuild, CodePipeline, Github, VSCode, IAM, Chatgpt. Key concepts I learnt include; Rollback, disaster recovery, automation, deployment, cicd,

Project reflection

This project took me approximately 4 hoursThe most challenging part was solving erros to deal witth CodeBuild and CodeDeploy. It was most rewarding to see the changes deployed live into production by CodeDeploy.

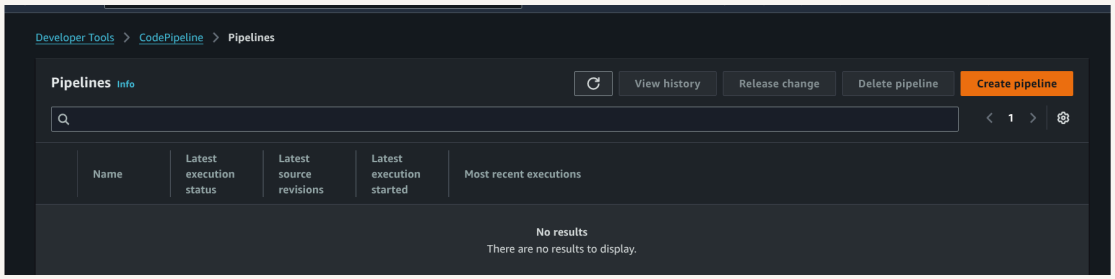
S
Shadrack Kalukwo
NextWork Student

Starting a CI/CD Pipeline

AWS CodePipeline is a AWS DevOps tool that helps us create a workflow that automatically moves code from Github our source code repo, all the way to CodeDeploy (Deployment Tool). Helps to make sure that deployments are consistent and reliable.

CodePipeline offers different execution modes based on hotreat multiple runs of the same pipeline. I chose superceeded which means we will focus on running the latest run that we start and cancel older ones. Other options include queue and parallel.

A service role gets created automatically during setup so that CodePipeline can have permision to access services like; CodeConnections, CodeBuild and CodeDeploy.

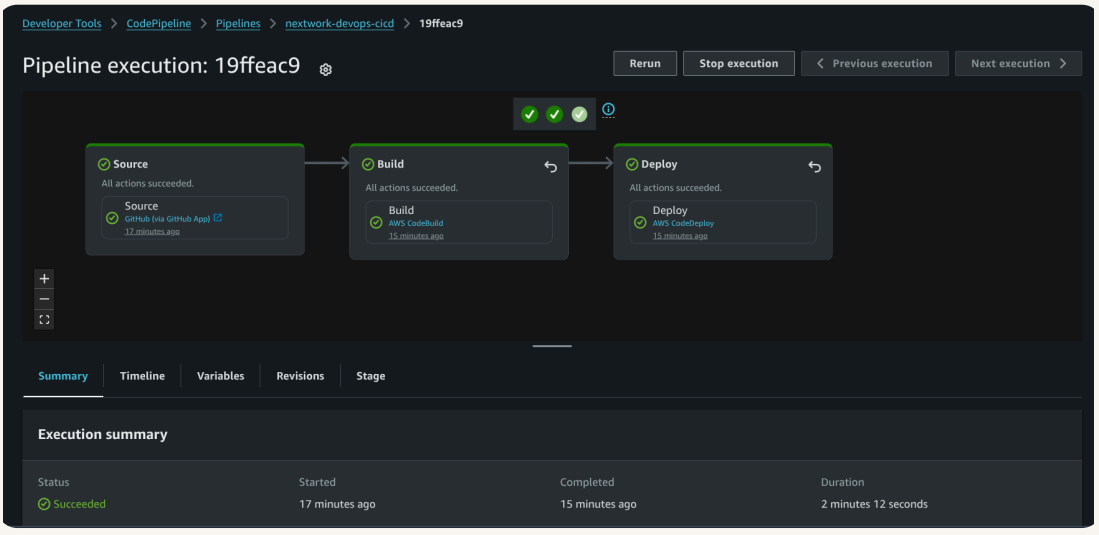


Shadrack Kalukwo
NextWork Student

CI/CD Stages

The three stages I've set up in my CI/CD pipeline are source; source code for our web project in githib. Builstage; building the web app using CodeBuild. Deploy stage; deploy changes 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 and the shortcuts to the connected service.

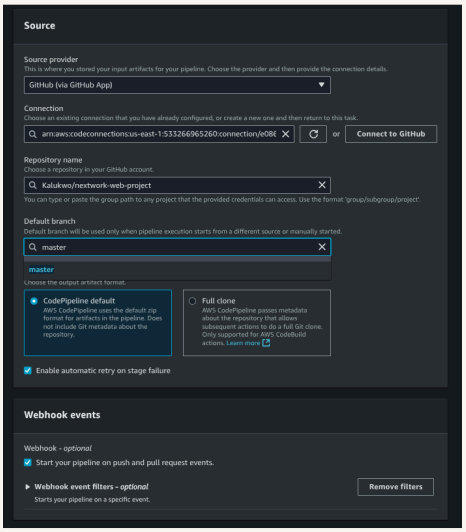


Shadrack Kalukwo
NextWork Student

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 production environments we can have many different branches to represent many different version of your code.

