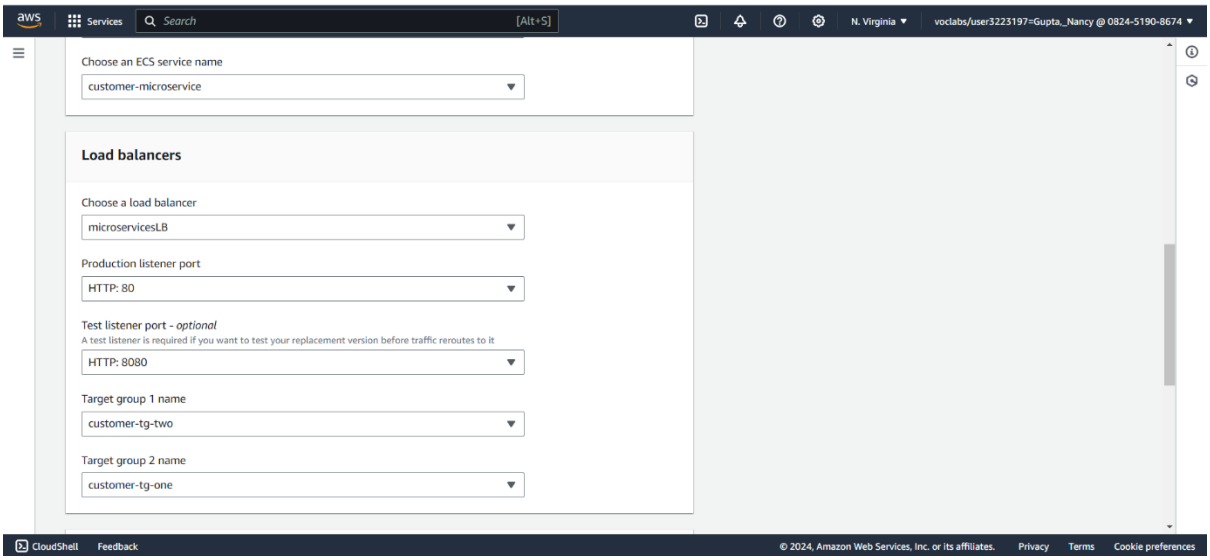
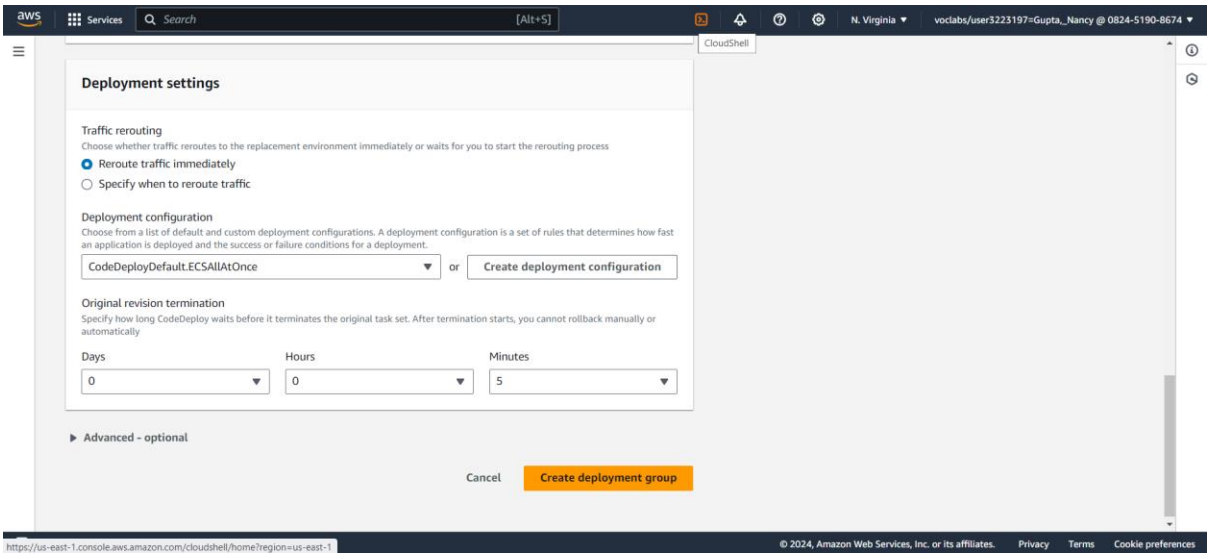
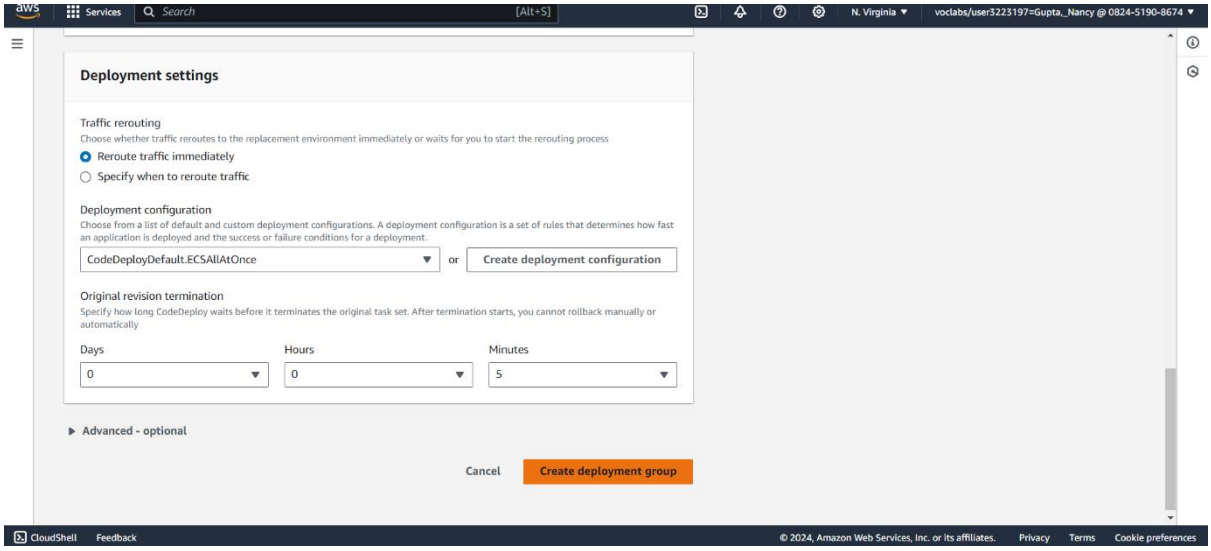
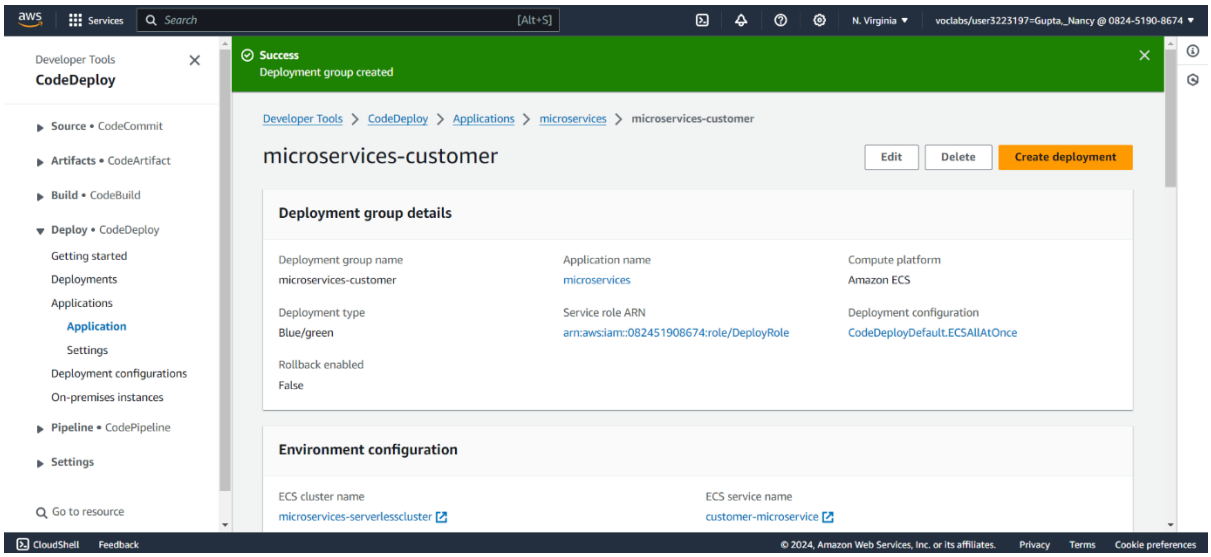


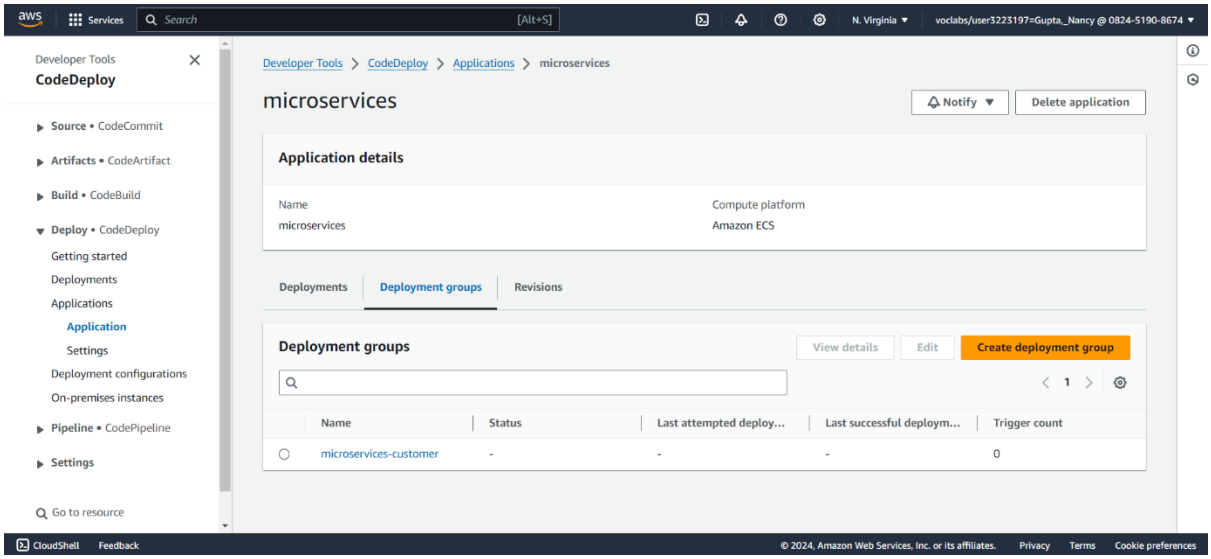
- In the **Deployment settings** section:
  - **Traffic rerouting:** Choose **Reroute traffic immediately**.
  - **Deployment configuration:** Choose **CodeDeployDefault.ECSAllAtOnce**.
  - **Original revision termination:** Days: **0**, Hours: **0**, Minutes: **5**



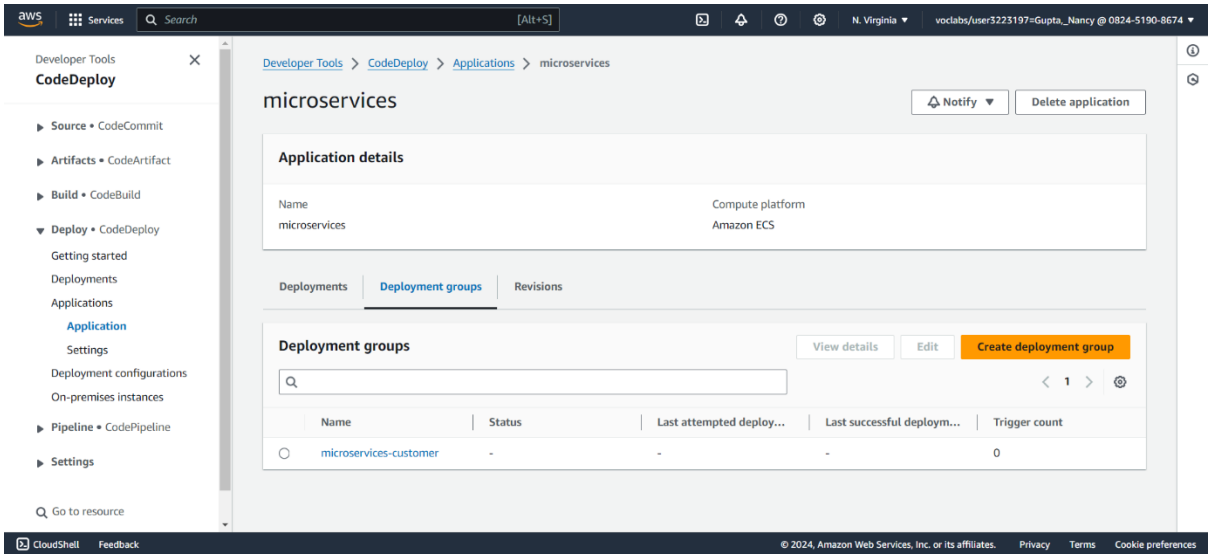
- Choose **Create deployment group**.



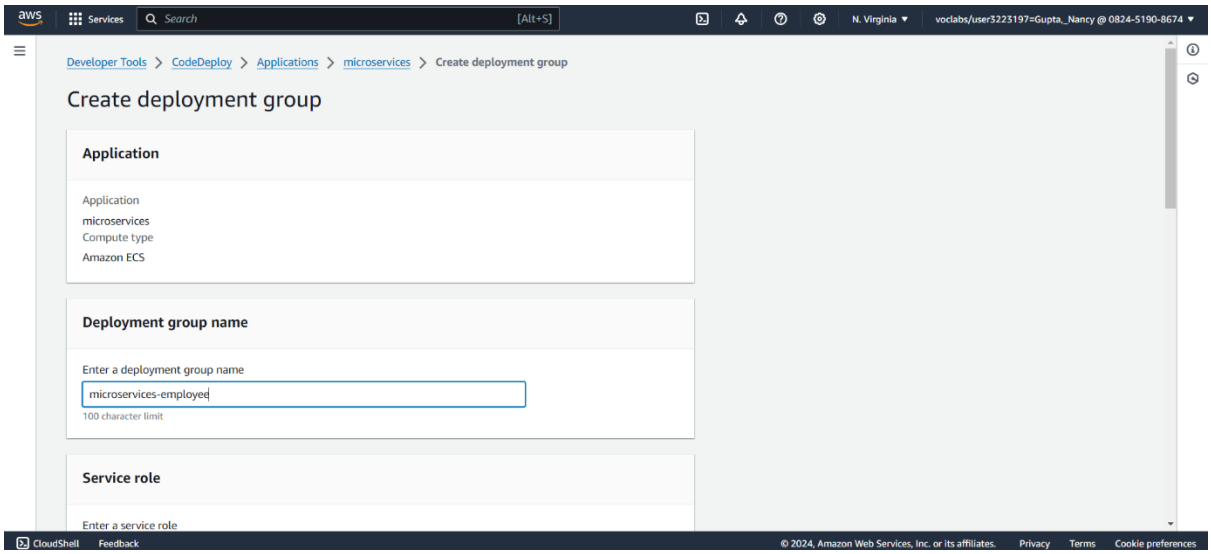


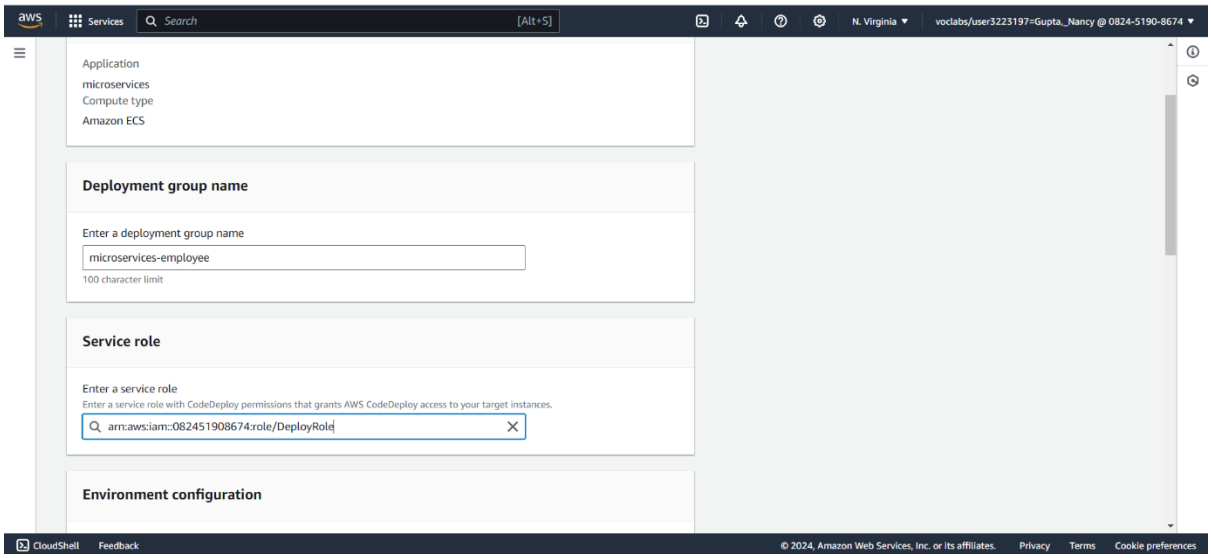


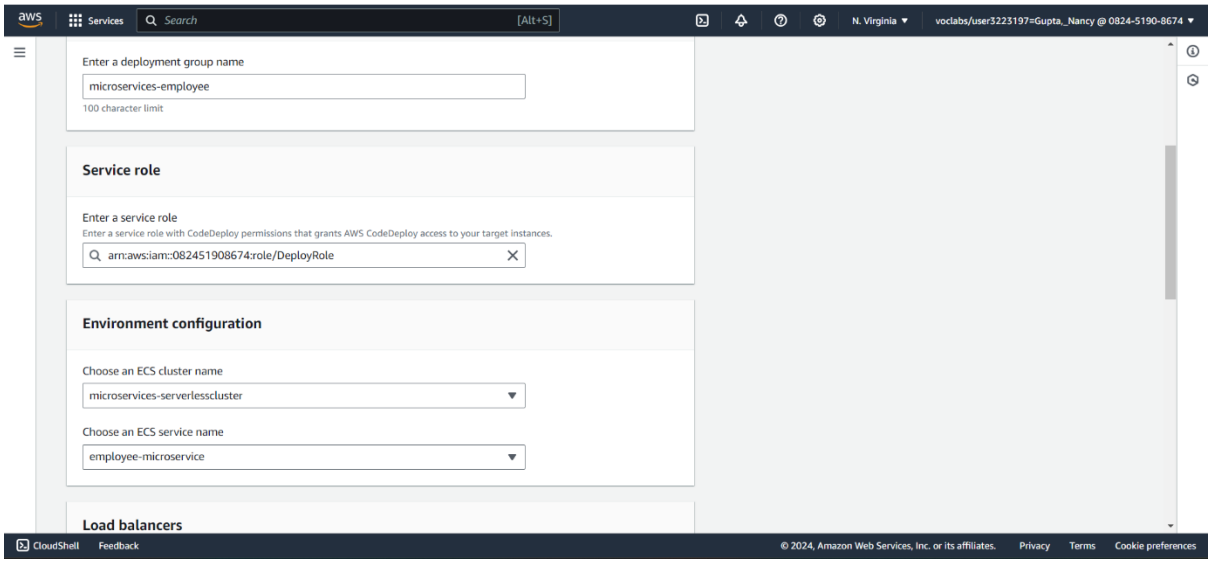
3. Create a CodeDeploy deployment group for the *employee* microservice. Specify the same settings that you did in the prior step, except for the following:

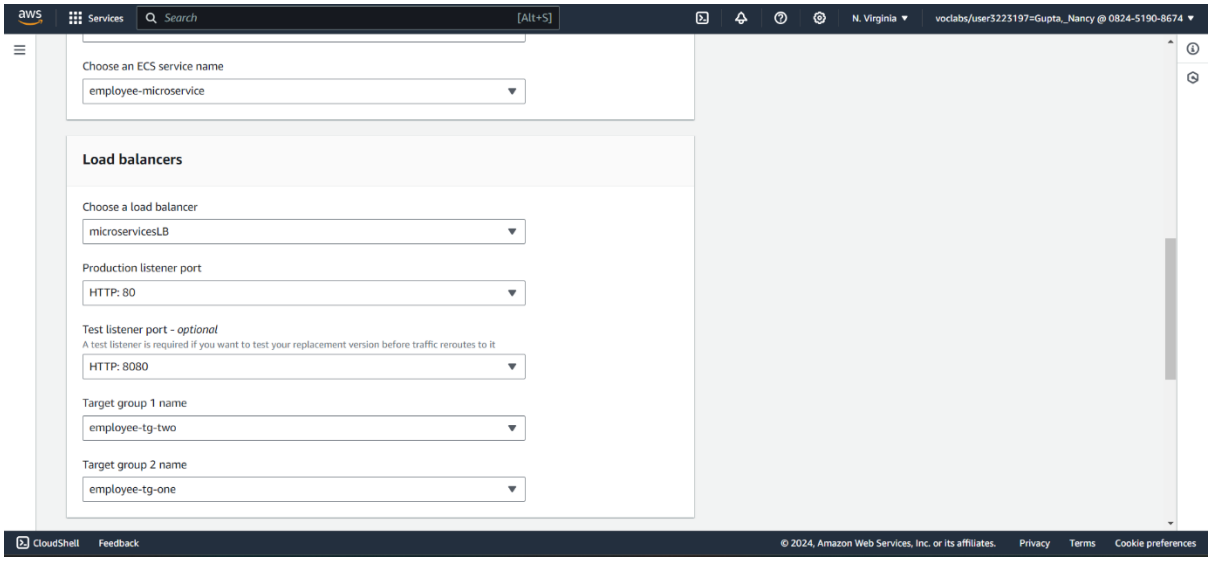


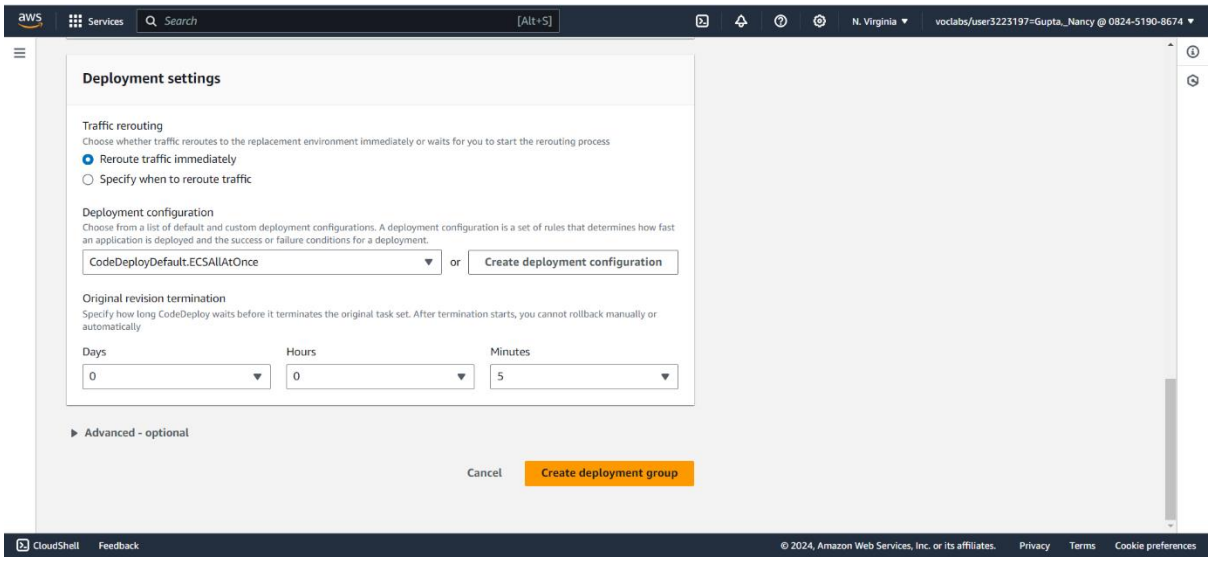
- **Deployment group name:** Enter microservices-employee
- **ECS service name:** Choose **employee-microservice**.
- **Target group 1 name:** Choose **employee-tg-two**.
- **Target group 2 name:** Choose **employee-tg-one**.

