Infrastructure as Code: Deploy CloudFormation Stack

We'll cover the following

Objective

- Recreate our infrastructure using CloudFormation.

Steps

- Deploy the CloudFormation Stack.

Deploying

Now it's time to deploy our infrastructure. Let's run the `deploy-infra.sh` command. We can check the status of our stack from the CloudFormation console. The events tab shows which resources are being created, modified, or destroyed.

When successful, the script should show us the URL for reaching our application.

```
./deploy-infra.sh

Waiting for changeset to be created..
Waiting for stack create/update to complete
Successfully created/updated stack - awsbootstrap
[
    "http://ec2-35-174-3-173.compute-1.amazonaws.com:8080"
]
```

terminal

And now we can test that our application is up and running with `curl`.

```
curl ec2-35-174-3-173.compute-1.amazonaws.com:8080
Hello World
```

terminal

Now we can commit our infrastructure code to GitHub to checkpoint our progress.

```
git add deploy-infra.sh main.yml
git commit -m "Create infrastructure via CloudFormation"
git push
```

terminal

Github access token

> **NOTE:** Before we run the code, we will require a Github access token so you don't need to put in your *username* and *password* whenever you access Github through the terminal widget. To generate an access token, go to https://github.com/settings/tokens/new and click *Generate new token*. Give it *repo* and *admin:repo_hook* permissions, and click *Generate token*.

> **NOTE:** You just have to put your Github token **ONCE** in the terminal widget, it will persist that information in all the other terminal widgets automatically.

Run and push the code

All of the files have already been placed. You just have to run the code.

> **NOTE:** It will automatically push the code to your Github too.

> **NOTE:** In `deploy-infra.sh` file, on **Line #1** (which is also highlighted in the code), another file by the name of `aws_credentials` is getting executed to set your AWS credentials but you can't see it. Similarly in `github.sh`, on **Line #1**, `github_credentials.sh` is getting executed to set your Github credentials. We have hidden these two files so your main focus is on the code, not to set up the credentials.

This code requires the following API keys to execute:

username	
AWS_ACCESS_KE...	Not Specified...
AWS_SECRET_AC...	Not Specified...
AWS_REGION	us-east-1
Github_Token	Not Specified...

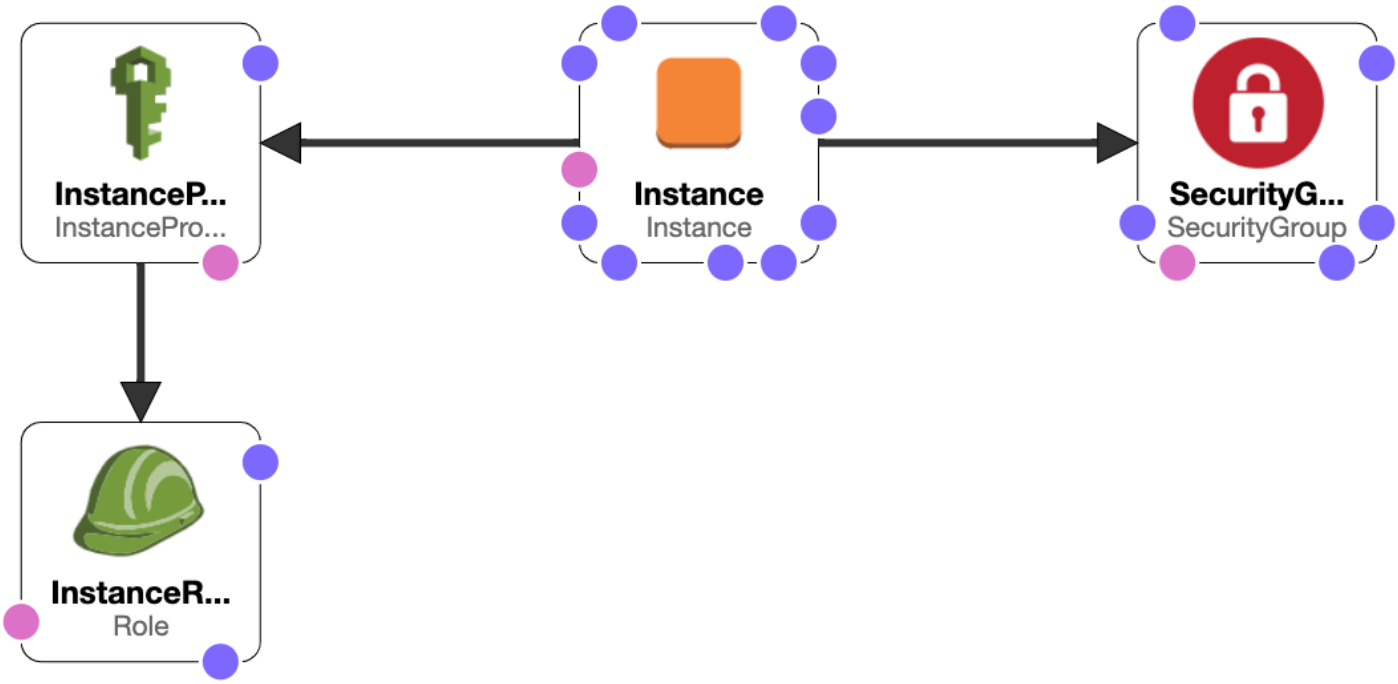
```
git clone https://github.com/{{username}}/aws-bootstrap.git
cp /usercode/infrastructureAsCode/main.yml aws-bootstrap/
cp /usercode/infrastructureAsCode/deploy-infra.sh aws-bootstrap/
cd aws-bootstrap
git config --global user.email "you@example.com"
git config --global user.name "Your Name"
git remote set-url origin https://{{username}}:{{Github_Token}}@github.com/{{username}}/aws-bootst
```

We now have our application running in the cloud, with its basic infrastructure managed through code. However, if we make a change to our application, our EC2 instance won't be updated.

> 🔍 Only one action at a time can be in progress for a given CloudFormation stack. If you get an error that says your stack "is in [?] state and cannot be updated", then wait until the stack has finished its current update and try again.

> ⚠ If there is an error with the creation of your stack, you may get a message saying that your stack "is in ROLLBACK_COMPLETE state and cannot be updated." When this happens, you will not be able to deploy again. CloudFormation does this to give you a chance to inspect the error that caused the deployment to fail. Once you've addressed the issue, you'll need to delete the stack and redeploy it.

In order to get a pictorial view of our developed CloudFormation stack so far, below is the design view which shows the resources we created and their relationships.



Infrastructure as Code

In the next lesson, we will make our instance receive a new version of our application automatically as soon as a change is pushed to GitHub.