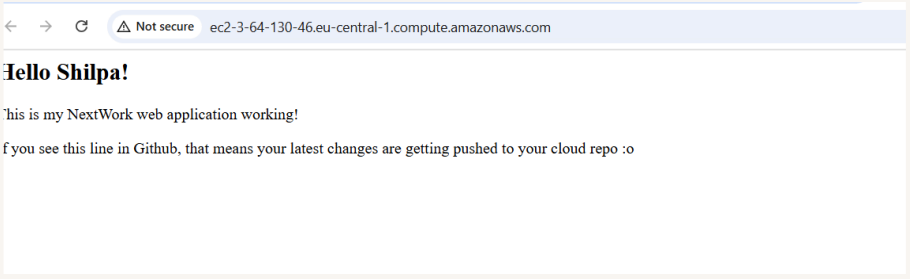
nextwork.org

Deploy a Web App with CodeDeploy

shilpa kale



Introducing Today's Project!

In this project, we'll demonstrate how to use CodeDeploy to deploy a web app. We're doing this project to learn about how deployment works, and how it can be automated using a combination of CodeDeploy and deployment scripts.

Key tools and concepts

Services we used were CodeDeploy, CodeBuild, CodeArtifact, IAM, EC2, S3, CloudFormation, CodeConnection, VS Code, GitHub. Key concepts we learnt include deployment, deployment groups, scripts, appsec and the importance of rebuilding your project.

Project reflection

This project took us approx 3 hrs including troubleshooting and documentation time. The most challenging part was understanding the seperation of the development instance from deployment instance. It was most rewarding to see the deployed web app.

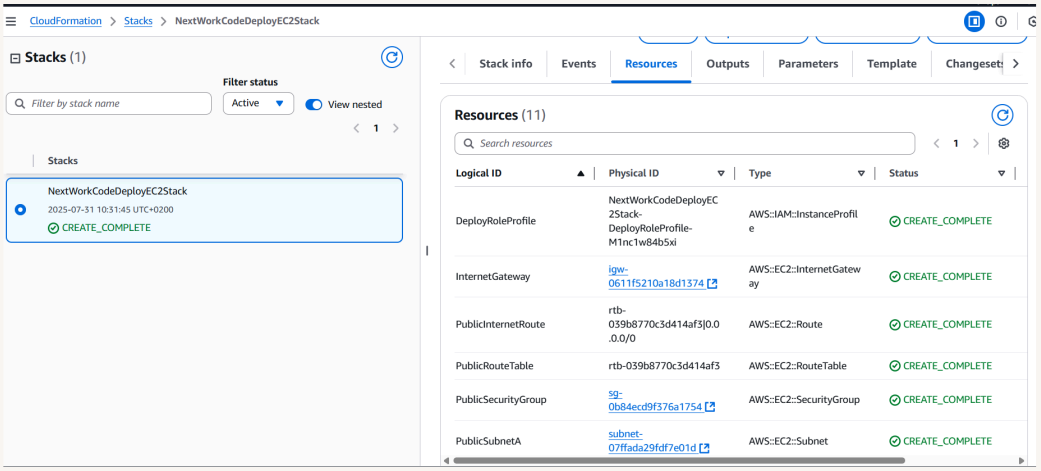
This project is part five of a series of DevOps projects where we're building a CI/CD pipeline! we'll be working on the next project tomorrow.

Deployment Environment

To set up for CodeDeploy, we launched an EC2 instance and VPC because they will make up our production env. We need to separate our development and production env. so that the code we are writing don't get shown to users until we are ready.

Instead of launching these resources manually, we used CloudFormation. When we need to delete these resources created by CloudFormation template we can simply delete the entire stack. This will automatically delete the entire resources that are insid

Other resources created in this template include networking resources like VPC, internet gateways, routables and subnets. They're also in the template because production env. has specific req around traffic that it should enable and blockout.


CloudFormation > Stacks > NextWorkCodeDeployEC2Stack
Stacks (1)
Filter status
Filter by stack name
Active
View nested
Stacks
NextWorkCodeDeployEC2Stack
2025-07-31 10:31:45 UTC+0200
CREATE_COMPLETE
Stack info
Events
Resources
Outputs
Parameters
Template
Changesets
Resources (11)
Search resources
Logical ID
Physical ID
Type
Status
DeployRoleProfile
NextWorkCodeDeployEC2Stack-DeployRoleProfile-M1nc1w84b5xi
AWS::IAM::InstanceProfile
CREATE_COMPLETE
InternetGateway
igw-0611f5210a18d1374
AWS::EC2::InternetGateway
CREATE_COMPLETE
PublicInternetRoute
rtb-039b8770c3d414af3|0.0.0.0/0
AWS::EC2::Route
CREATE_COMPLETE
PublicRouteTable
rtb-039b8770c3d414af3
AWS::EC2::RouteTable
CREATE_COMPLETE
PublicSecurityGroup
sg-0b84ecd9f376a1754
AWS::EC2::SecurityGroup
CREATE_COMPLETE
PublicSubnetA
subnet-07ffada29fdf7e01d
AWS::EC2::Subnet
CREATE_COMPLETE