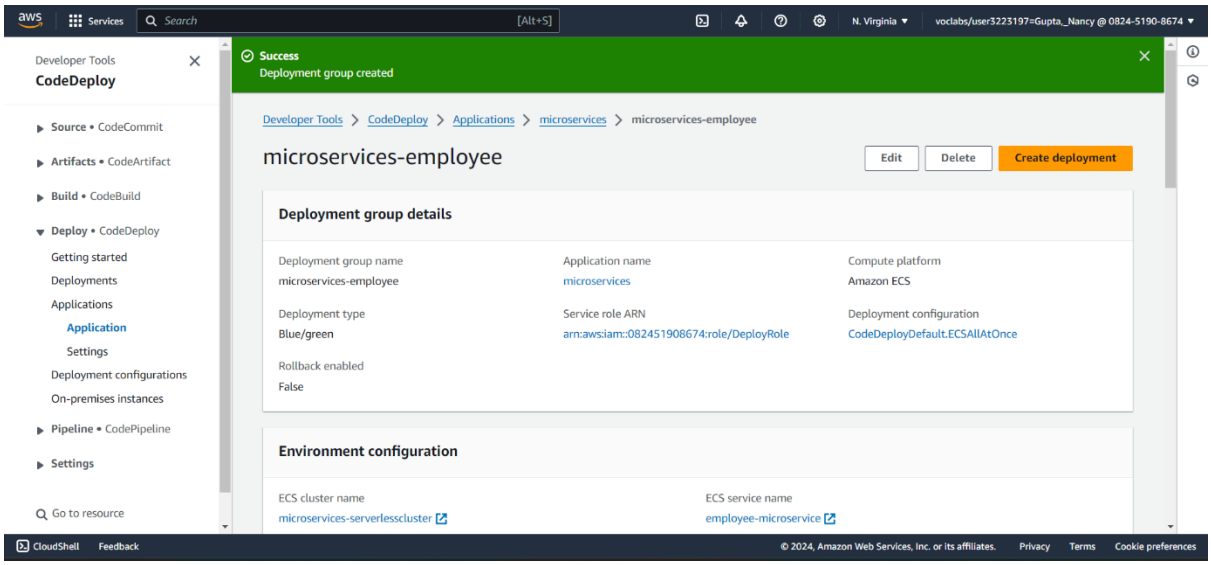


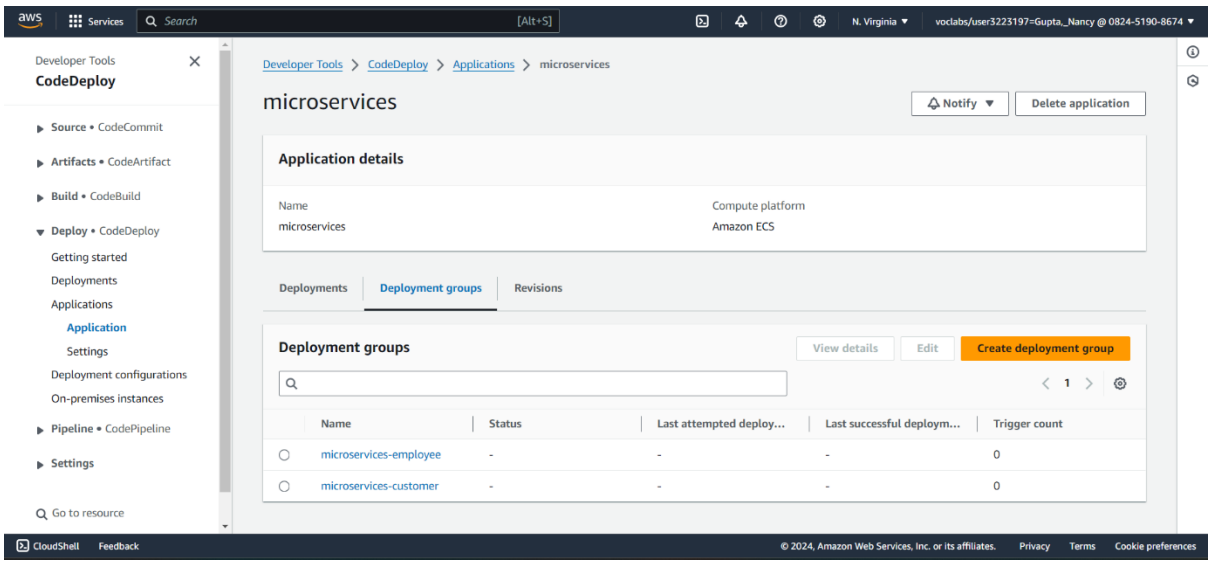

Application
microservices
Compute type
Amazon ECS
Deployment group name
Enter a deployment group name
microservices-employee
100 character limit
Service role
Enter a service role
Enter a service role with CodeDeploy permissions that grants AWS CodeDeploy access to your target instances.
arn:aws:iam::082451908674:role/DeployRole
Environment configuration


Enter a deployment group name
microservices-employee
100 character limit
Service role
Enter a service role
Enter a service role with CodeDeploy permissions that grants AWS CodeDeploy access to your target instances.
arn:aws:iam::082451908674:role/DeployRole
Environment configuration
Choose an ECS cluster name
microservices-serverlesscluster
Choose an ECS service name
employee-microservice
Load balancers


Choose an ECS service name
employee-microservice
Load balancers
Choose a load balancer
microservicesLB
Production listener port
HTTP: 80
Test listener port - optional
A test listener is required if you want to test your replacement version before traffic reroutes to it
HTTP: 8080
Target group 1 name
employee-tg-two
Target group 2 name
employee-tg-one







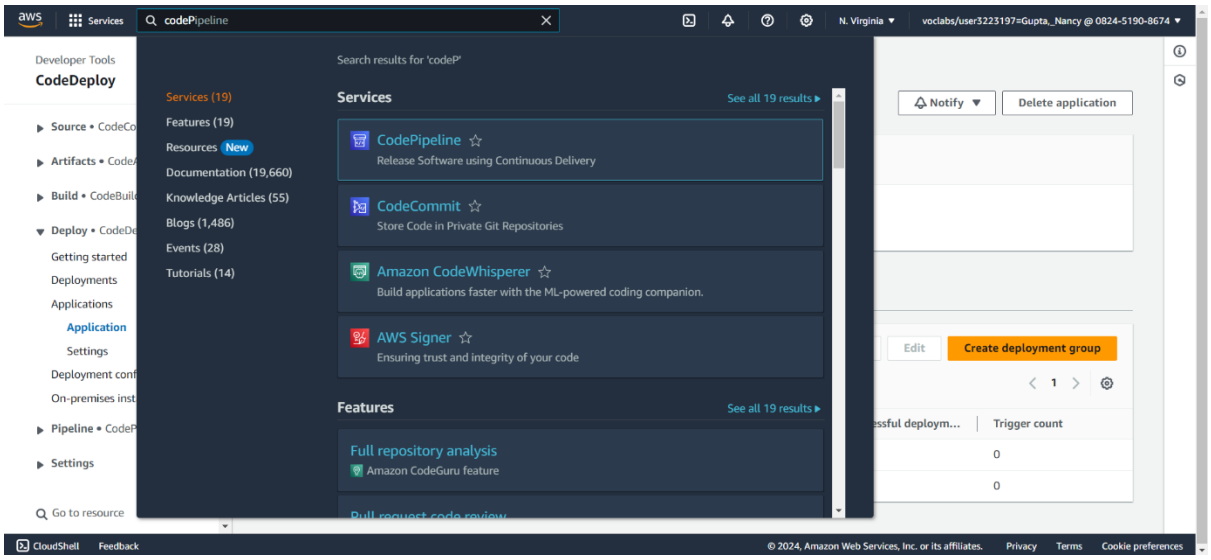
Task 8.2: Create a pipeline for the *customer* microservice

In this task, you will create a pipeline to update the *customer* microservice. When you first define the pipeline, you will configure CodeCommit as the source and CodeDeploy as the service that is responsible for deployment. You will then edit the pipeline to add the Amazon ECR service as a second source.

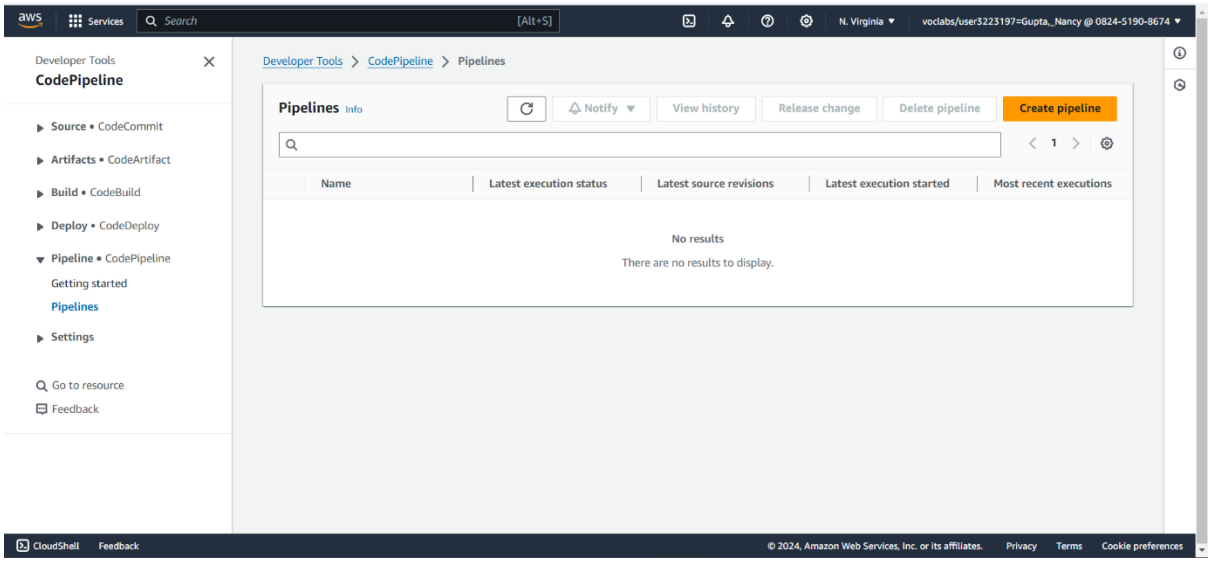
With an Amazon ECS blue/green deployment, which you will specify in this task, you provision a new set of containers, which CodeDeploy installs the latest version of your application on. CodeDeploy then reroutes load balancer traffic from an existing set of containers, which run the previous version of your application, to the new set of containers, which run the latest version. After traffic is rerouted to the new containers, the existing containers can be terminated. With a blue/green deployment, you can test the new application version before sending production traffic to it.

References

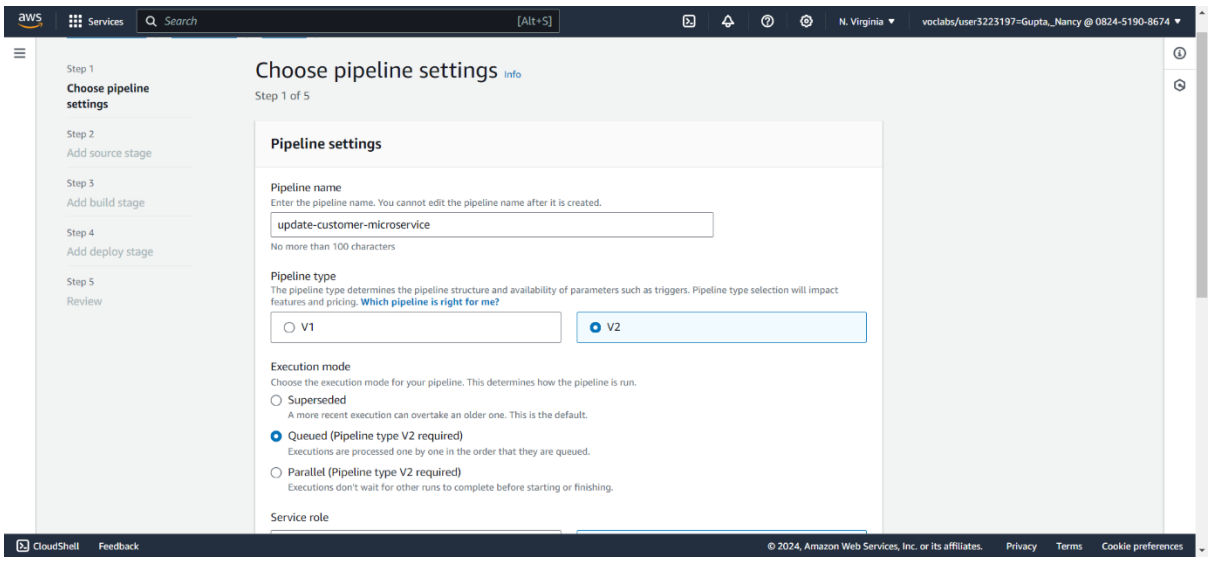
- The AWS Academy Cloud Architecting and AWS Academy Cloud Developing courses include hands-on labs that explore CodePipeline features.
- AWS CodePipeline User Guide



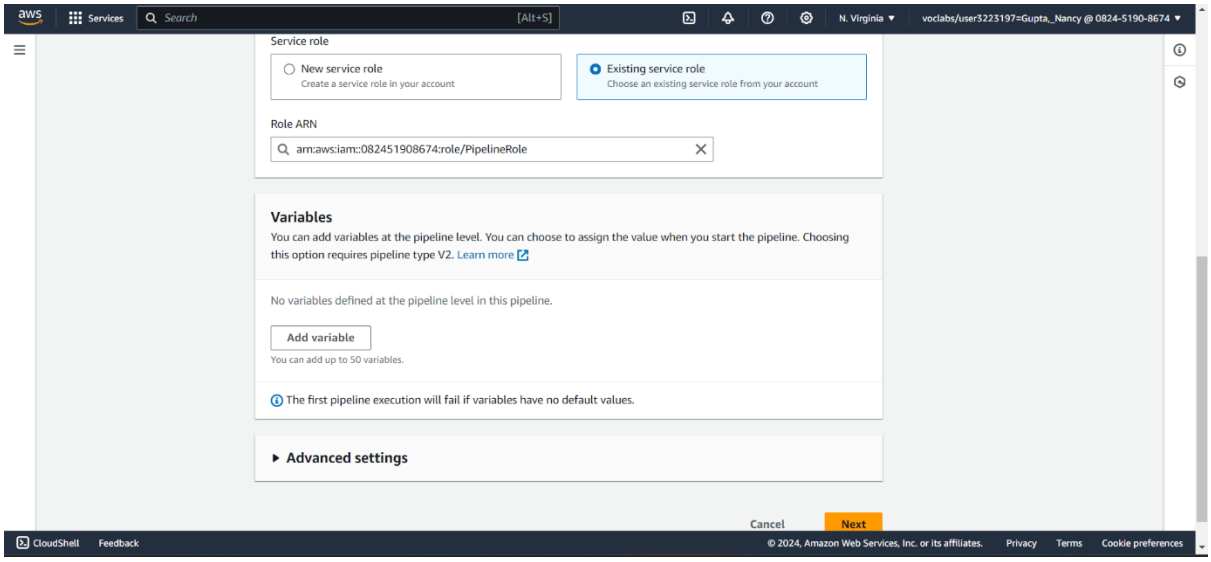
1. **In the CodePipeline console, create a *customer* pipeline with the following settings:**



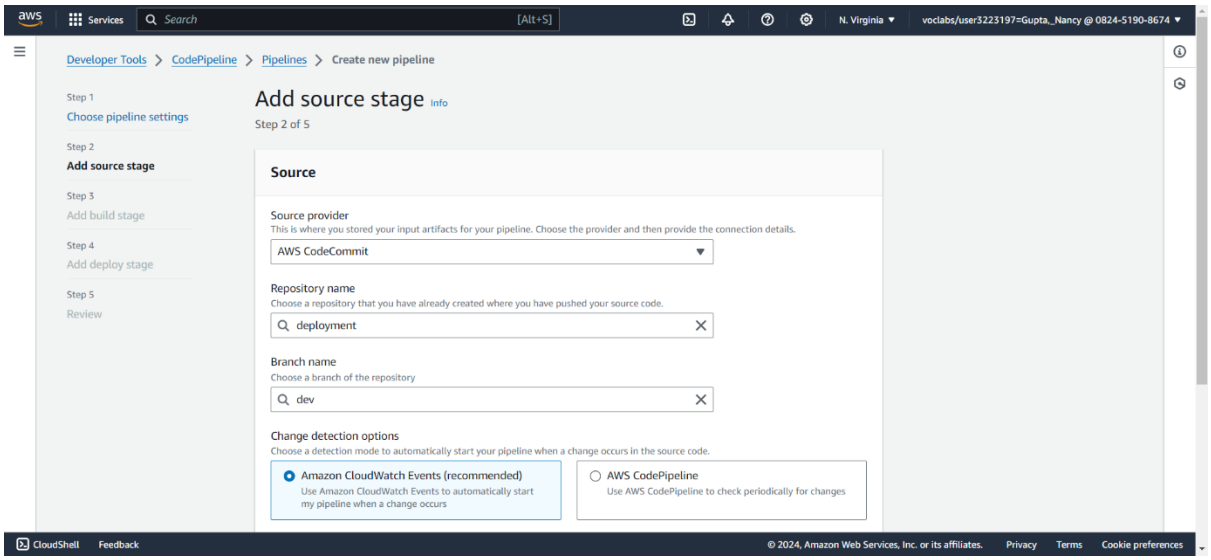
- **Pipeline name:** Enter update-customer-microservice



- **Service role:** Choose the ARN for **PipelineRole**.

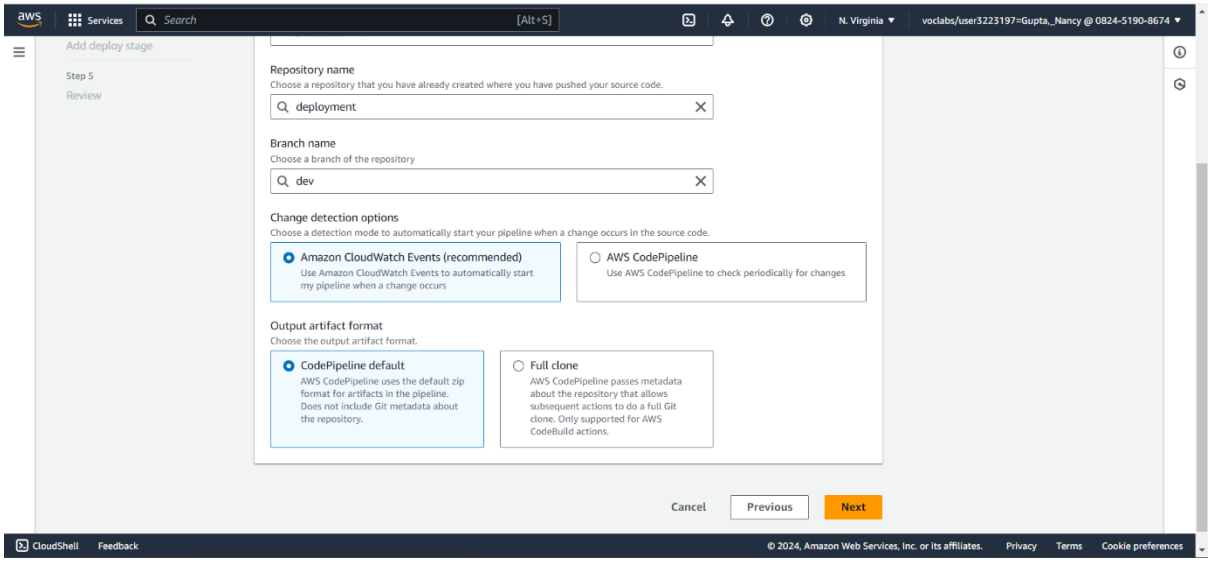


- **Source provider:** Choose **AWS CodeCommit**.
    - **Repository name:** Choose **deployment**.

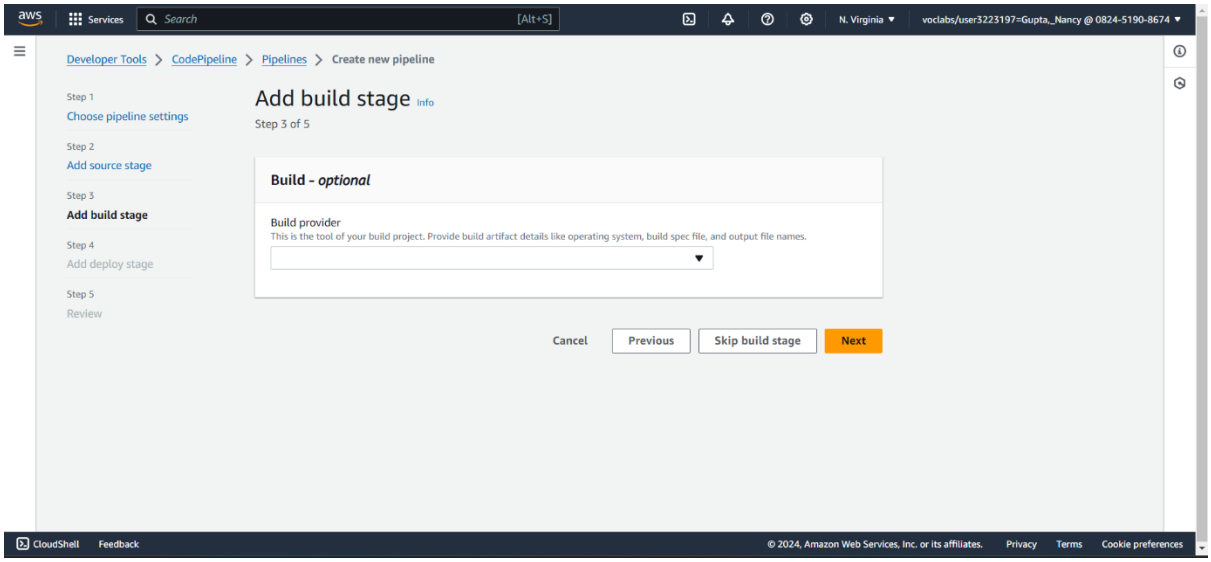


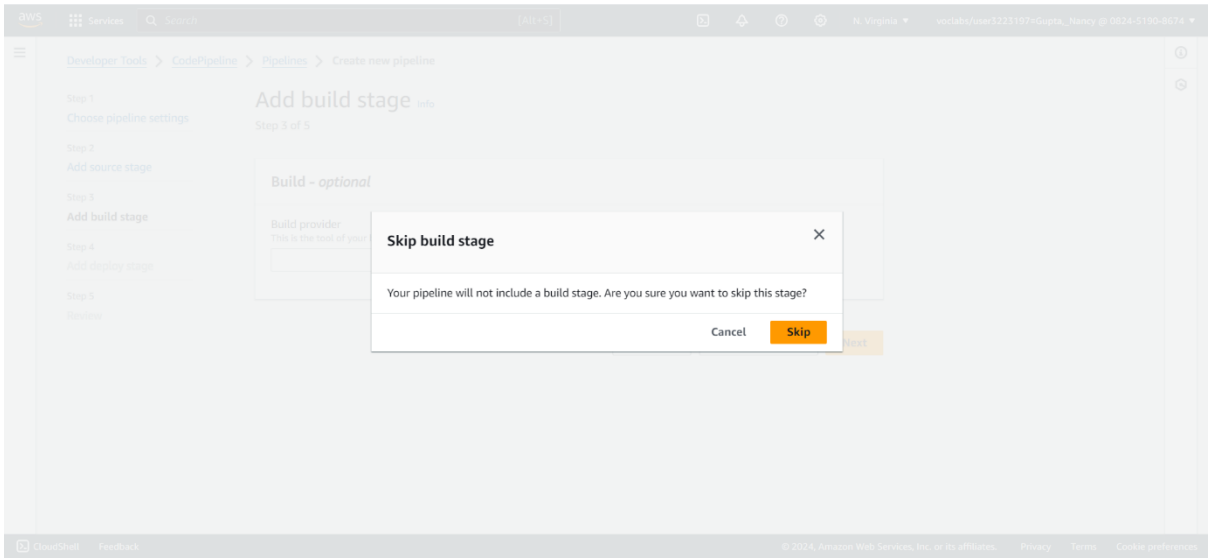
Note: You have defined two CodeCommit repositories. The *deployment* repository contains the Amazon ECS task definition files and CodeDeploy AppSpec files that your pipeline will need, so that is the one you choose here.

- **Branch name:** Choose **dev**.

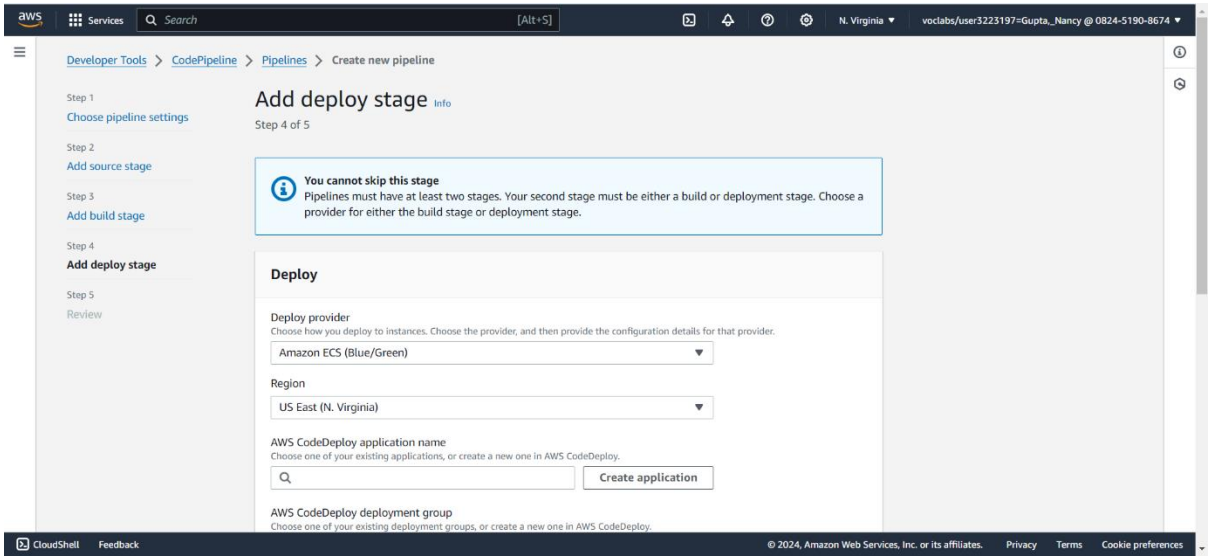


Note: *Skip* the build stage.