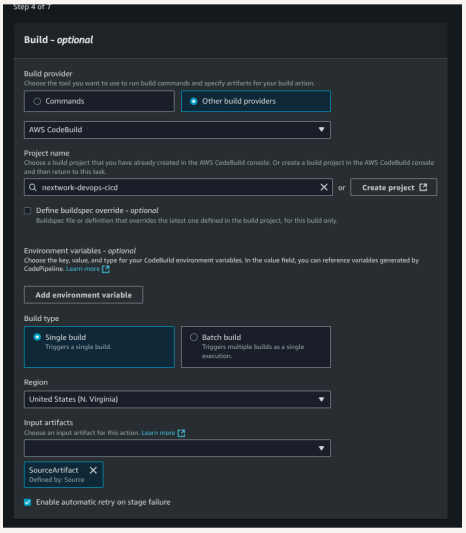
The source stage is also where you enable webhook events, which are like notifications whenever you make a change to the source code webhook event will detect the change and alert CodePipeline to trigger a new run.



Shadrack Kalukwo
NextWork Student

Build Stage

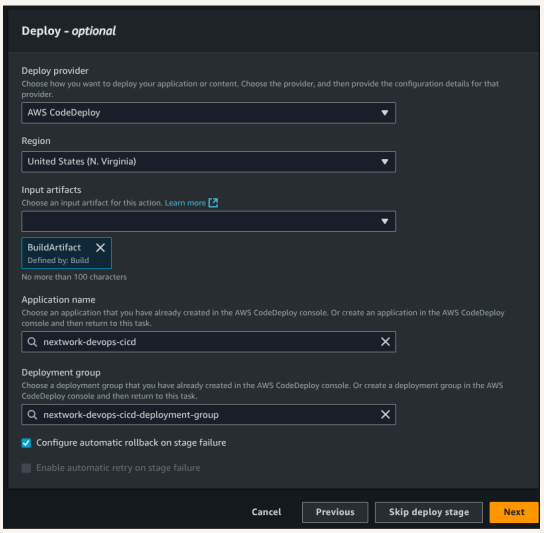
The Build stage sets up how we'll build our web app and make it ready for deployment. I configured CodeBuild to be my build provider and to use the input artifatcs from the source stage. The input artifact is the compressed code from github.



Shadrack Kalukwo
NextWork Student

Deploy Stage

The Deploy stage is where we setup CodeDeploy to be our deploy provider. It takes the build artifact from CodeBuild and the application deployment settings that we've defined in our deployment group.



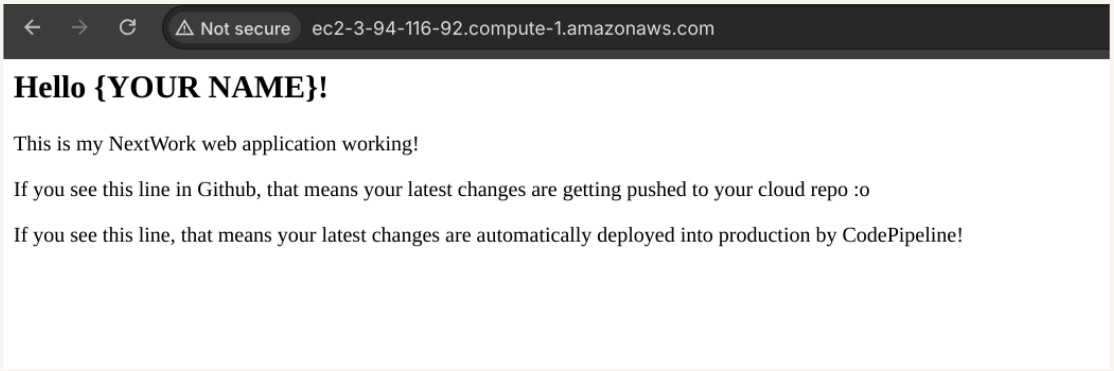
S
Shadrack Kalukwo
NextWork Student

Success!

Since my CI/CD pipeline gets triggered by code changes and webhook events, we tested our pipeline by updating our code, this means I added a new line to my web app'sindex.jsp file and pushed those changes too.

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

Once my pipeline executed successfully, I checked our live web app and confirm that it without having us to manually rebuild our project in CodeBuild and redeploy in CodeDeploy, the changes went live into production straight away.



NextWork.org

Everyone should be in a job they love.

Check out nextwork.org for more projects