Deployment Scripts

Scripts are mini programes that helps us automate the running of commands. To set up CodeDeploy, we also wrote scripts to automate deployment commands i.e. these are the commands that our deployment EC2 instance needs to run in order to host our appl

The 'install_dependencies.sh' will help our EC2 instance to install all the dependency it needs to host the web app like tomcat.

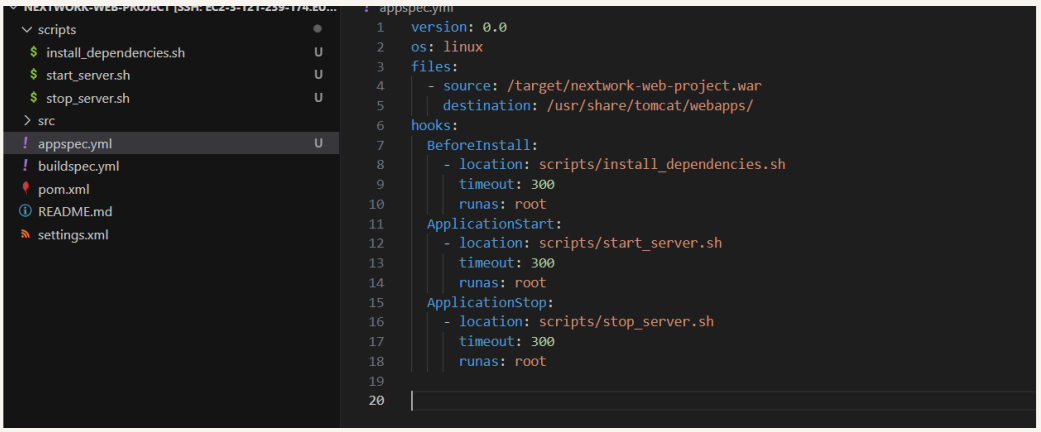
The start_server.sh will start the two servers that we're using in our EC2 instance- one, it start tomcat responsible for running our web app and two, it starts up apache- web server responsible for handling web traffic and passing request to tomcat.

The stop_server.sh is a script that stops apache and tomcat server when they are no longer needed to serve a webapp to a user.

appspec.yml

Then, we wrote an appspec.yml file to give CodeDeploy the instructions for deploying our web app. The key sections in appspec.yml are BeforeInstall(i.e. install_dependencies.sh), ApplicationStart(i.e. start_server.sh) and ApplicationStop.

We also updated buildspec.yml to tell CodeBuild that it should also package the new appspec.yml and set of scripts that we generated inside the build artifact i.e. the compressed war file.

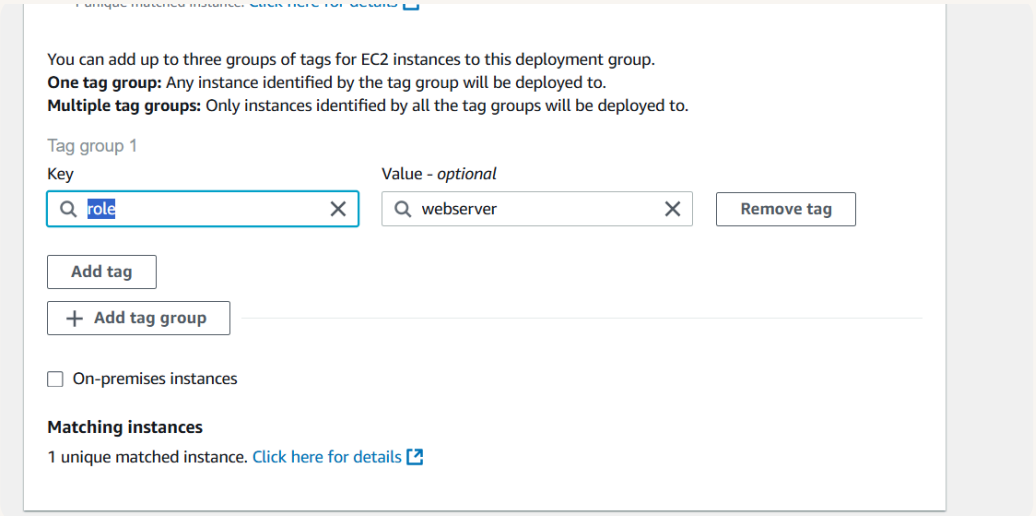


Setting Up CodeDeploy

A deployment group is a group of EC2 instances that you can deploy to and also a collection of settings that determine how you want to deploy. A CodeDeploy application is simply a folder that holds together all deployment groups for the same app.