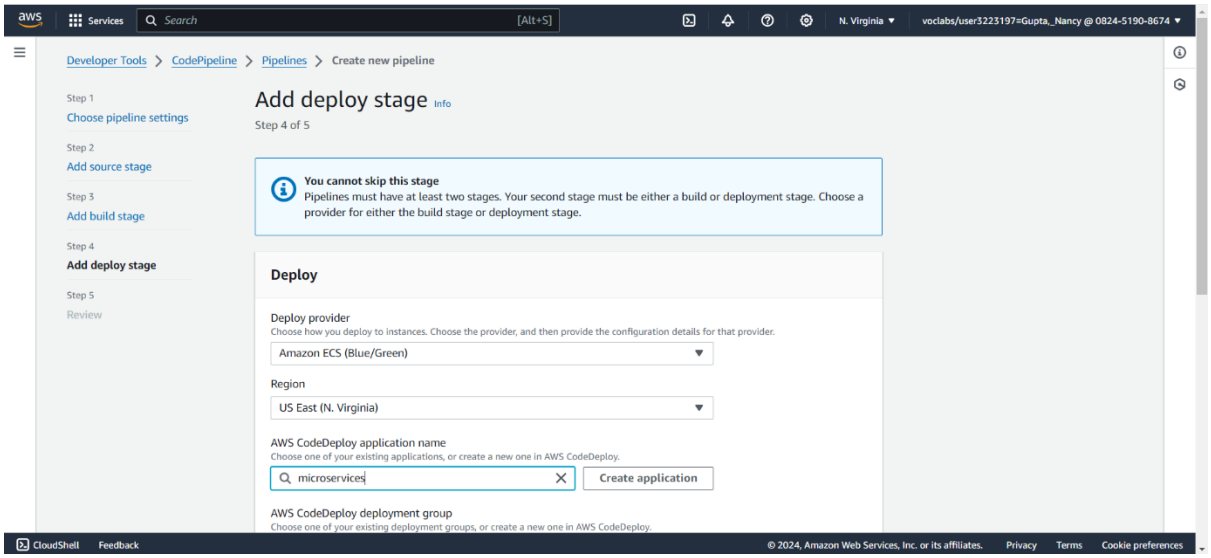


- **Deploy provider: Amazon ECS (Blue/Green)**

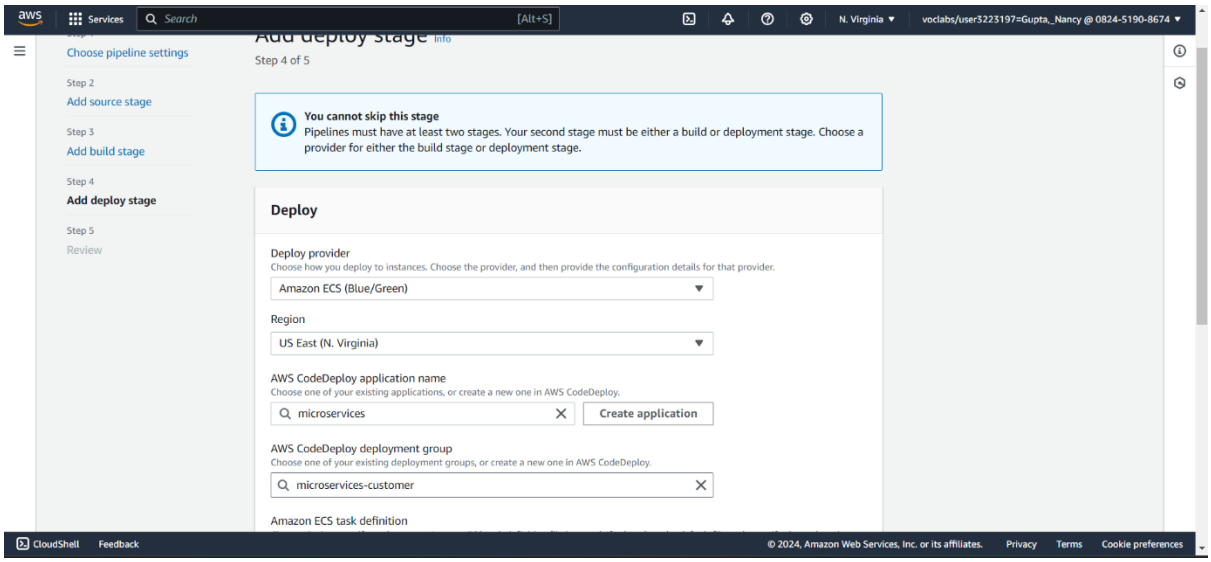


- **Region:** US East (N. Virginia).

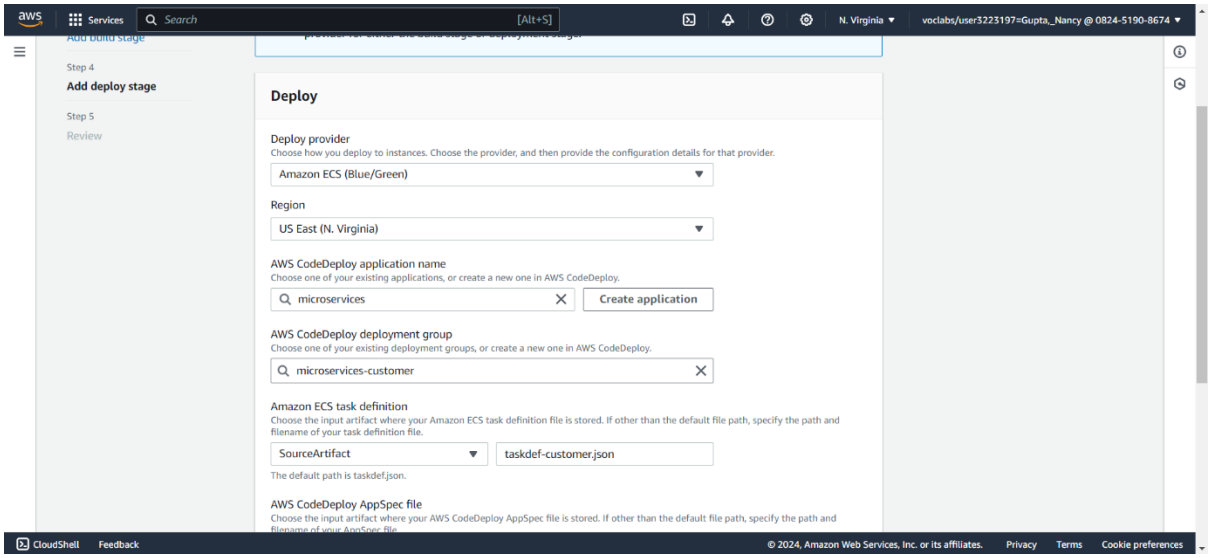
- **AWS CodeDeploy application name:** microservices



- **AWS CodeDeploy deployment group:** microservices-customer

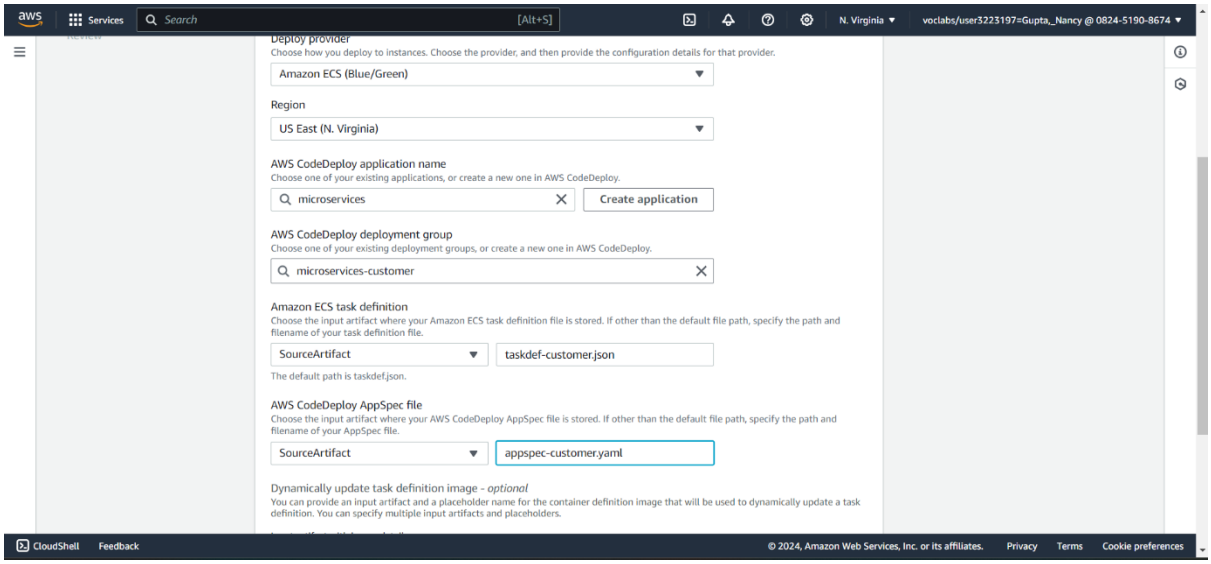


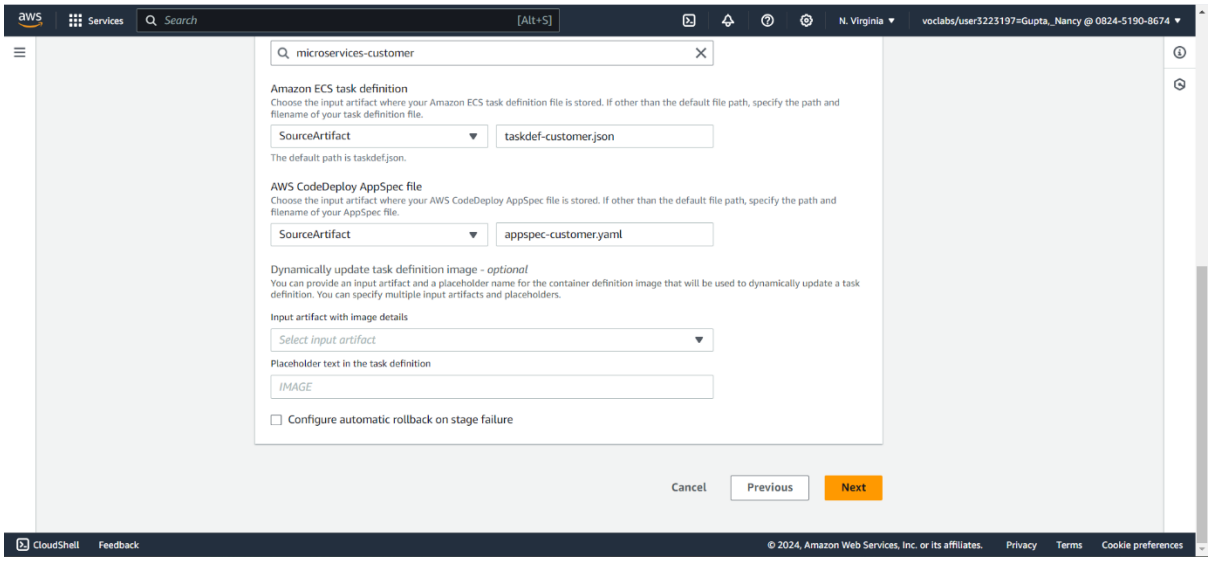
- Under **Amazon ECS task definition**:
    - Set a **SourceArtifact** with a value of taskdef-customer.json

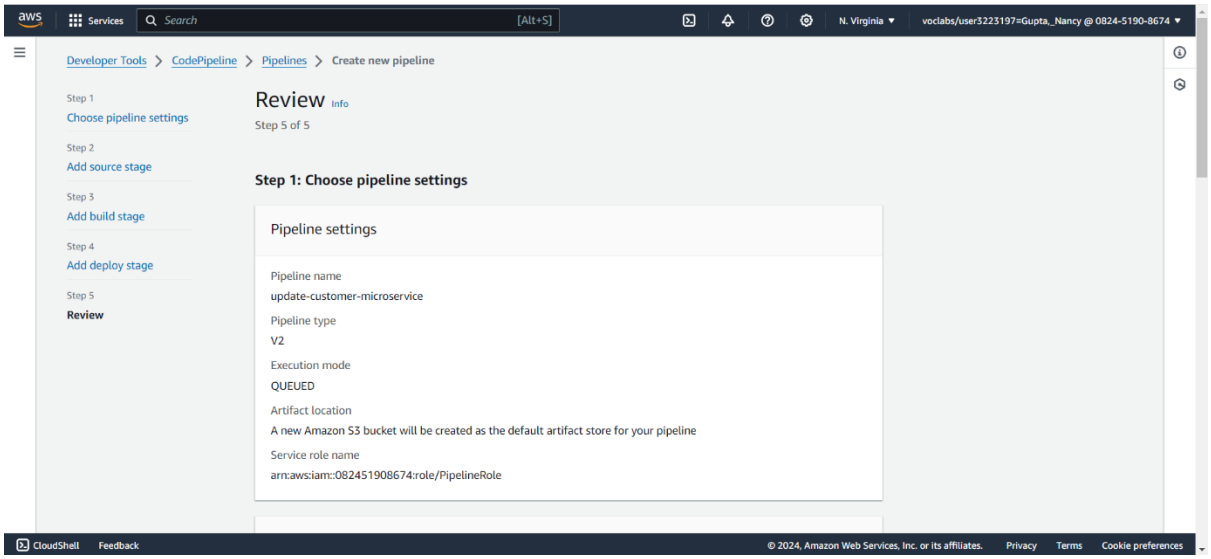


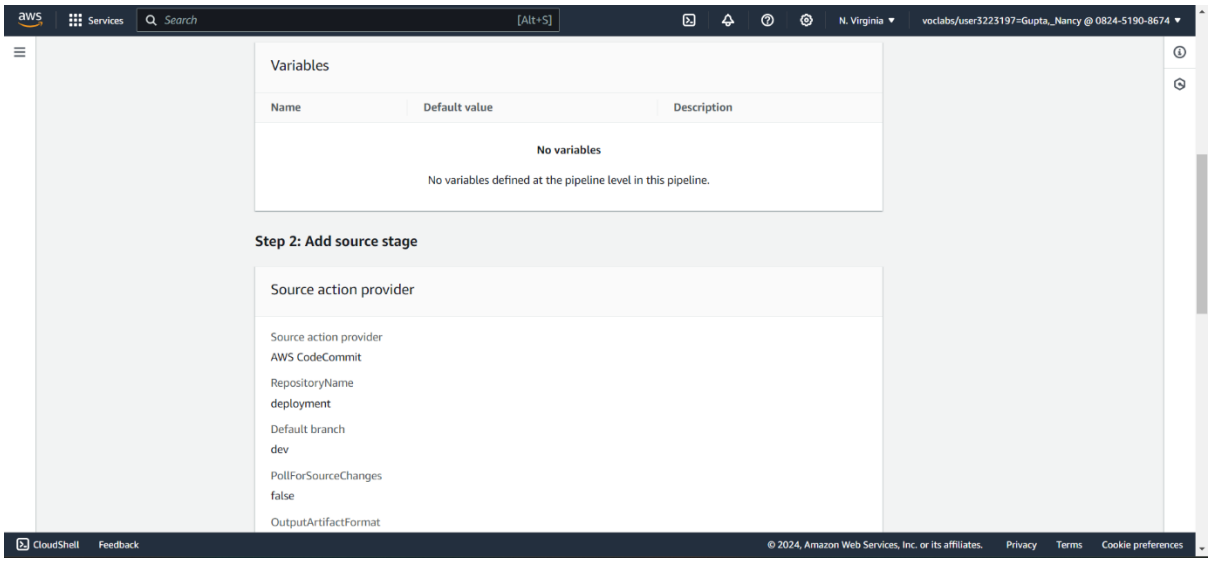
- Under **AWS CodeDeploy AppSpec file**:
  - Set a **SourceArtifact** with a value of appspec-customer.yaml

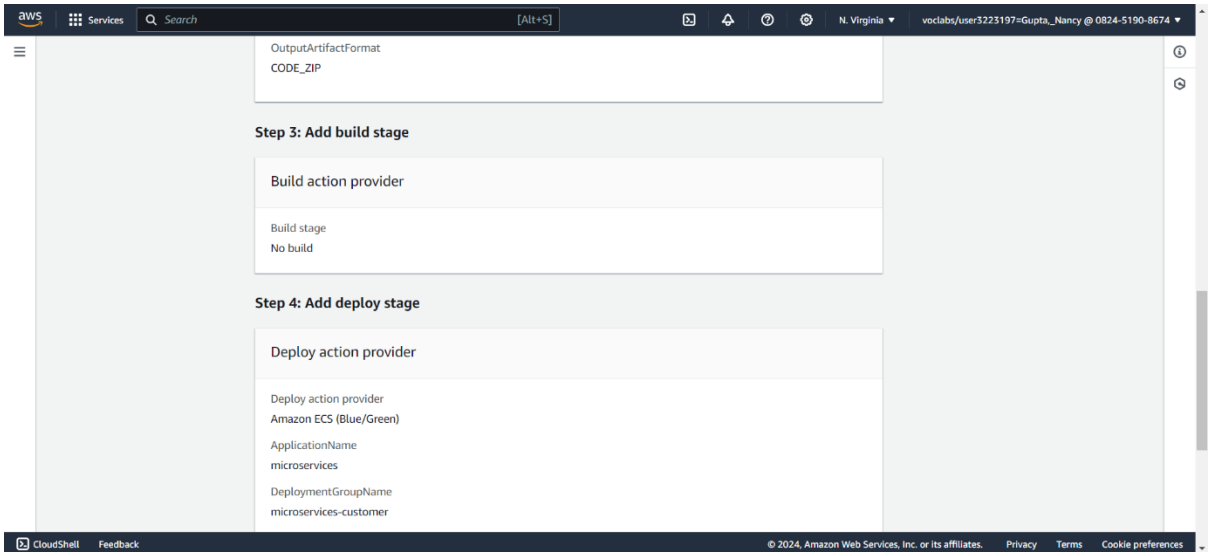
Note: Leave the *Dynamically update task definition image* fields *blank* for now.

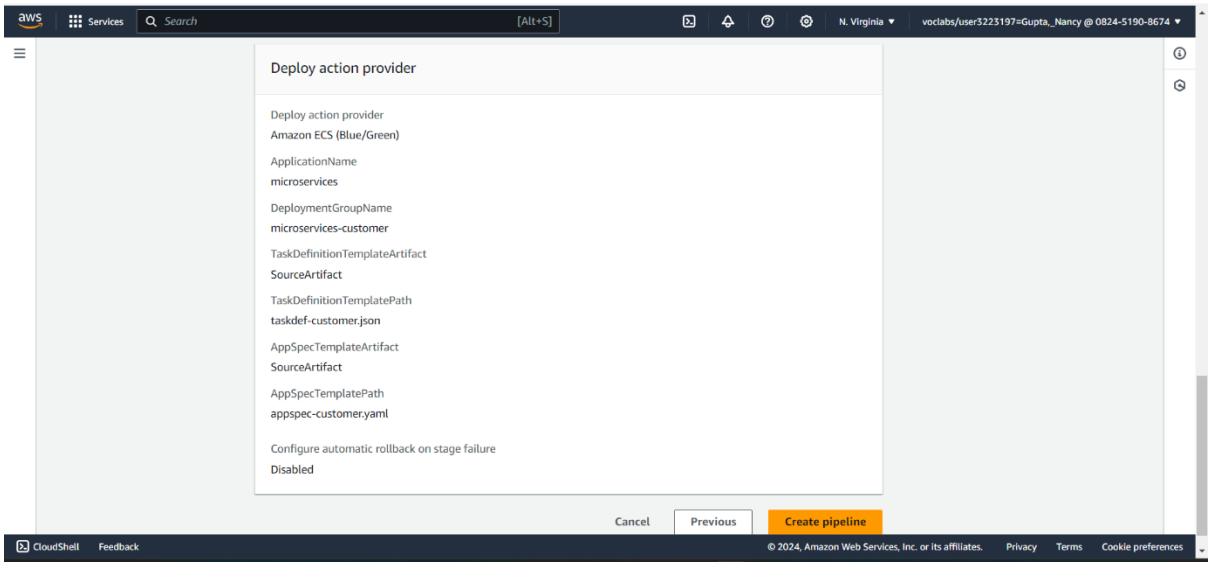



Services
Search
[Alt+S]
N. Virginia
voclabs/user3223197=Gupta,_Nancy @ 0824-5190-8674
microservices-customer
Amazon ECS task definition
Choose the input artifact where your Amazon ECS task definition file is stored. If other than the default file path, specify the path and filename of your task definition file.
SourceArtifact
taskdef-customer.json
The default path is taskdef.json.
AWS CodeDeploy AppSpec file
Choose the input artifact where your AWS CodeDeploy AppSpec file is stored. If other than the default file path, specify the path and filename of your AppSpec file.
SourceArtifact
appspec-customer.yaml
Dynamically update task definition image - optional
You can provide an input artifact and a placeholder name for the container definition image that will be used to dynamically update a task definition. You can specify multiple input artifacts and placeholders.
Input artifact with image details
Select input artifact
Placeholder text in the task definition
IMAGE
Configure automatic rollback on stage failure
Cancel
Previous
Next
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

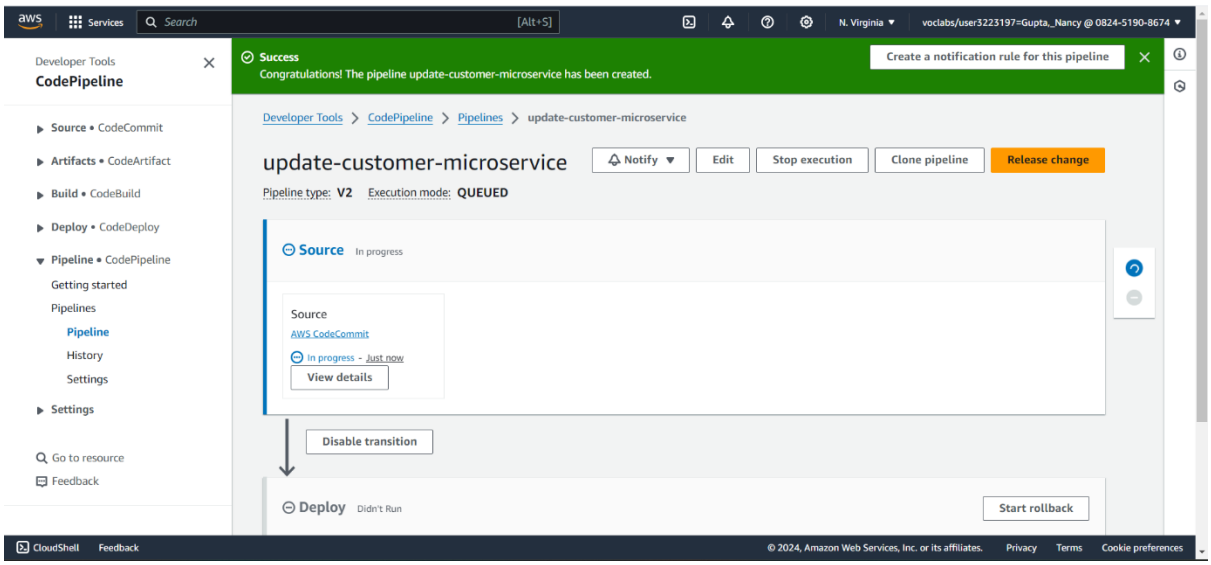

Services
Search
[Alt+S]
N. Virginia
voclabs/user3223197=Gupta,_Nancy @ 0824-5190-8674
Developer Tools
CodePipeline
Pipelines
Create new pipeline
Step 1
Choose pipeline settings
Step 2
Add source stage
Step 3
Add build stage
Step 4
Add deploy stage
Step 5
Review
Review
Info
Step 5 of 5
Step 1: Choose pipeline settings
Pipeline settings
Pipeline name
update-customer-microservice
Pipeline type
V2
Execution mode
QUEUED
Artifact location
A new Amazon S3 bucket will be created as the default artifact store for your pipeline
Service role name
arn:aws:iam::082451908674:role/PipelineRole
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

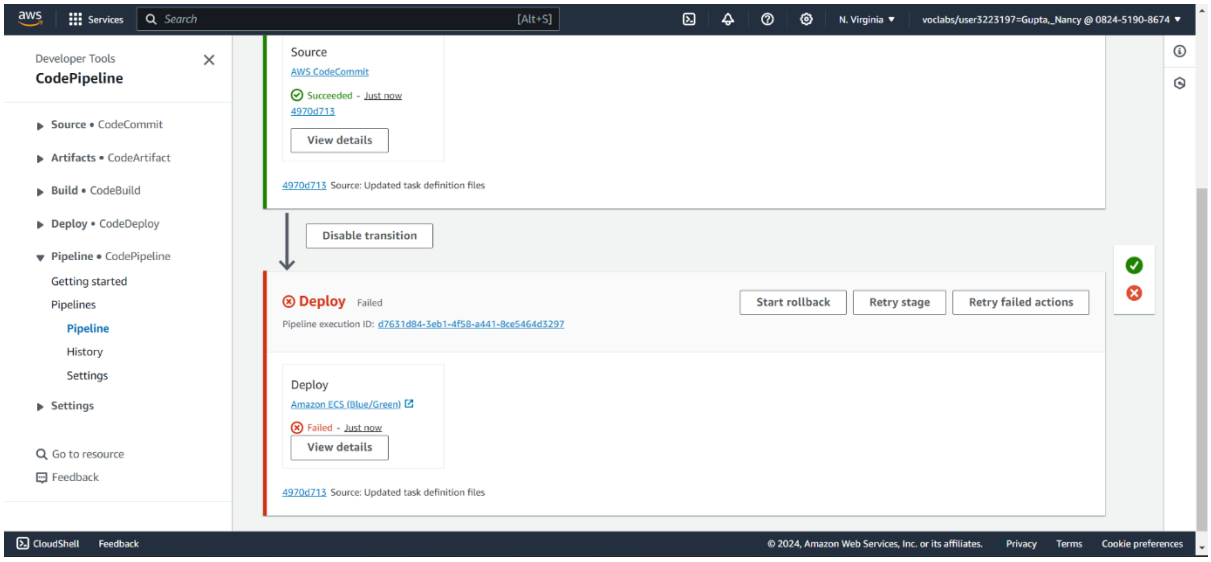

Services
Search
[Alt+S]
N. Virginia
voclabs/user3223197=Gupta,_Nancy @ 0824-5190-8674
Variables
Name
Default value
Description
No variables
No variables defined at the pipeline level in this pipeline.
Step 2: Add source stage
Source action provider
Source action provider
AWS CodeCommit
RepositoryName
deployment
Default branch
dev
PollForSourceChanges
false
OutputArtifactFormat
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences



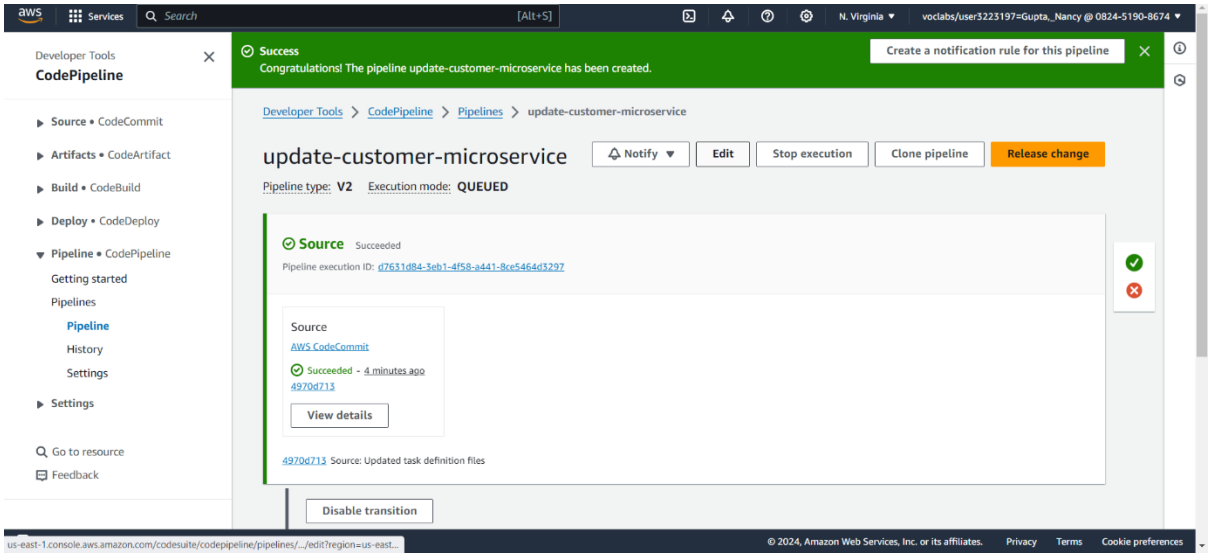


Note: After you create the pipeline, it will immediately start to run and will eventually fail on the *Deploy* stage. Ignore that for now and continue to the next step.

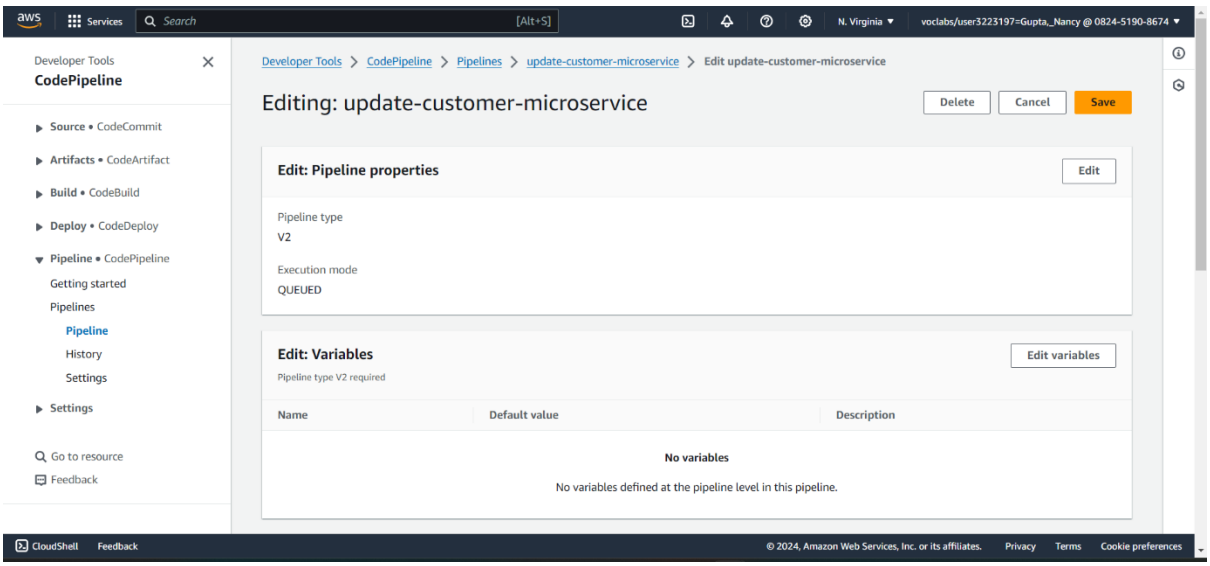


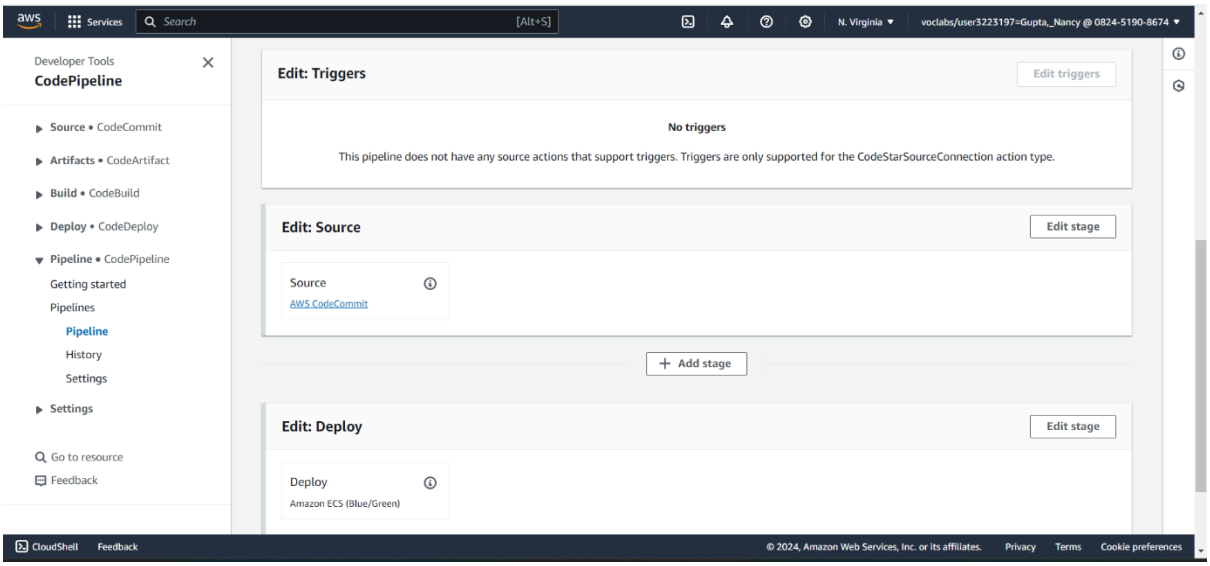


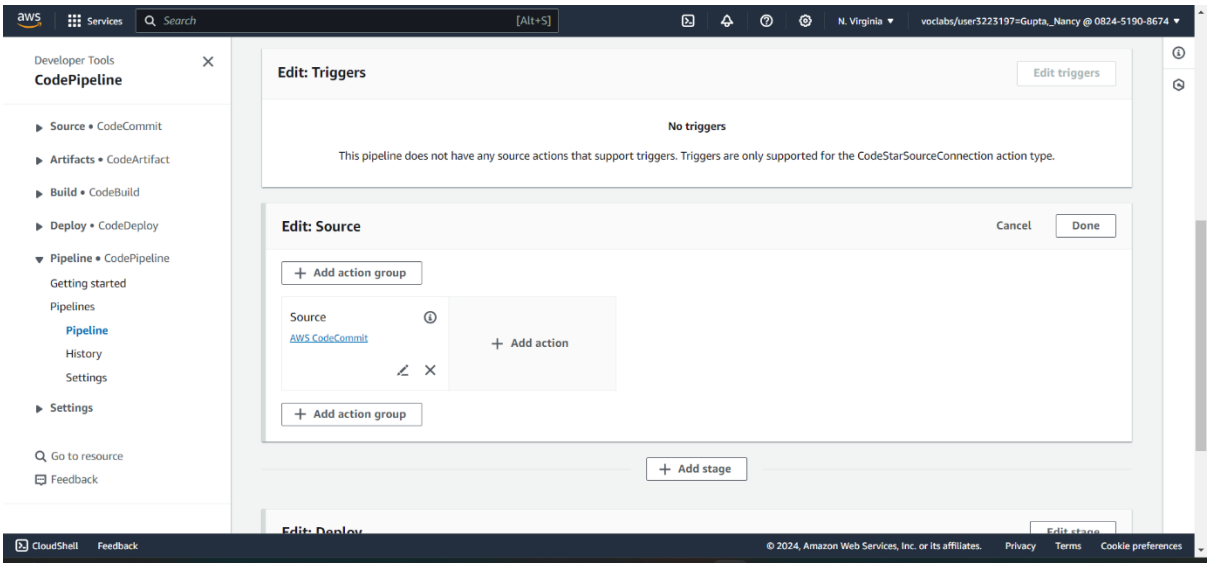
2. Edit the *update-customer-microservice* pipeline to add another *source*.



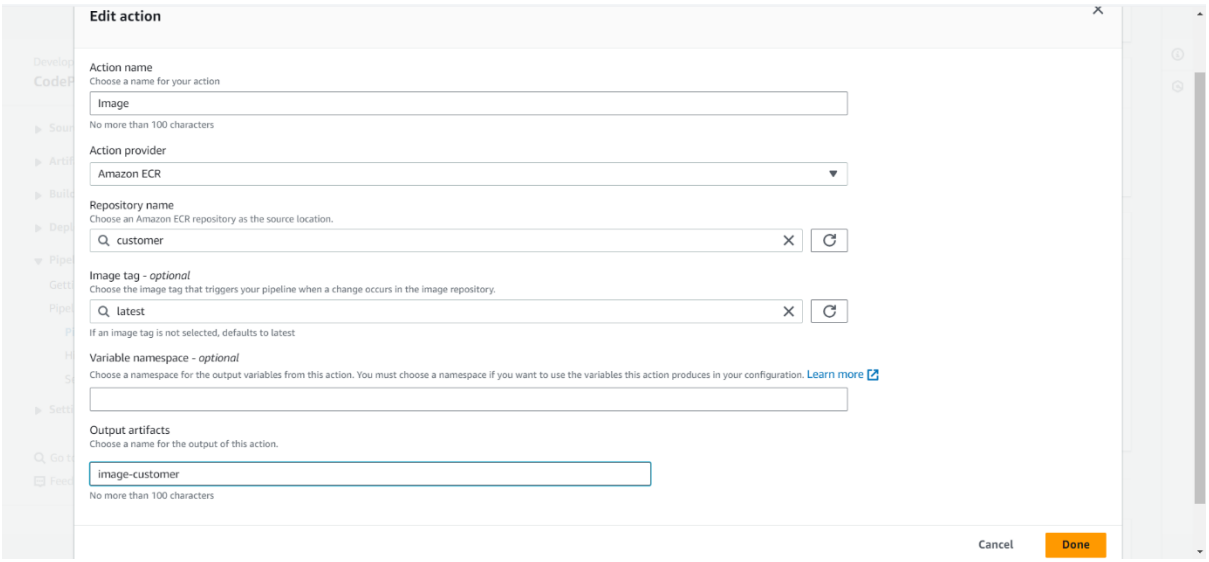
- In the **Edit: Source** section, choose **Edit stage**, then add an action with these details:

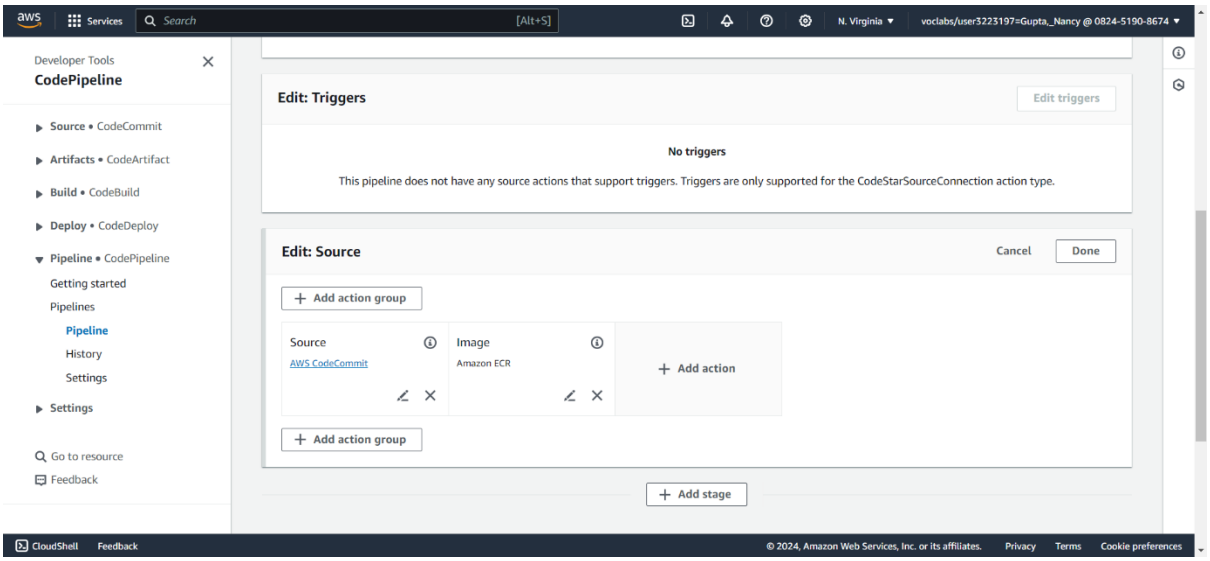






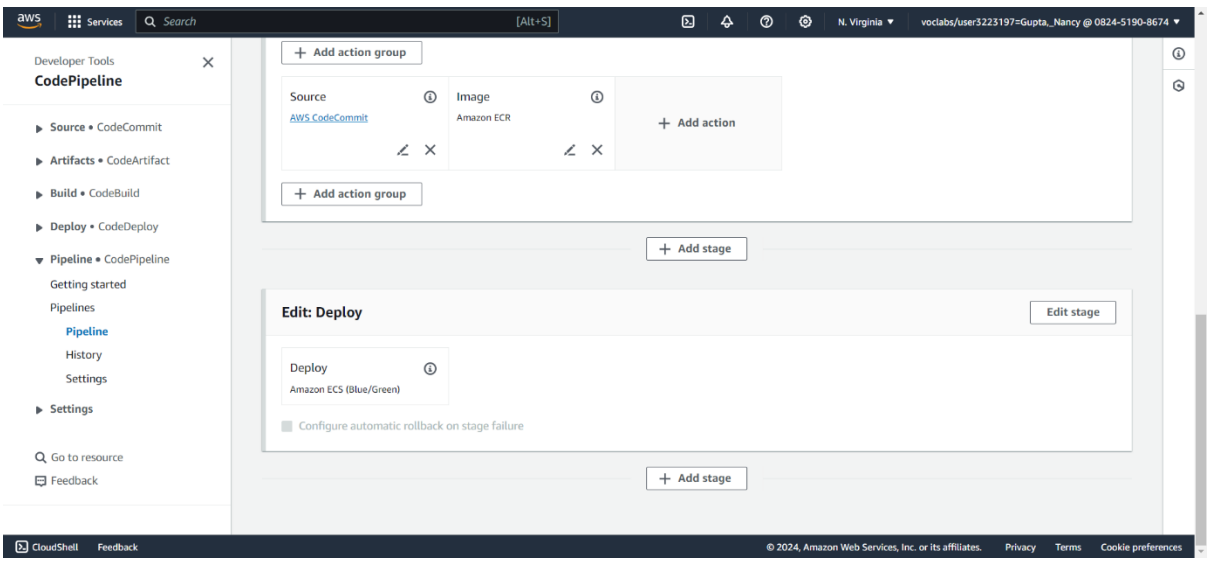
- **Action name:** Image
- **Action provider:** Amazon ECR
- **Repository name:** customer
- **Image tag:** latest
- **Output artifacts:** image-customer



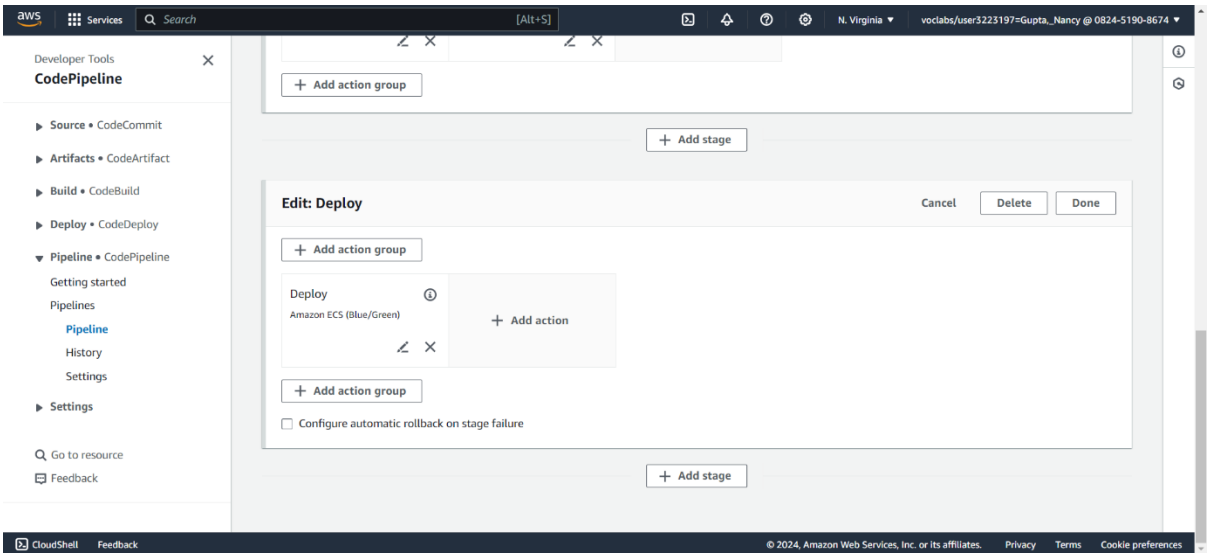


3. **Edit the *deploy* action of the *update-customer-microservice* pipeline.**

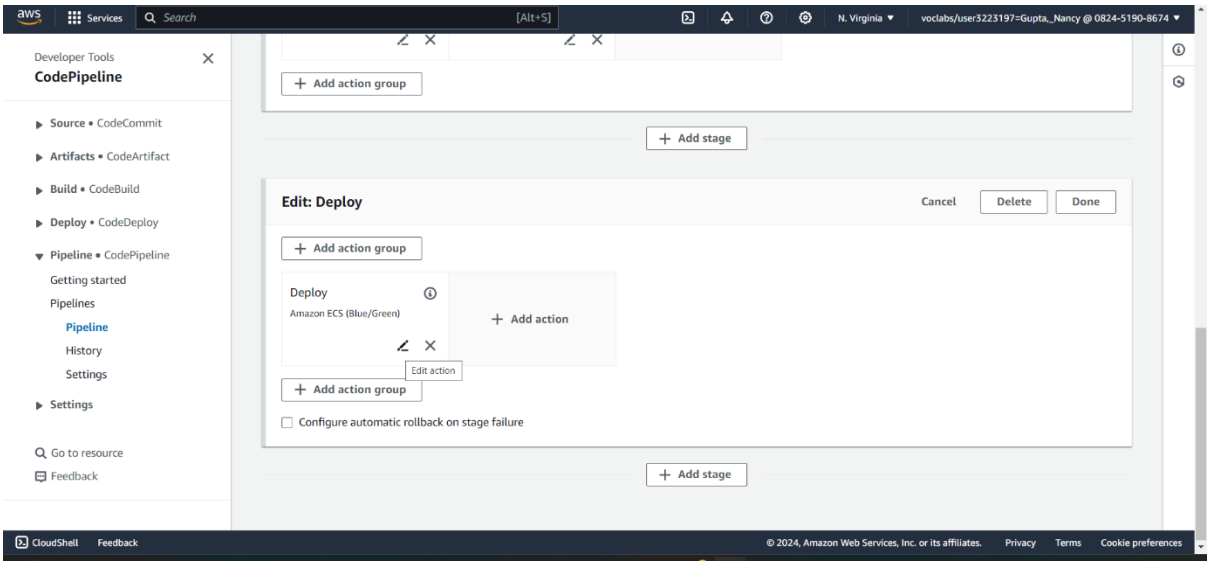
   - Edit the **update-customer-microservice** pipeline

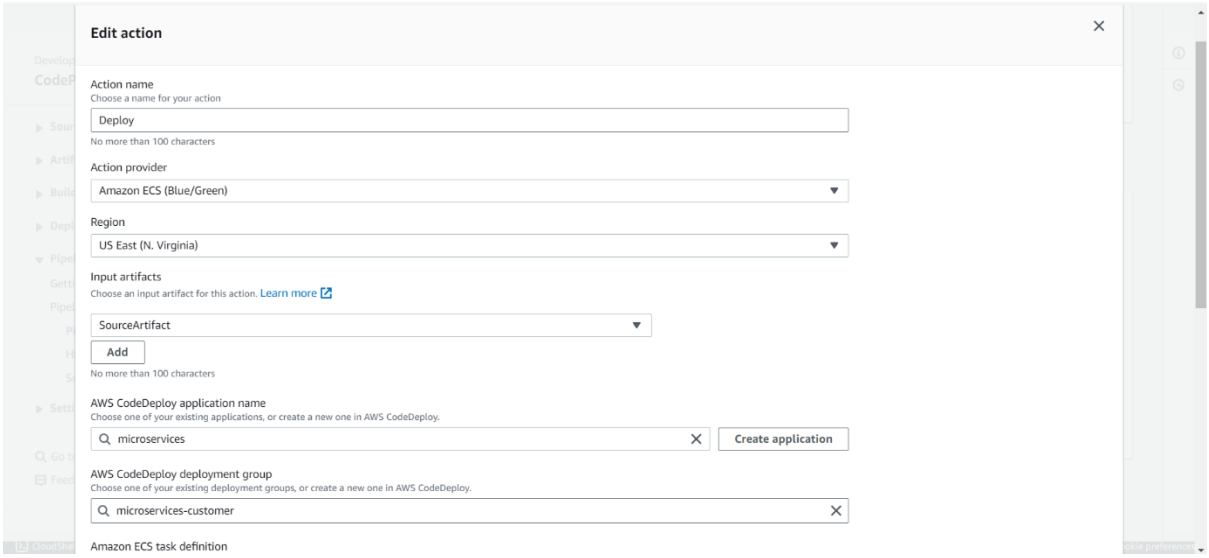


- In the **Edit: Deploy** section, choose **Edit stage**, then add an input artifact as described below:



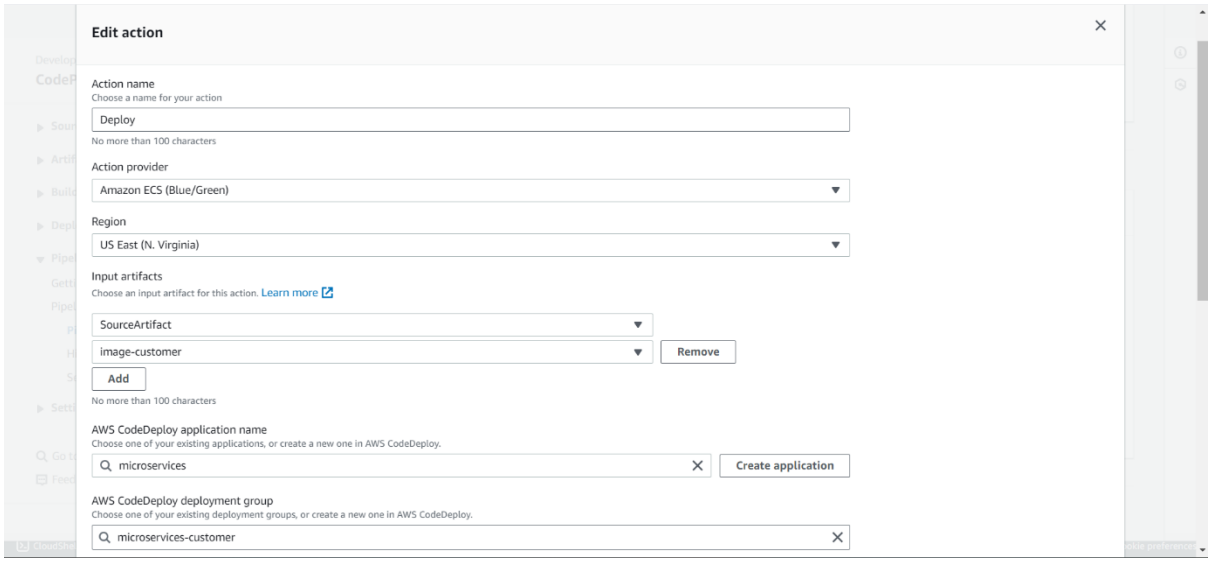
- On the **Deploy Amazon ECS (Blue/Green)** card, choose the edit (pencil) icon.