To set up a deployment group we also need to create an IAM role to give CodeDeploy the permission to access the EC2 instances that it needs to coordinate. Otherwise CodeDeploy doesn't have access to EC2 instances.

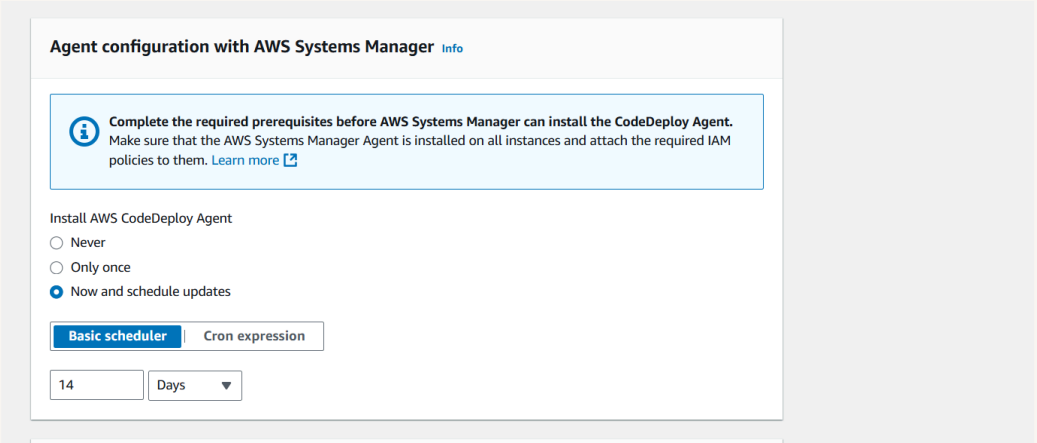
Tags are helpful for identifying the insatances that will be deploying the web app. We used the tag 'role=webserver' to automatically match the EC2 instance we deployed with our CloudFormation template. In future we can simply tag the new instance.


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
role
Value - optional
webserver
Remove tag
Add tag
Add tag group
On-premises instances
Matching instances
1 unique matched instance. Click here for details

Deployment configurations

Another key settings is the deployment configuration, which affects how quickly and risky we are deploying the webapp. We used CodeDeployDefault.AllAtOnce, so any changes that CodeDeploy deploys affects all instances in the deployment group at once,

In order to connect deployment instance with CodeDeploy, a CodeDeploy Agent is set up to receive instructions from CodeDeploy and make sure that the commands in appspec.yml are run.

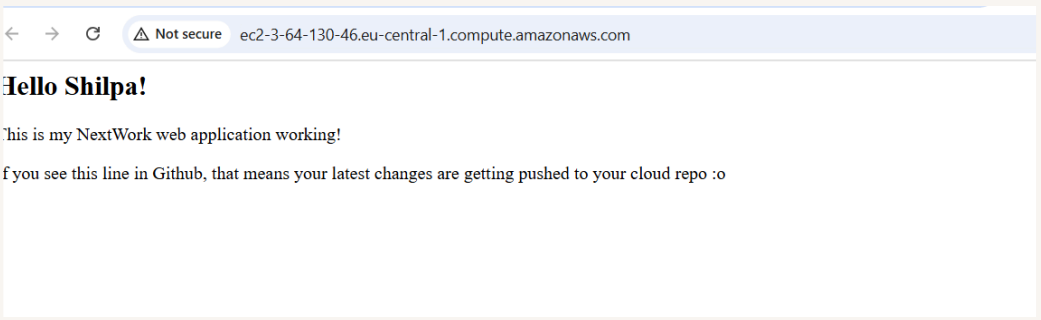


Success!

A CodeDeploy deployment is a specific update to our application that we are deploying to our users. The difference to a deployment group is that group is like a settings file and deployment itself is like specific update that we make using that file.

We had to configure a revision location, which means where our web apps war file is located. Our revision location was the S3 bucket that we created and linked with CodeBuild i.e build artifacts automatically go into that bucket.

To check that the deployment was a success, we visited the ipv4 DNS address of our EC2 instance. We saw a live web app that's working and serving our web app code to users.


Not secure
ec2-3-64-130-46.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

The place to learn & showcase your skills

Check out nextwork.org for more projects