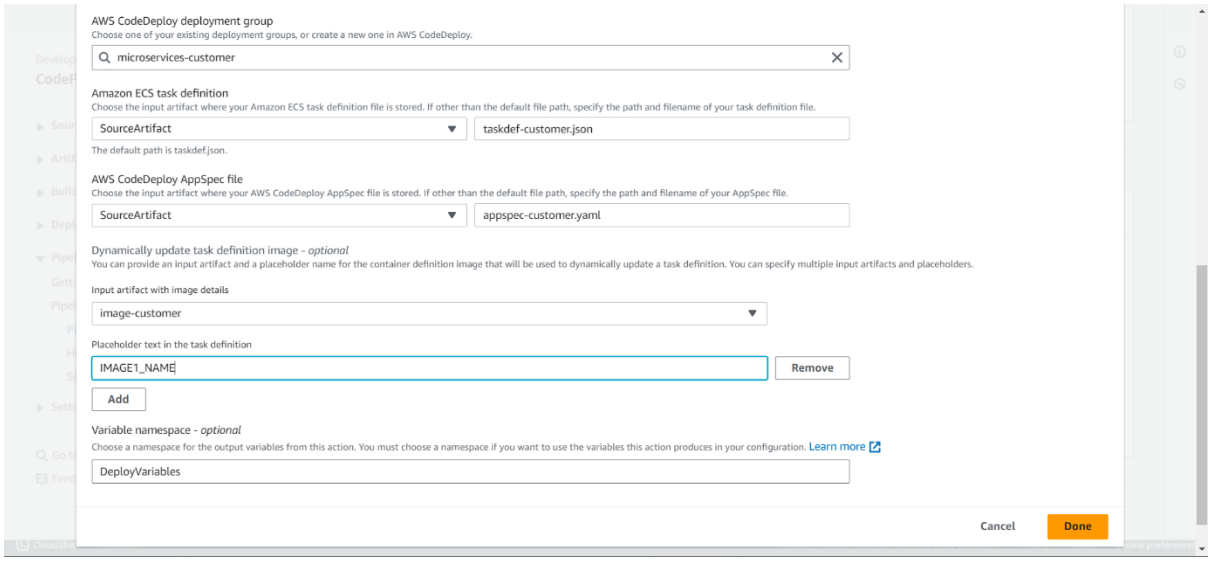


- Under **Input artifacts**, choose **Add** and then choose **image-customer**.

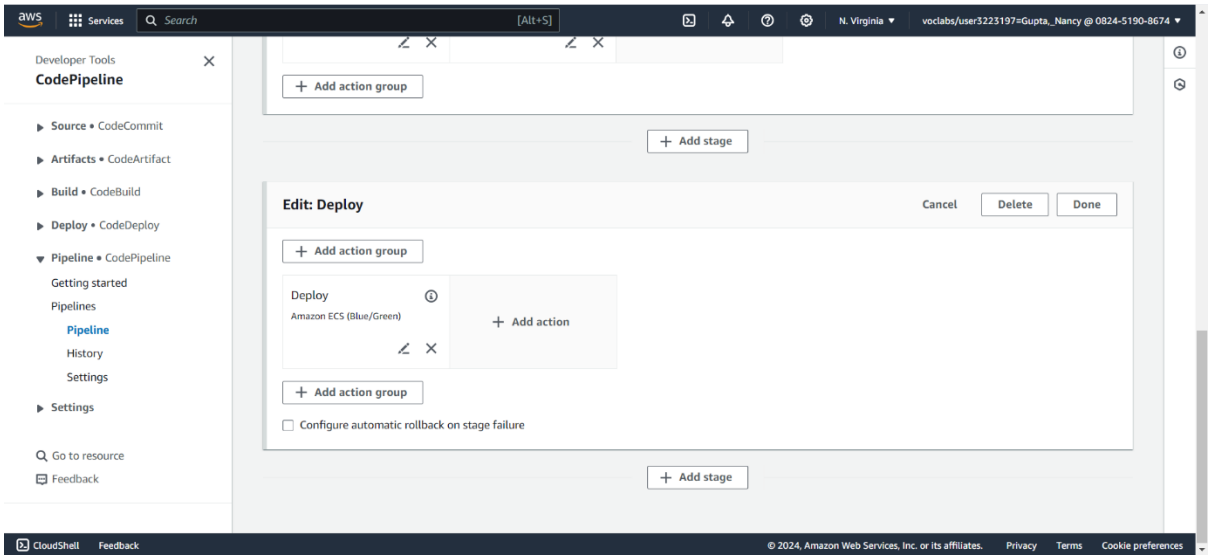
Note: You should now have *SourceArtifact* and *image-customer* as listed input artifacts.

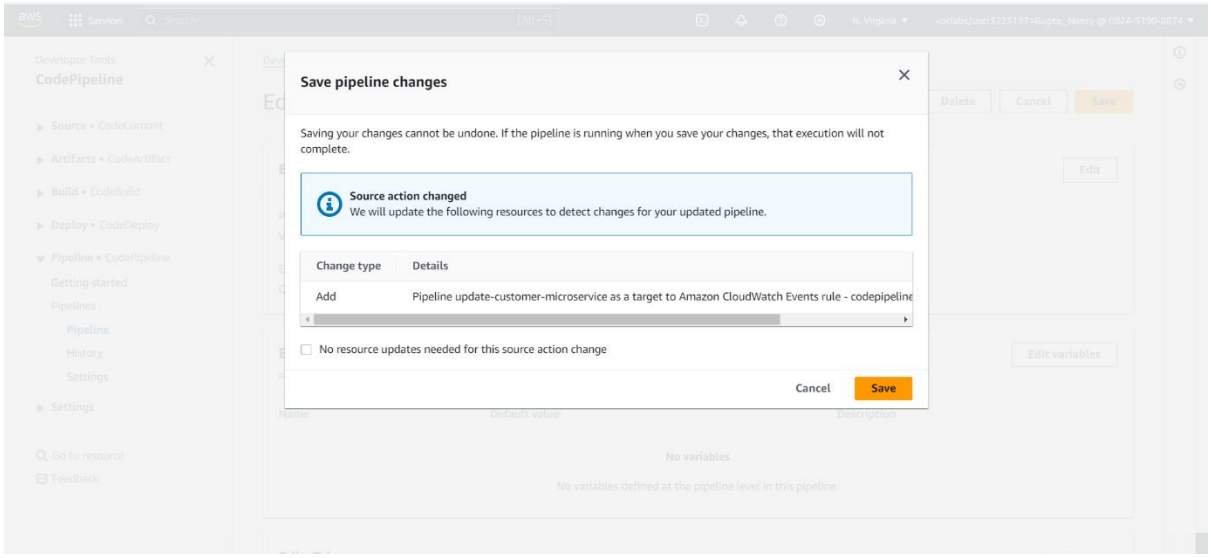


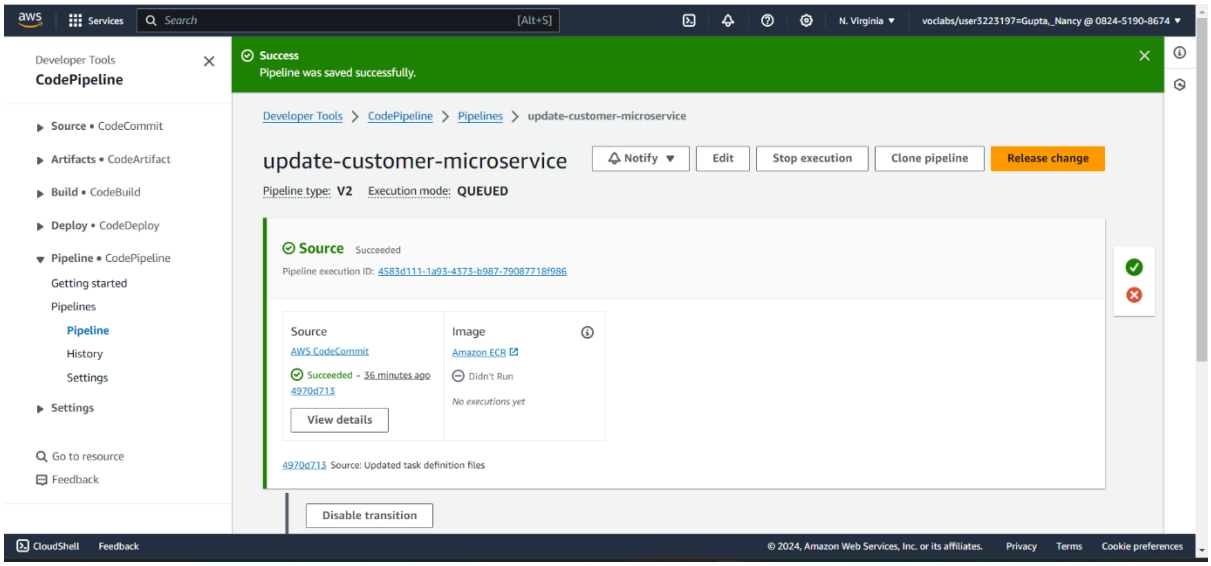
- Under **Dynamically update task definition image**, for **Input artifact with image details**, choose **image-customer**.
- For **Placeholder text in the task definition**, enter IMAGE1_NAME

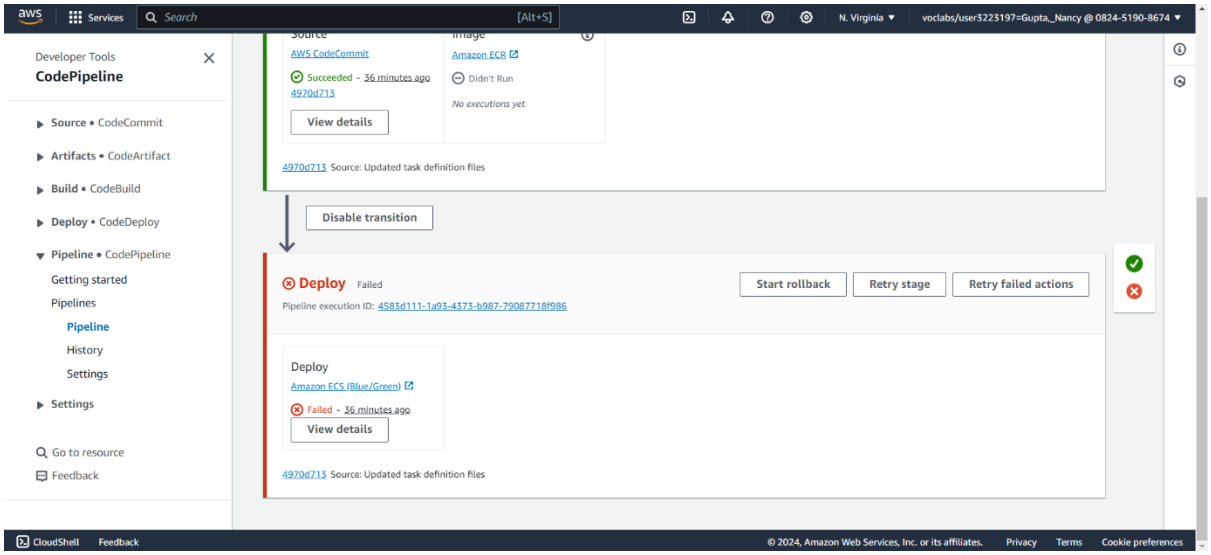


Analysis: Recall that in a previous phase, you entered the *IMAGE1_NAME* placeholder text in the *taskdef-customer.json* file before you pushed it to CodeCommit. In this current task, you configured the logic that will replace the placeholder text with the actual image name that the source phase of the CodePipeline returns.







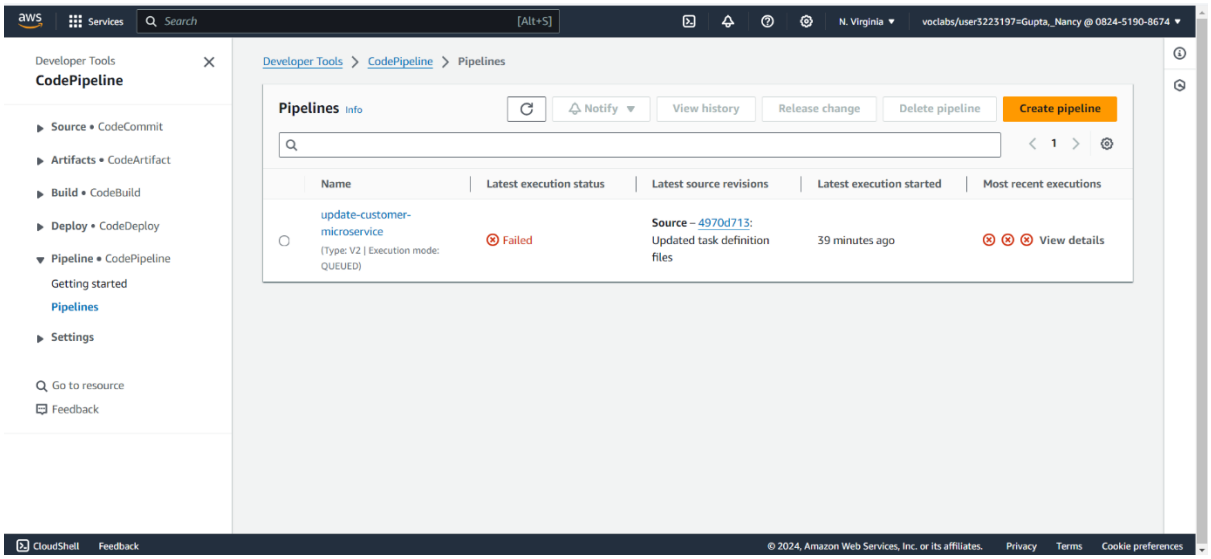


Task 8.3: Test the CI/CD pipeline for the *customer* microservice

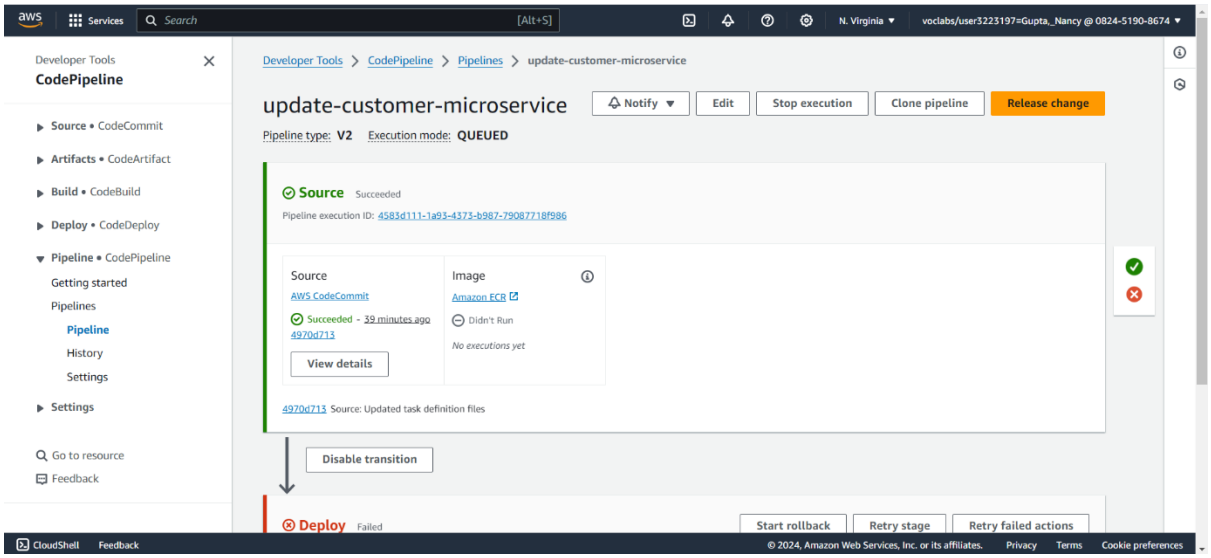
In this task, you will test that the CI/CD pipeline for the *customer* microservice functions as intended.

Important: If you are repeating this task, confirm that all target groups are still associated with the Application Load Balancer. The Reassociate Target Groups with Load Balancer section of the appendix provides more detail.

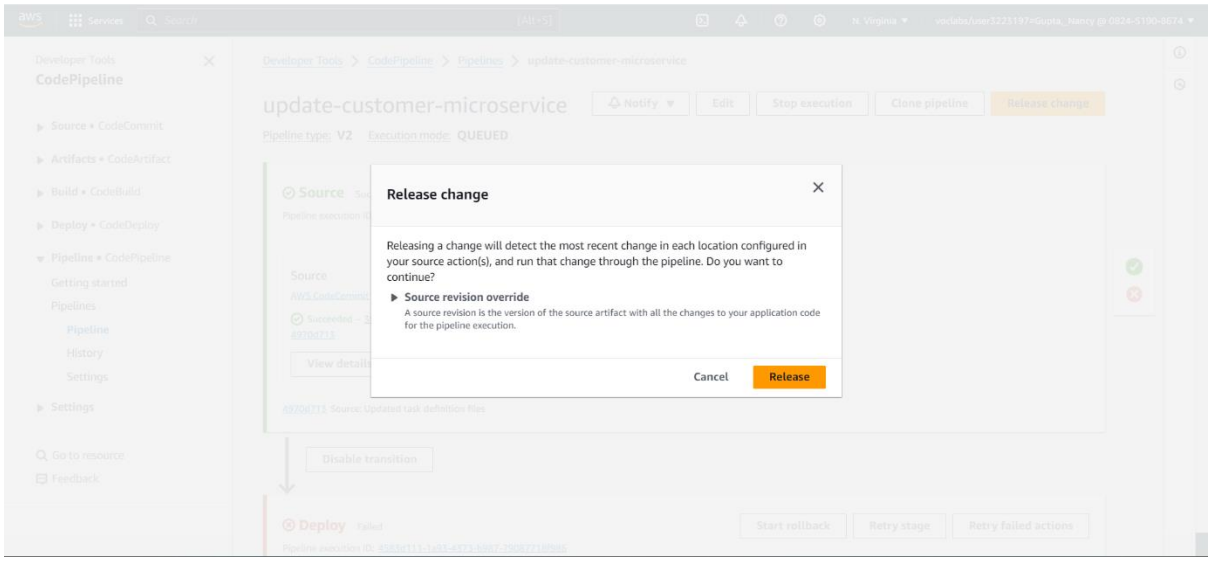
1. Launch a deployment of the *customer* microservice on Amazon ECS on Fargate.
    - Navigate to the CodePipeline console.



    - On the **Pipelines** page, choose the link for the pipeline that is named **update-customer-microservice**.

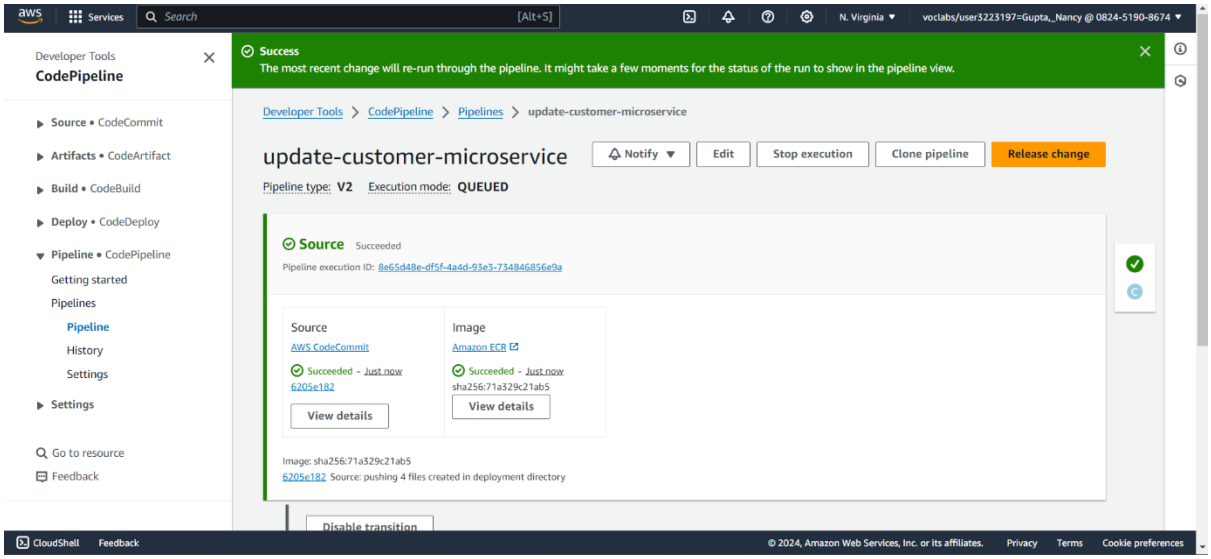


    - To force a test of the current pipeline settings, choose **Release change**, and then choose **Release**.



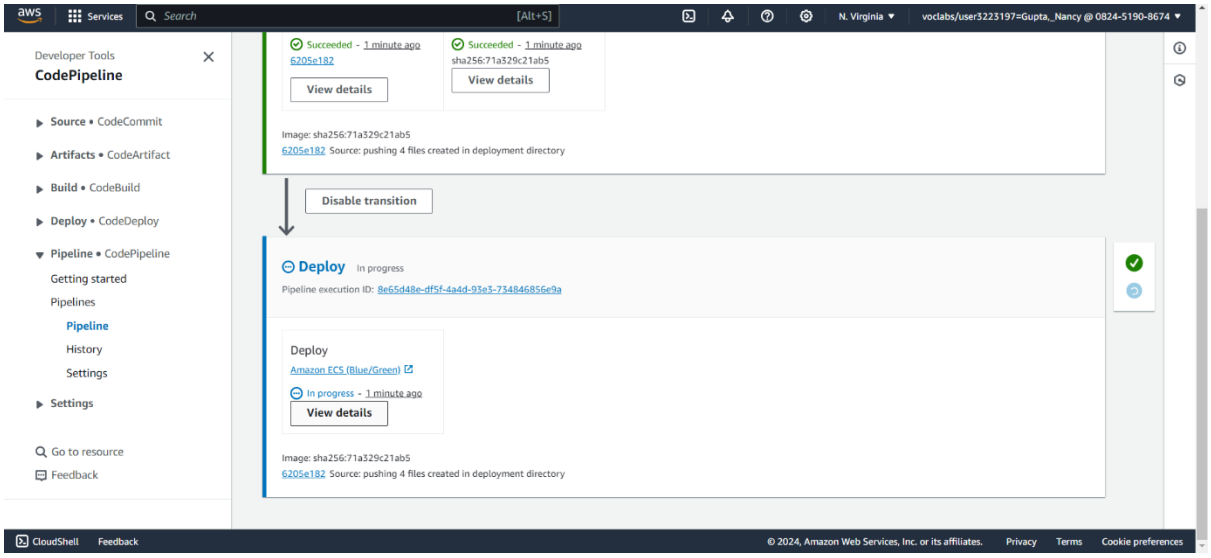
Note: By invoking the pipeline, you created a new revision of the task definition.

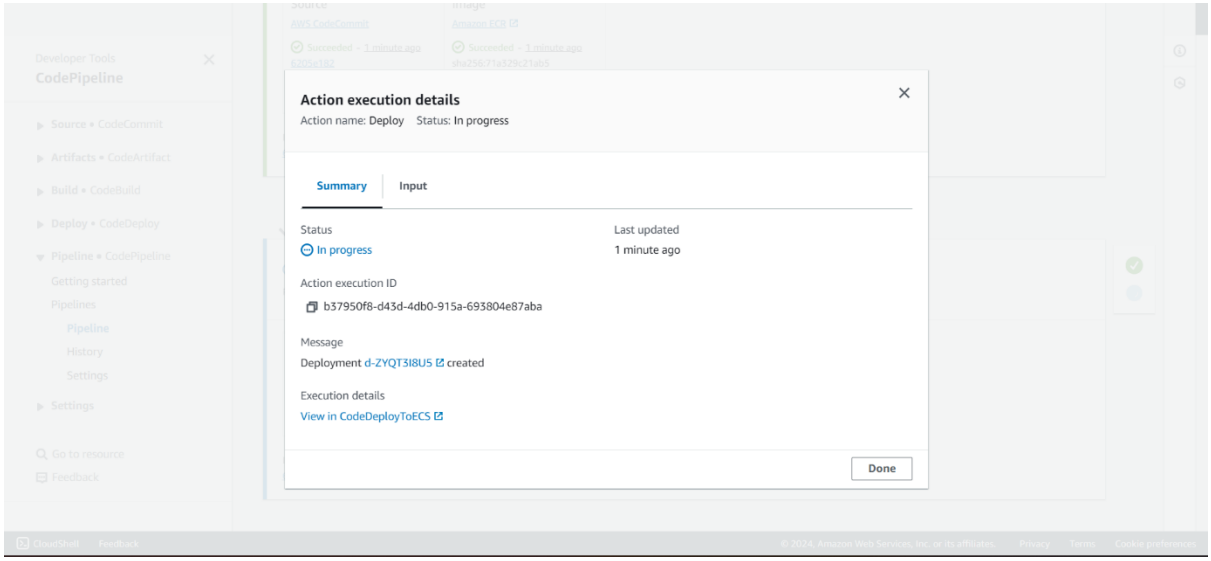
Wait for the two *Source* tasks to show a status of *Succeeded - just now*.

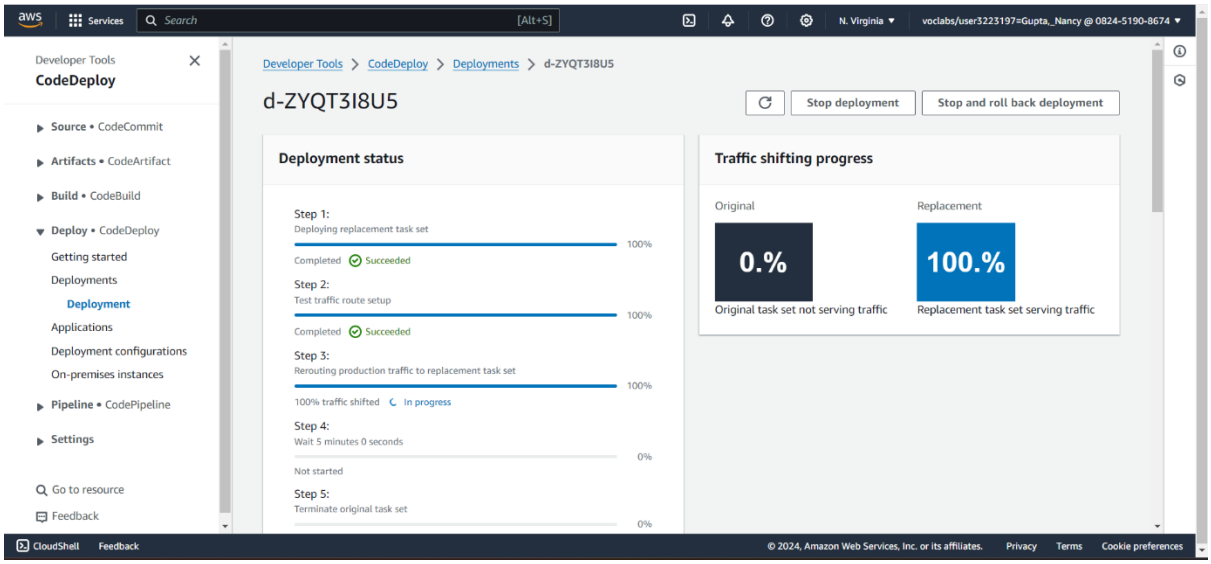


- In the **Deploy** section, wait for a **Details** link to appear, and then click the link.

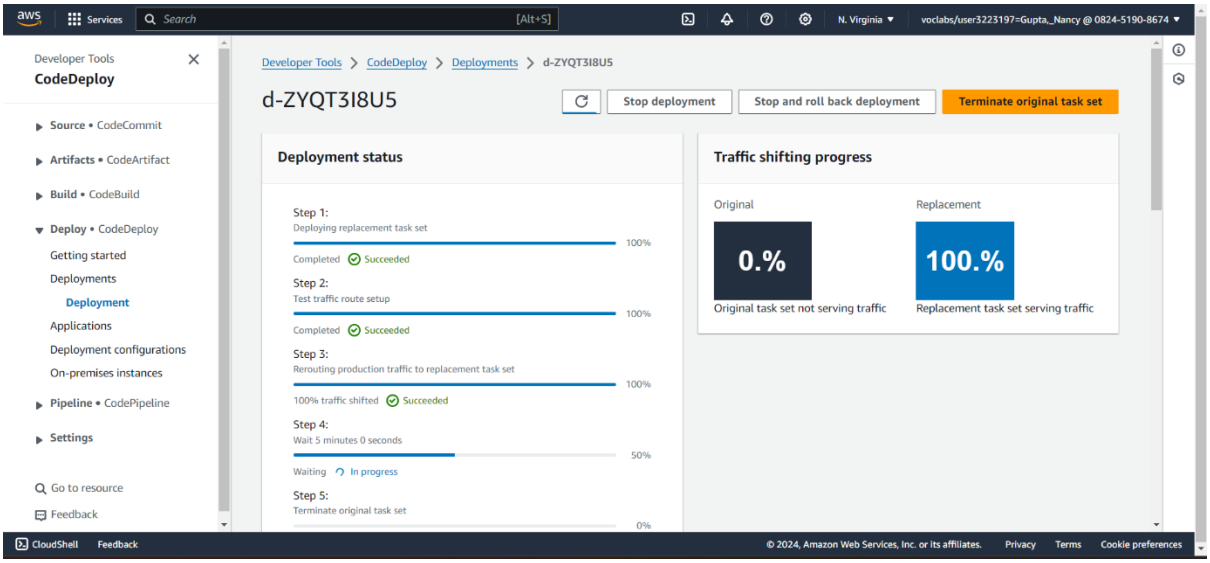
A CodeDeploy page opens in a new browser tab.



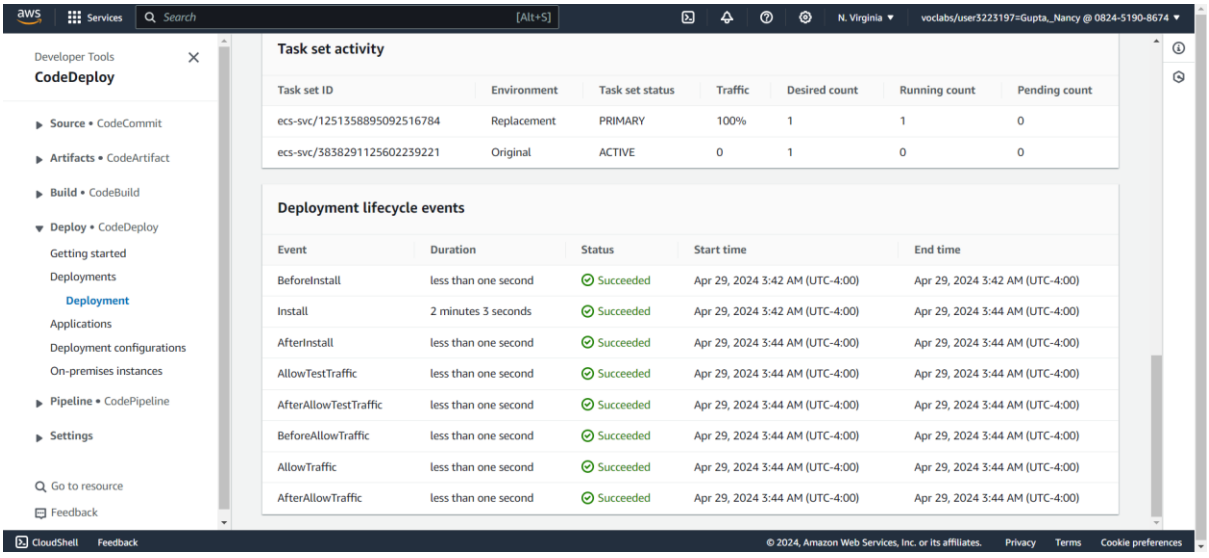




2. Observe the progress in CodeDeploy.

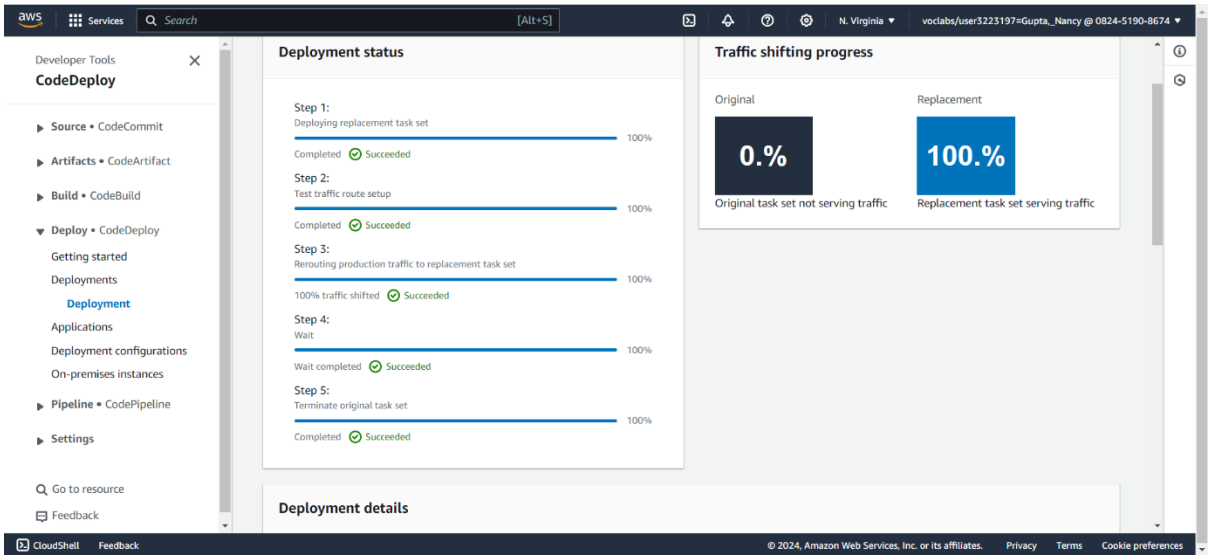
3. 

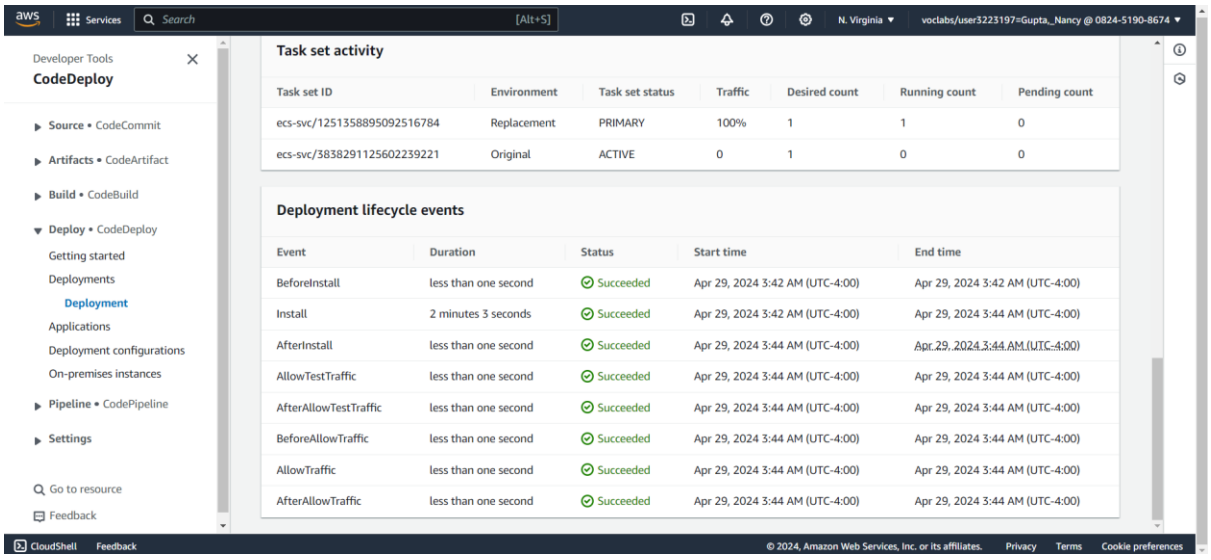
   - Scroll to the bottom of the page, and notice the **Deployment lifecycle events** section.



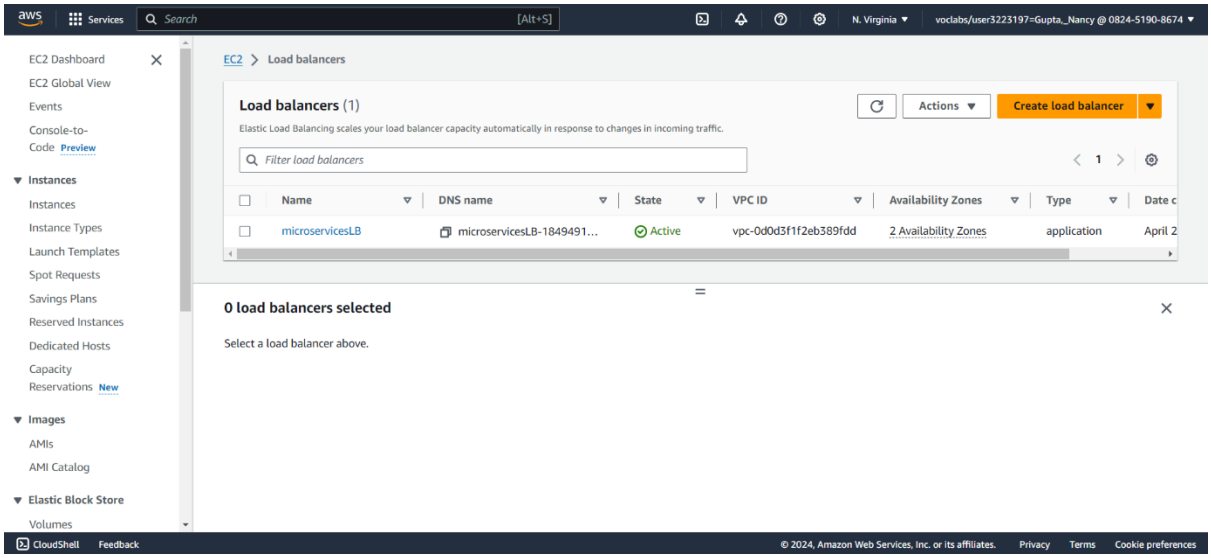
Tip: If you see a "Primary task group must be behind listener" error, refer to the Reassociate Target Groups with Load Balancer section in the appendix.

Within a few minutes, if everything was configured correctly, all of the deployment lifecycle events should succeed. Don't wait for that to happen—move to the next step. Keep this page open.

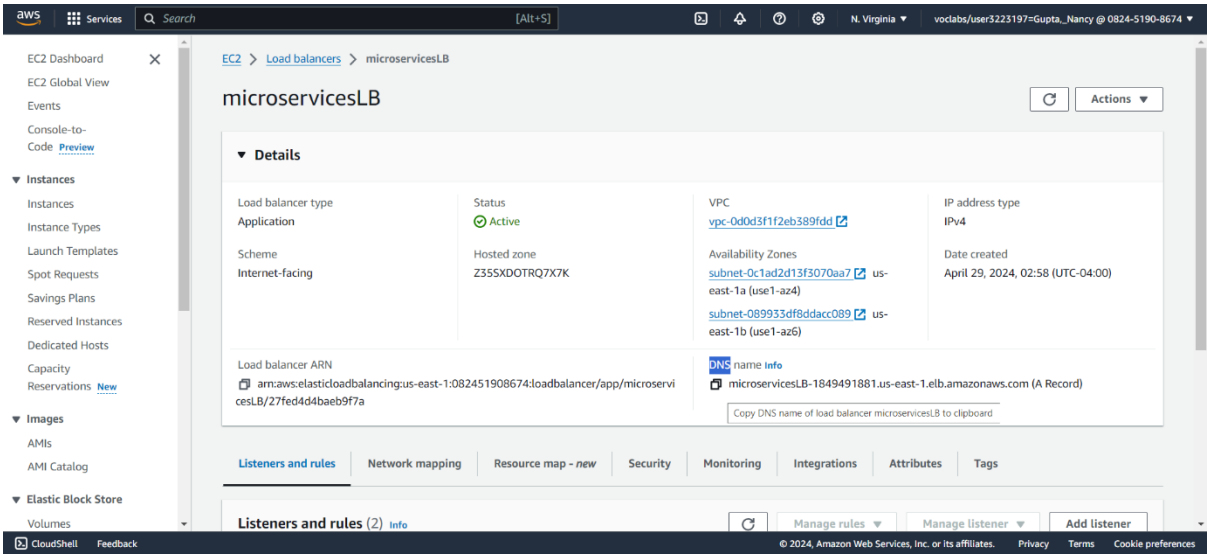


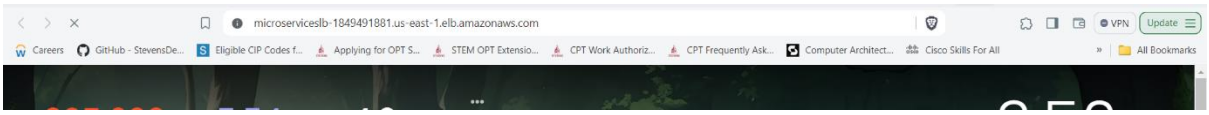


4. Load the *customer* microservice in a browser tab and test it.

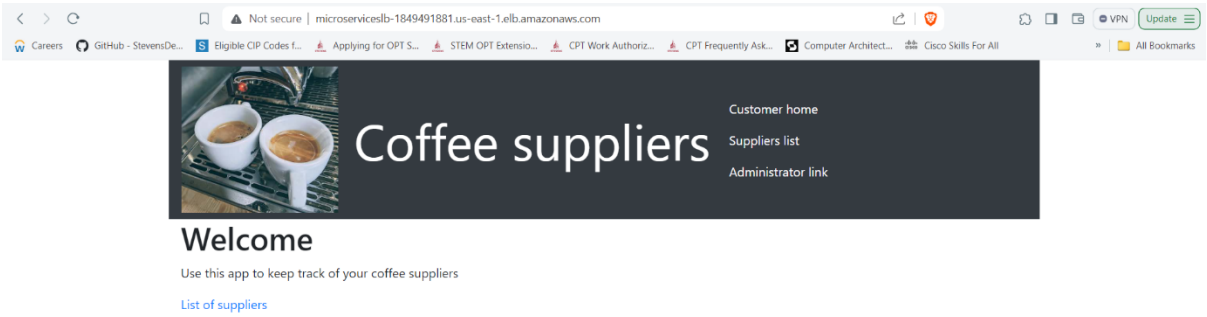


   - Locate the **DNS name** value of the **microservicesLB** load balancer, and paste it into a new browser tab.





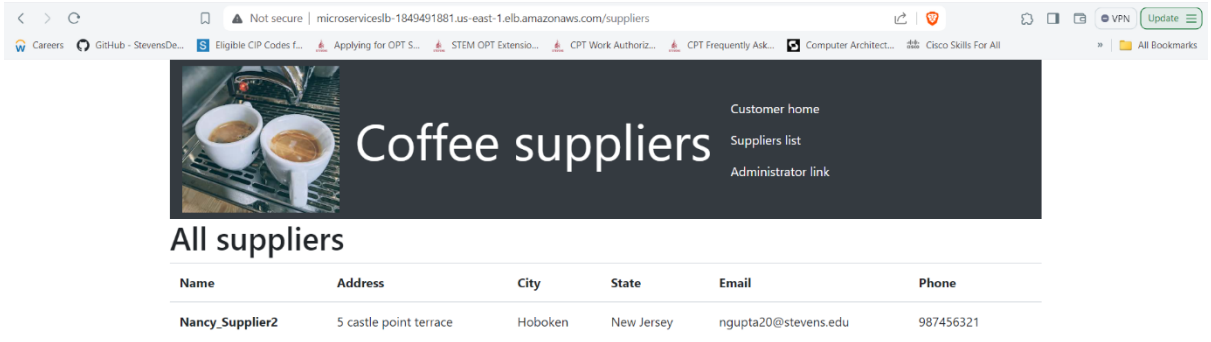
- The customer microservice loads. If it doesn't load, add :8080 to the end of the URL and try again.



Analysis: Recall that your load balancer has two listeners: one on port 80 and another on port 8080. Port 8080 is where the replacement task set will run for the first 5 minutes. Therefore, if you load the :80 URL within the first 5 minutes, the customer microservice page might not load, but you should already see the page at 8080. Then, after 5 minutes, you should see that the microservice is available at both ports.

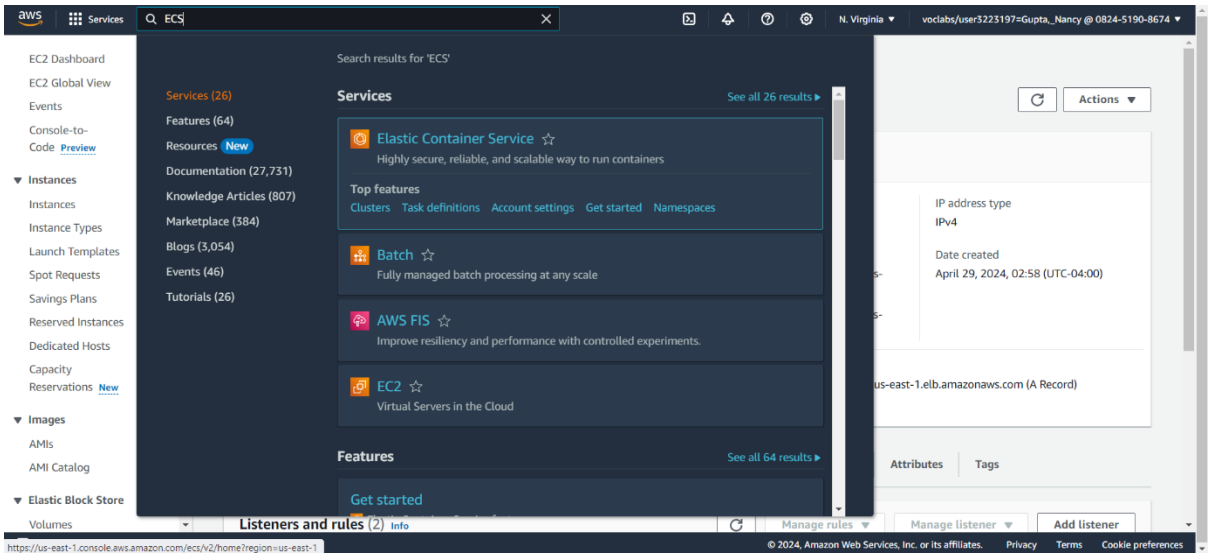
- In the café web application, choose **List of suppliers** or **Suppliers list**.

The suppliers page loads. It should *not* have the edit or add supplier buttons because it is a customer page.

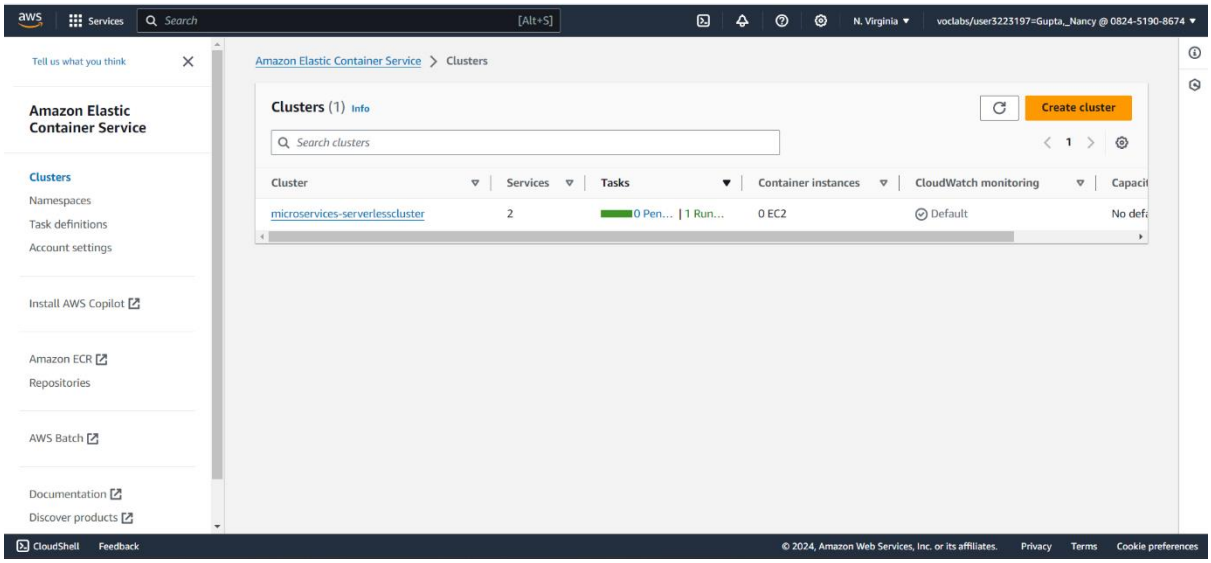


5. Observe the running tasks in the Amazon ECS console.

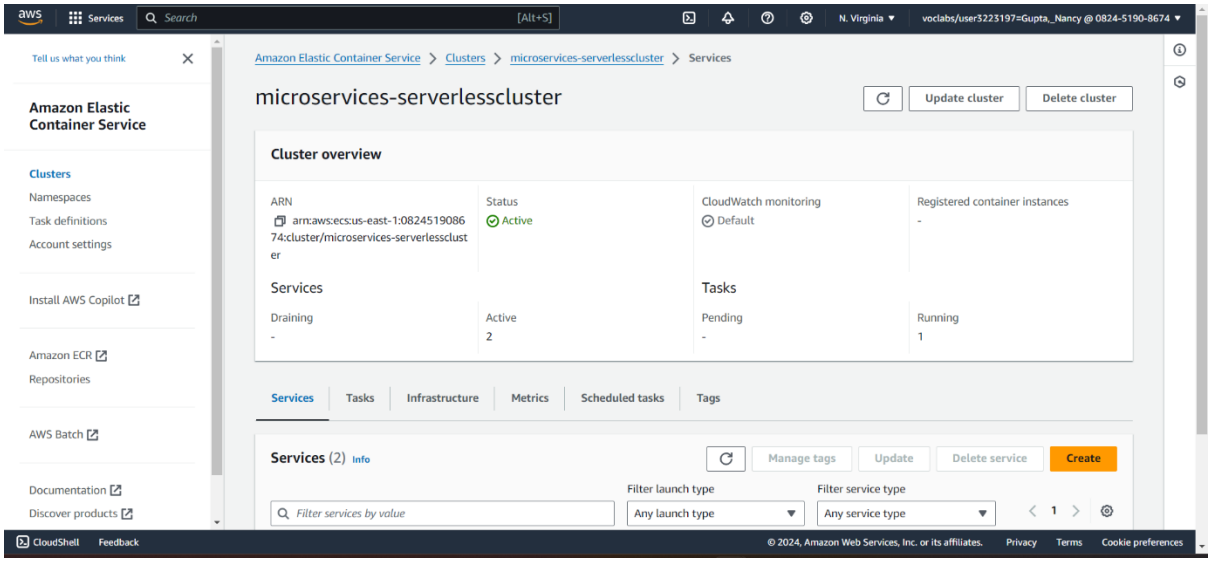
   - Navigate to the Amazon ECS console.

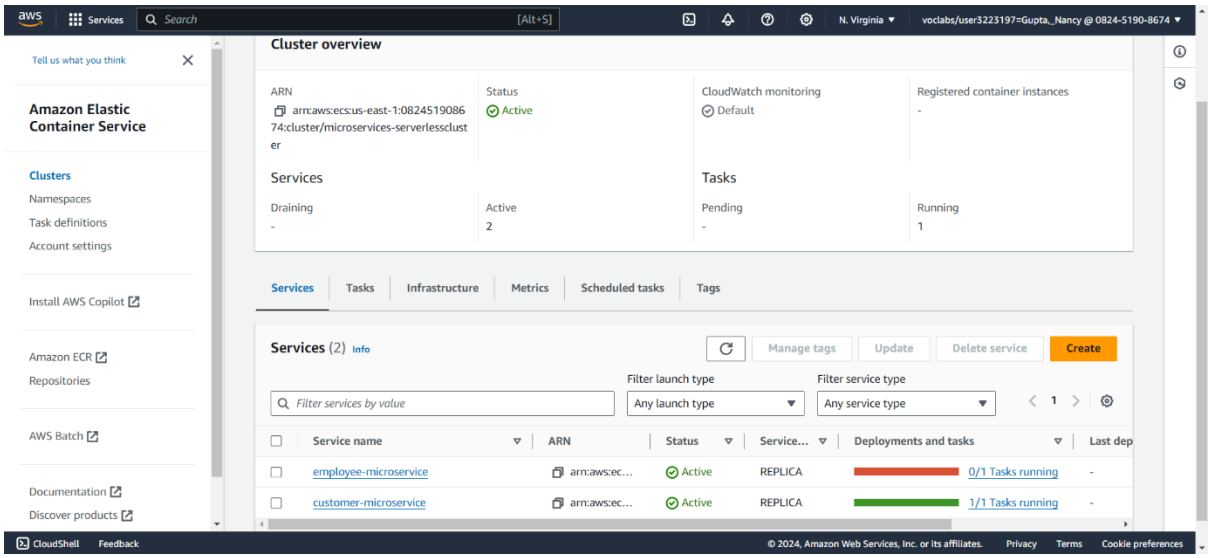


   - In the clusters list, choose the link for **microservices-serverlesscluster**.



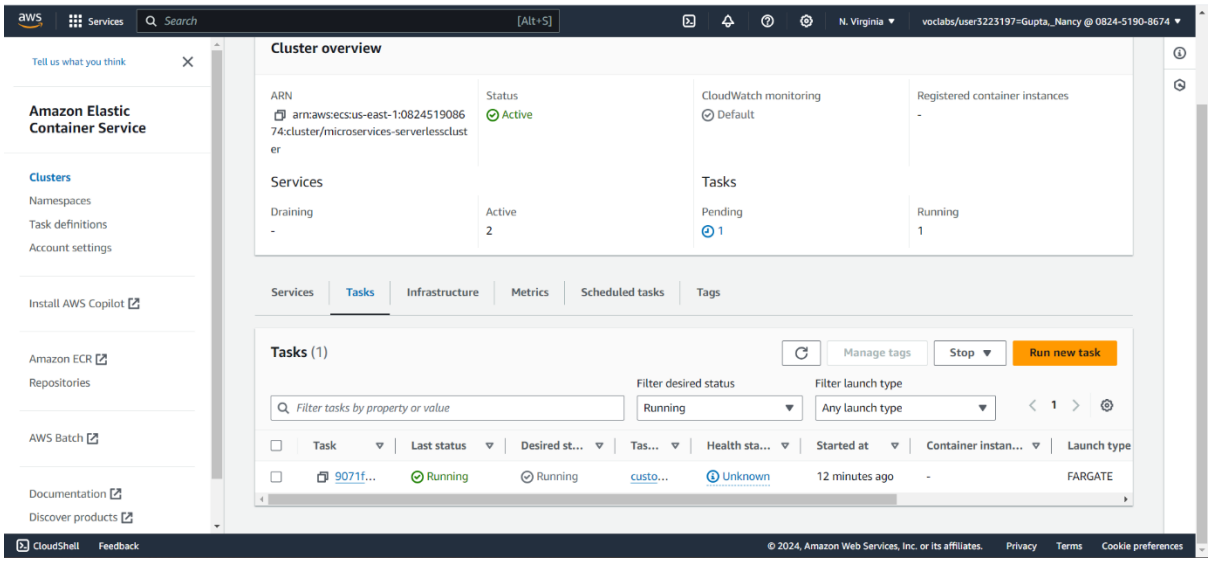
On the **Services** tab, notice that the *customer-microservice* service appears. The **Deployments and tasks** status will change as the blue/green deployment advances through its lifecycle events.





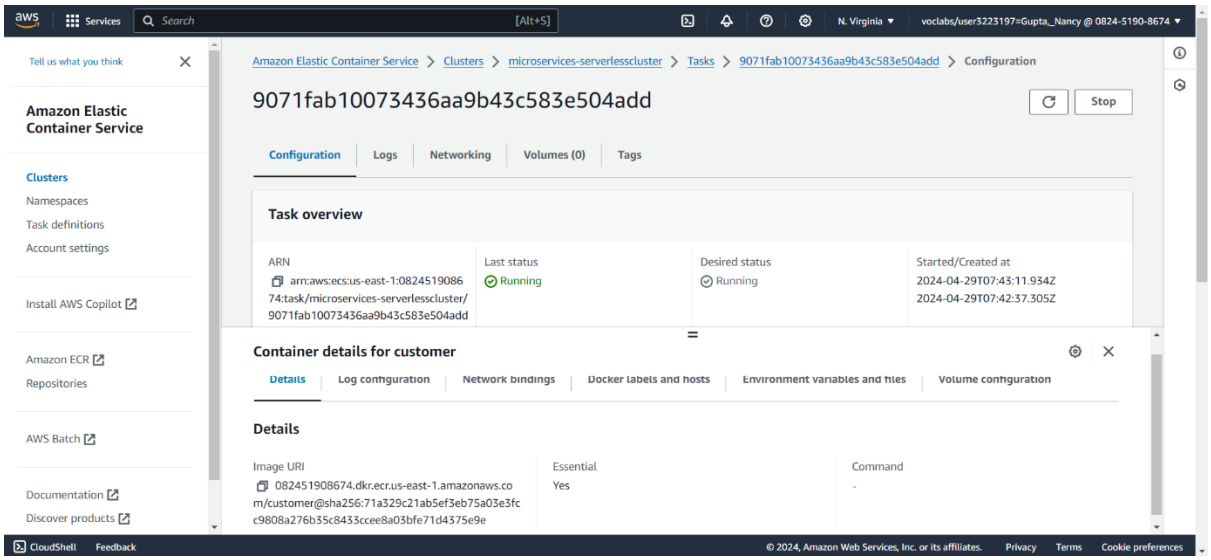
- Choose the **Tasks** tab.

Here you can see the actual tasks that are running. You might have more than one task running per service that you defined.



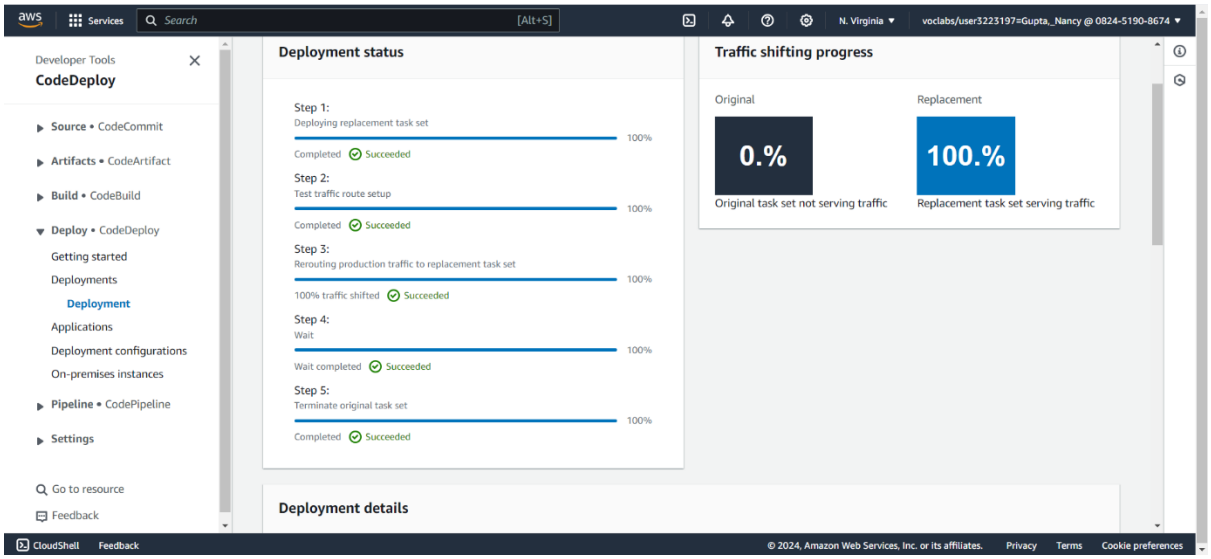
- Choose the link for one of the listed tasks. You might only have one.

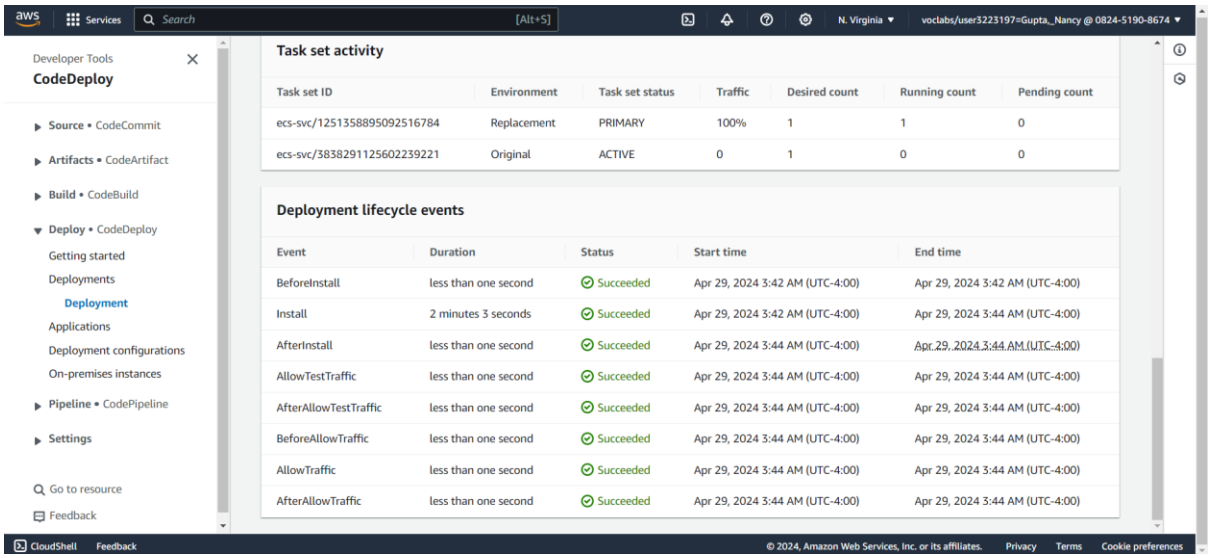
Here you can see the actual container details and the configuration information, such as the IP addresses that are associated with the running container.



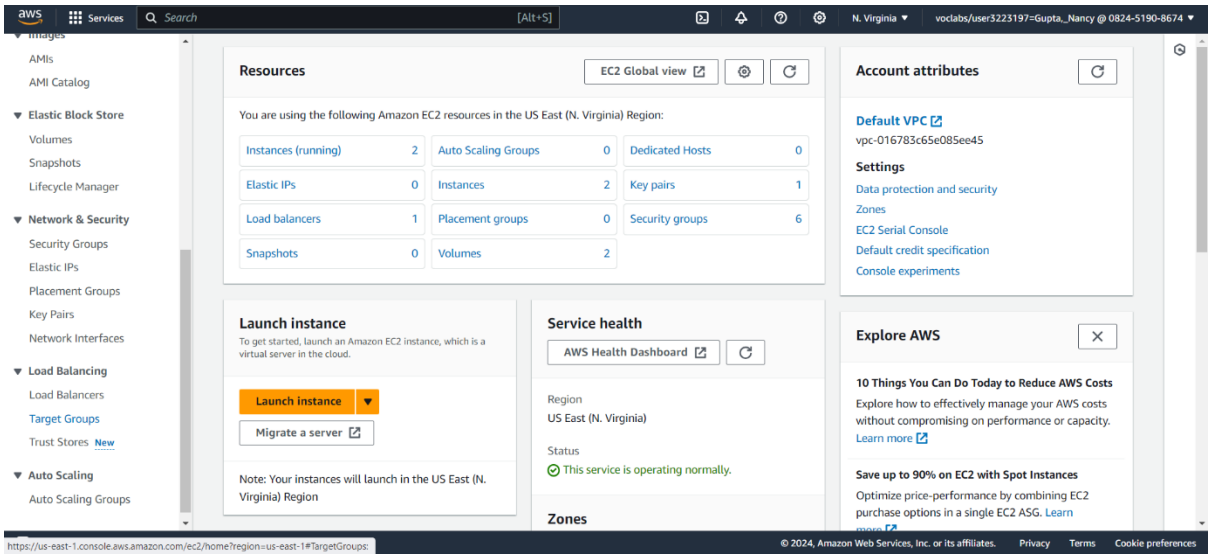
6. Return to the CodeDeploy page that is open in another browser tab.

You should now see that all five steps of the deployment succeeded and the replacement task set is now serving traffic.

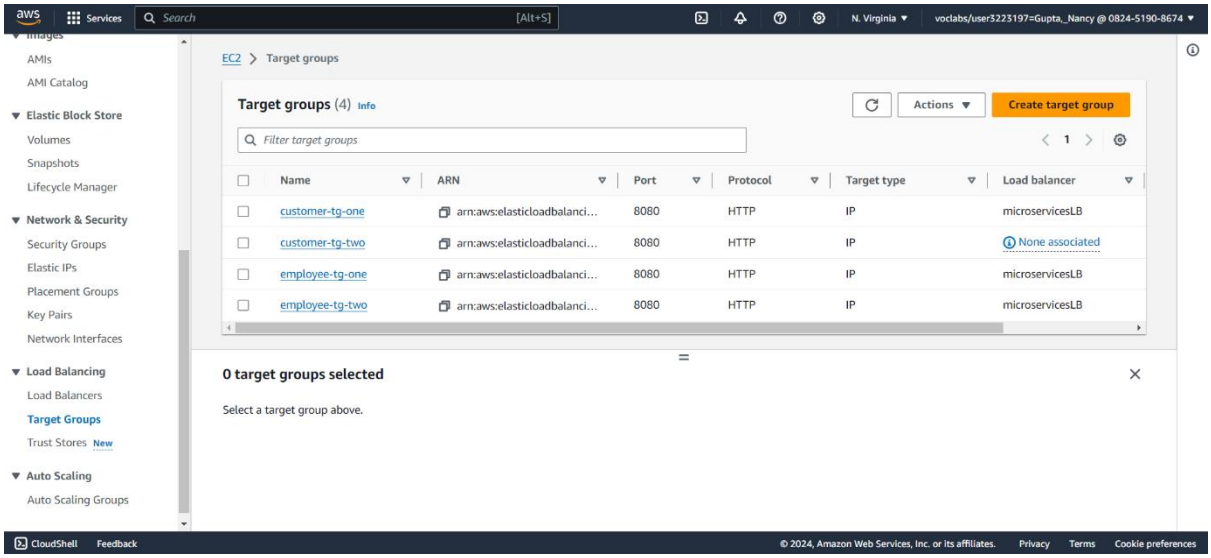




7. Observe the load balancer and target group settings.
    - In the Amazon EC2 console, choose **Target Groups**.

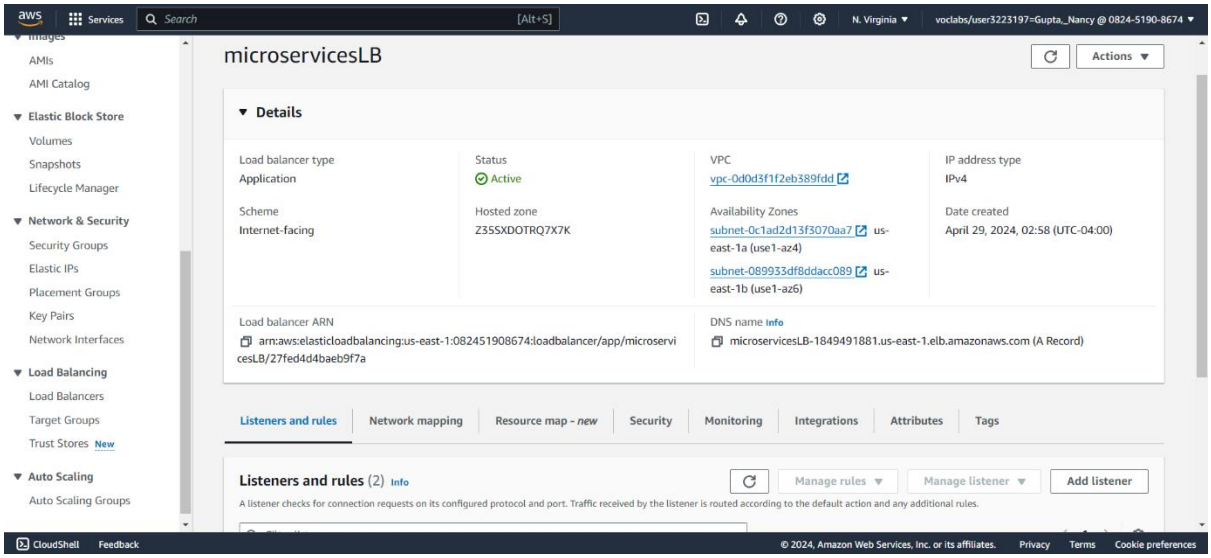


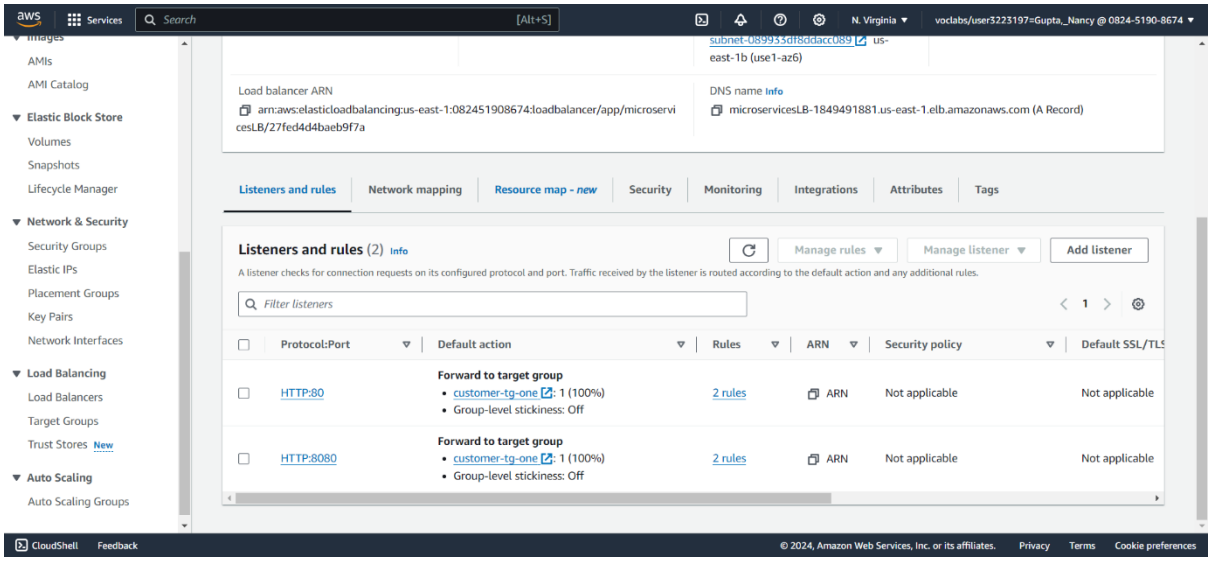
You might notice that the *customer-tg-two* target group is no longer associated with the load balancer. This is because CodeDeploy is managing the load balancer listener rules and might have determined that some of the target groups are no longer needed.

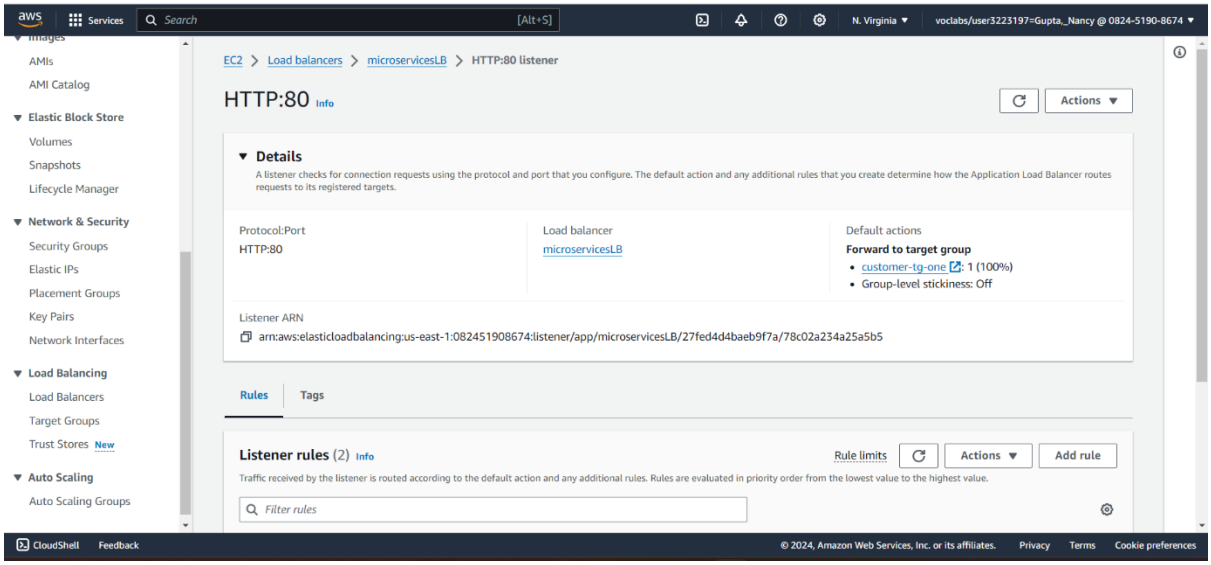


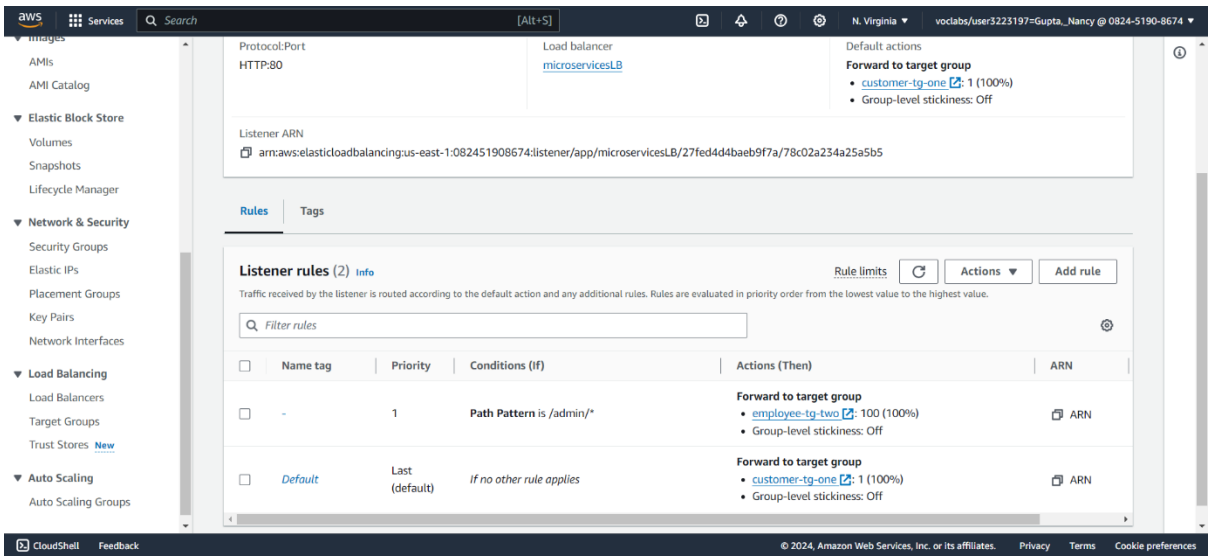
- Observe the HTTP:80 listener rules.

The default rule has changed here. The default "if no other rule applies" rule previously pointed to *customer-tg-two*, but now it points to *customer-tg-one*. This is because CodeDeploy actively managed your Application Load Balancer.



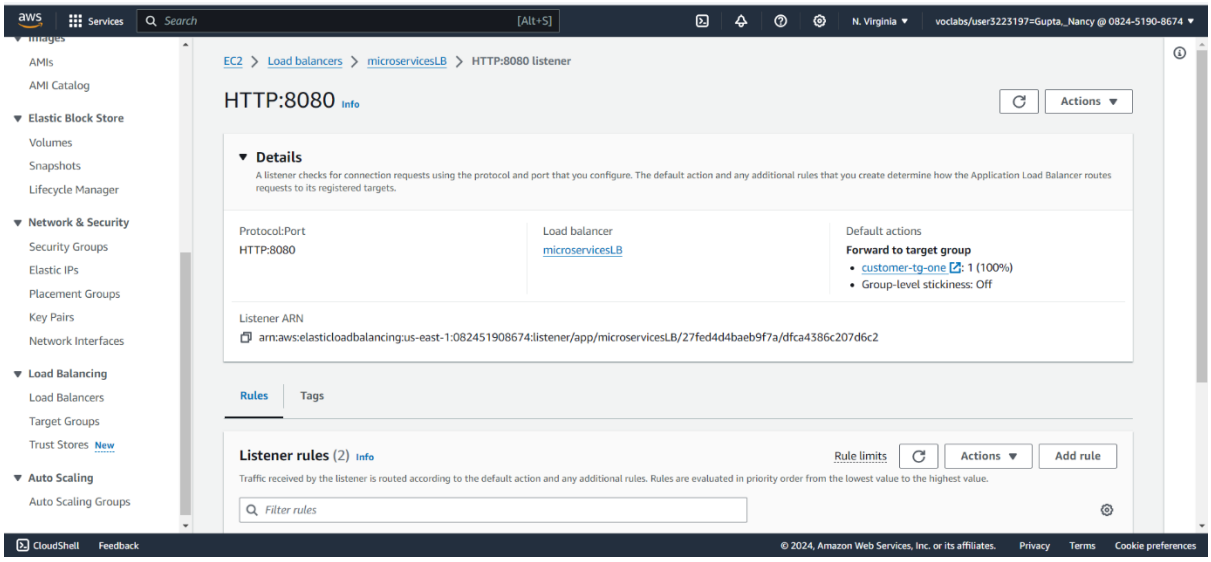


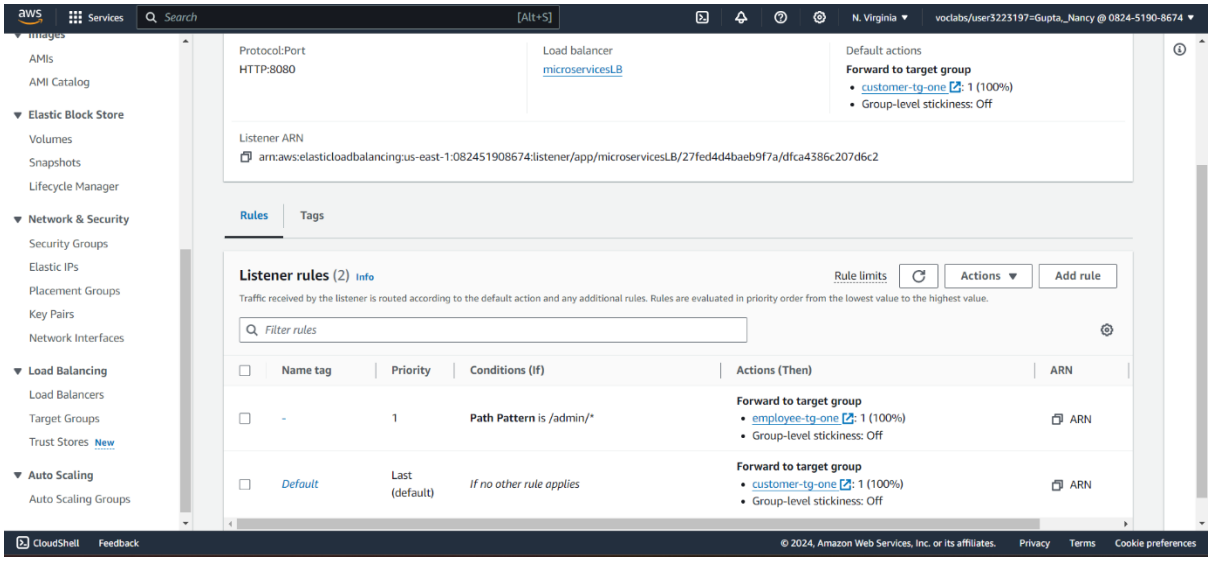




- Observe the HTTP:8080 listener rules.

The two rules still forward to the "one" target groups.

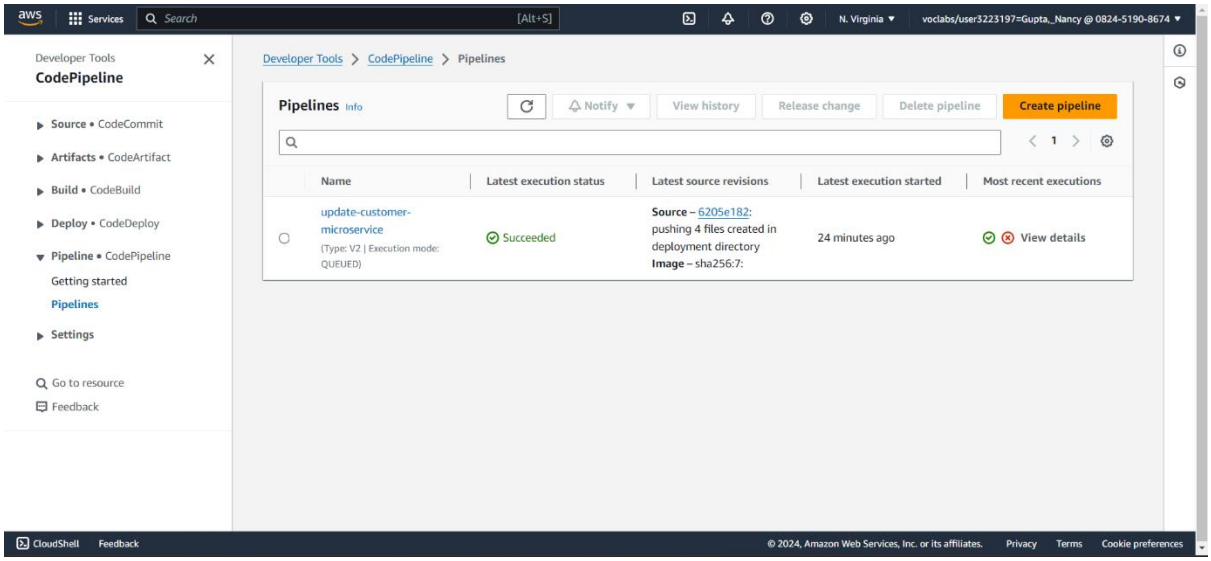




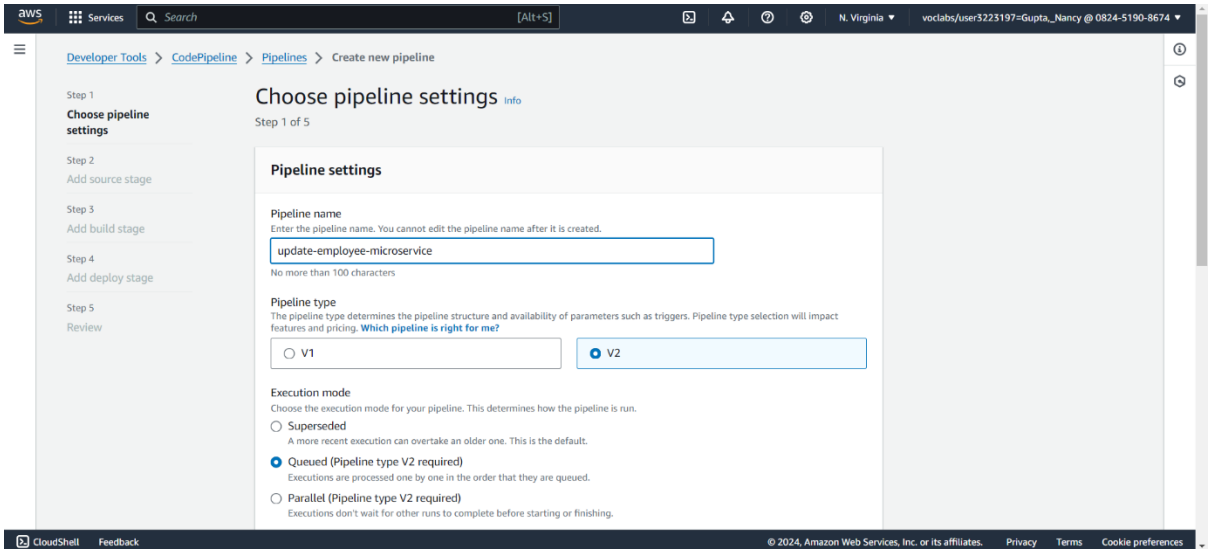
Congratulations! You successfully deployed one of the two microservices to Amazon ECS on Fargate by using a CI/CD pipeline.

Task 8.4: Create a pipeline for the *employee* microservice

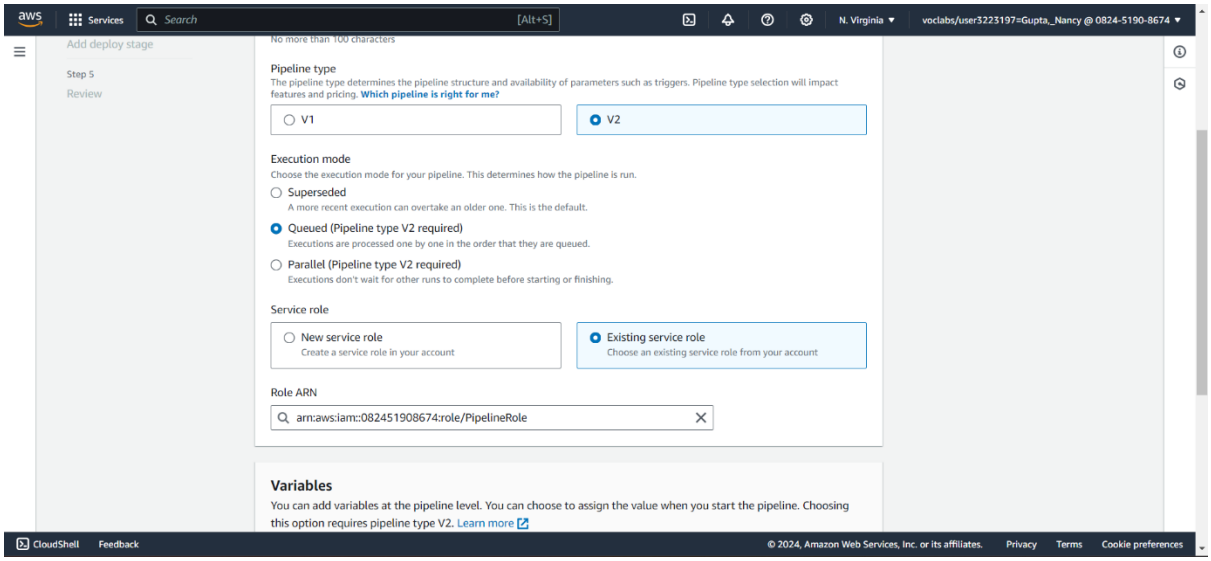
In this task, you will create the pipeline for the *employee* microservice.

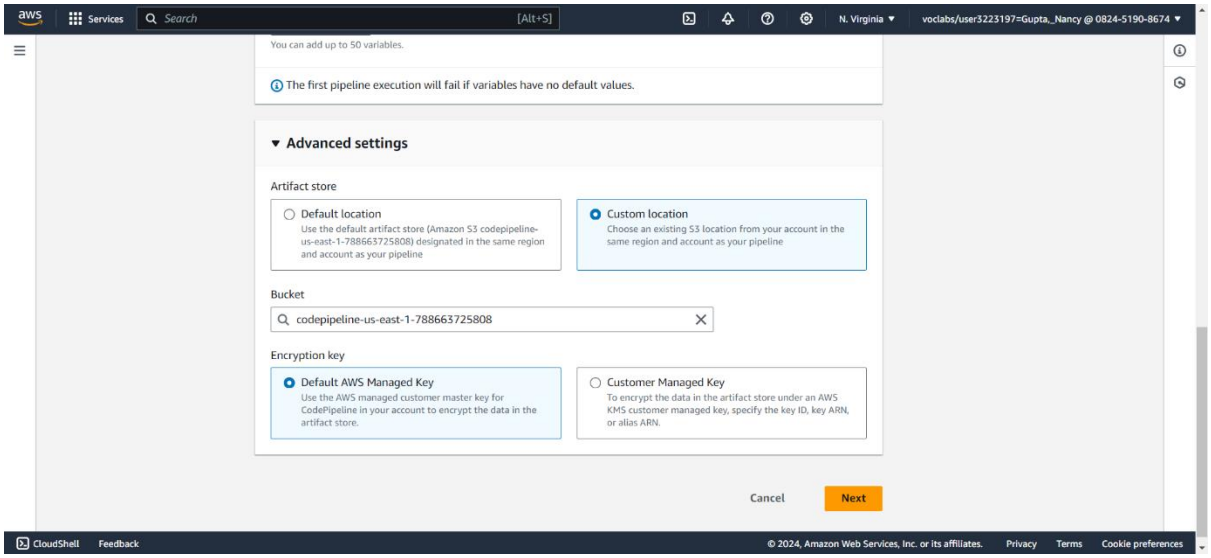


1. Create a pipeline for the *employee* microservice with the following specifications:
    - Pipeline name: update-employee-microservice

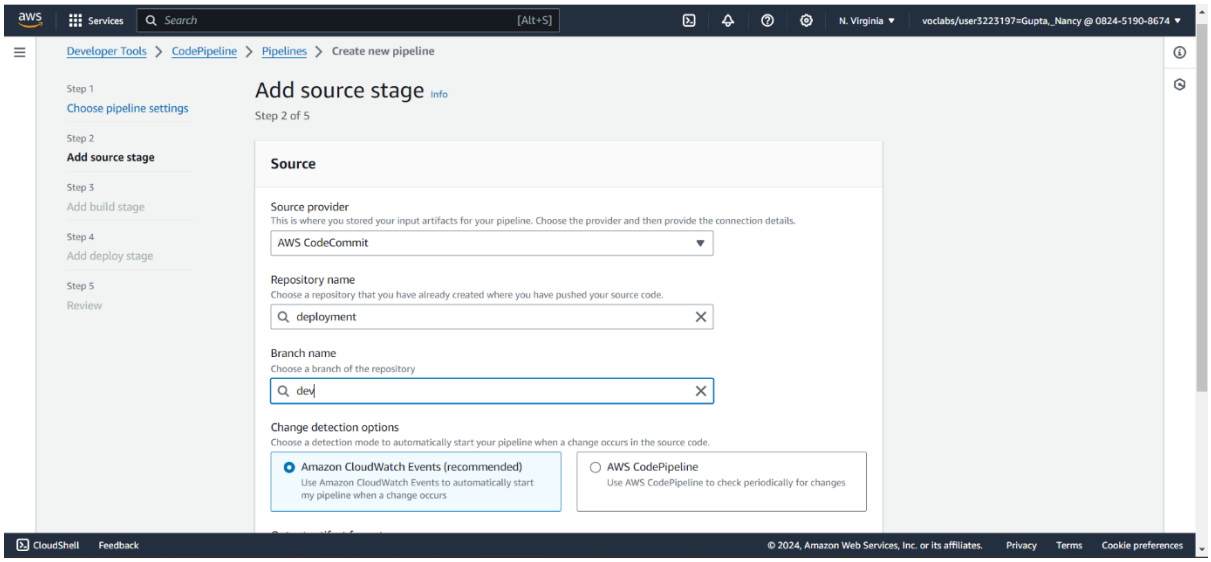


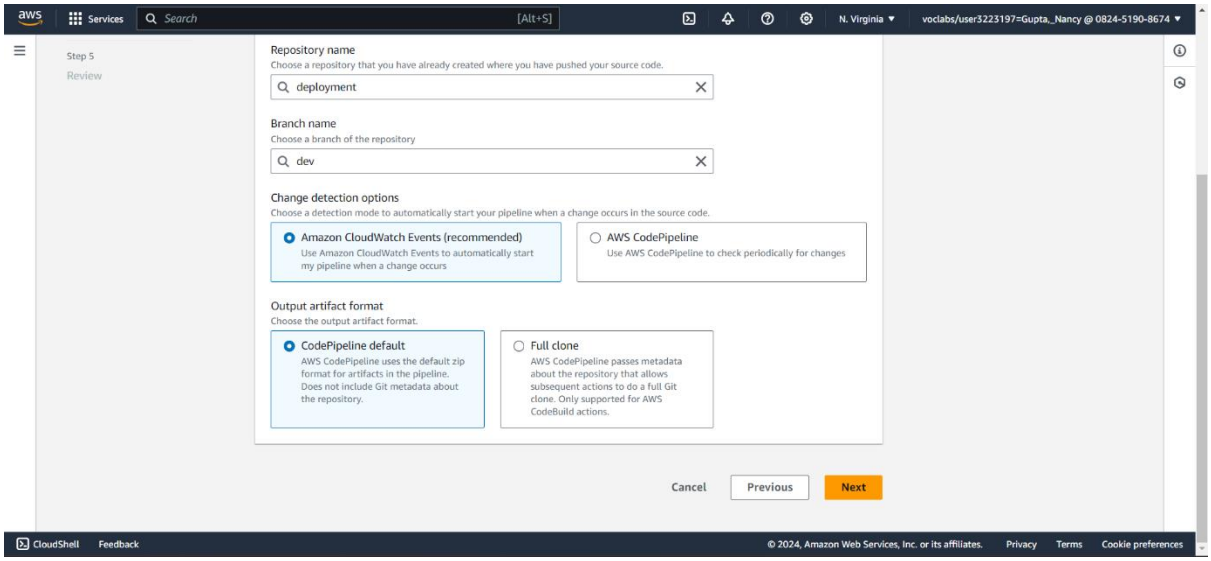
    - Role ARN: **PipelineRole**



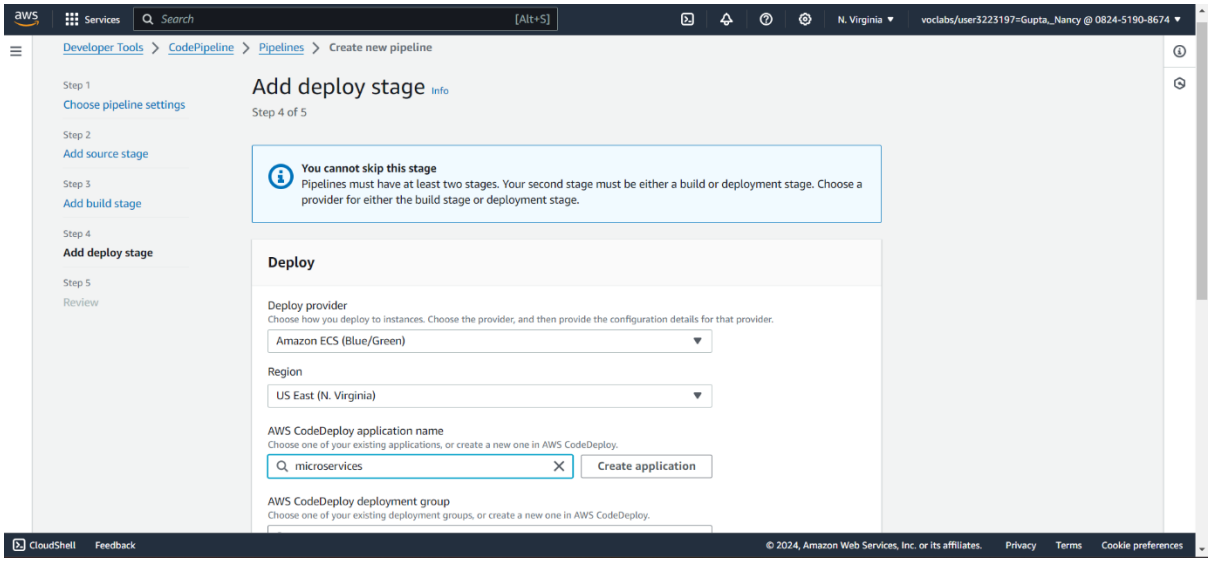


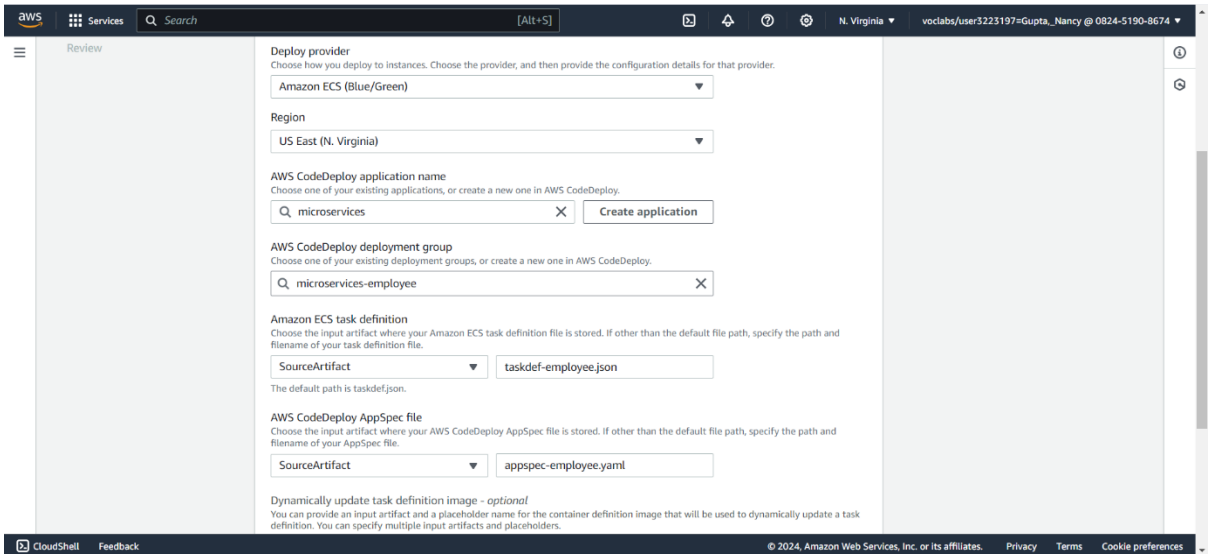
- Source provider: **AWS CodeCommit**
  - Repository name: **deployment**
  - Branch name: **dev**

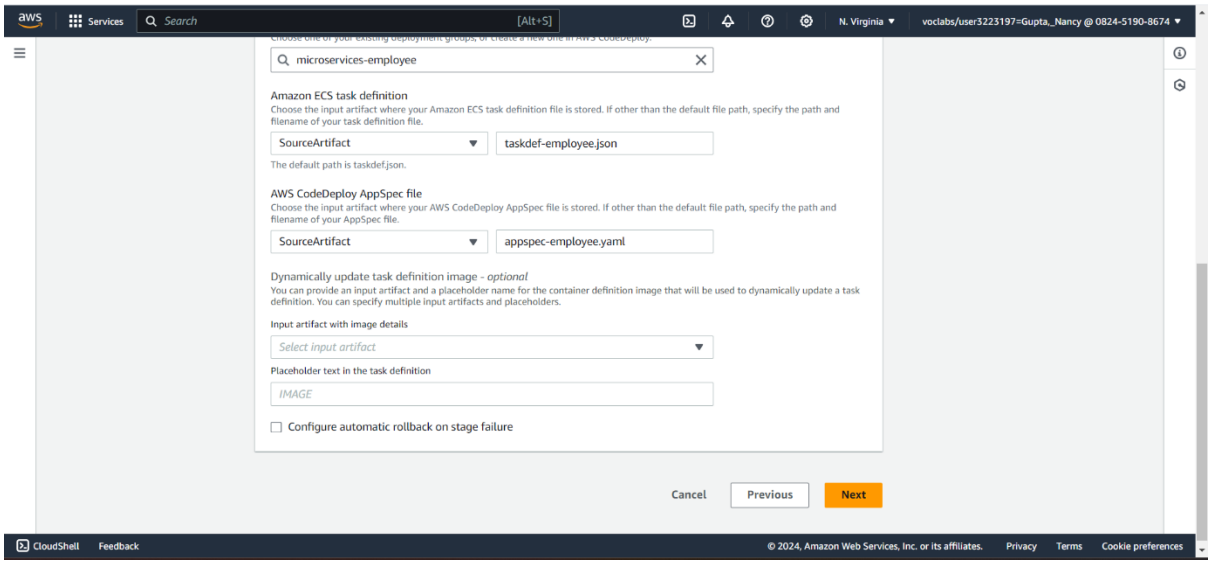


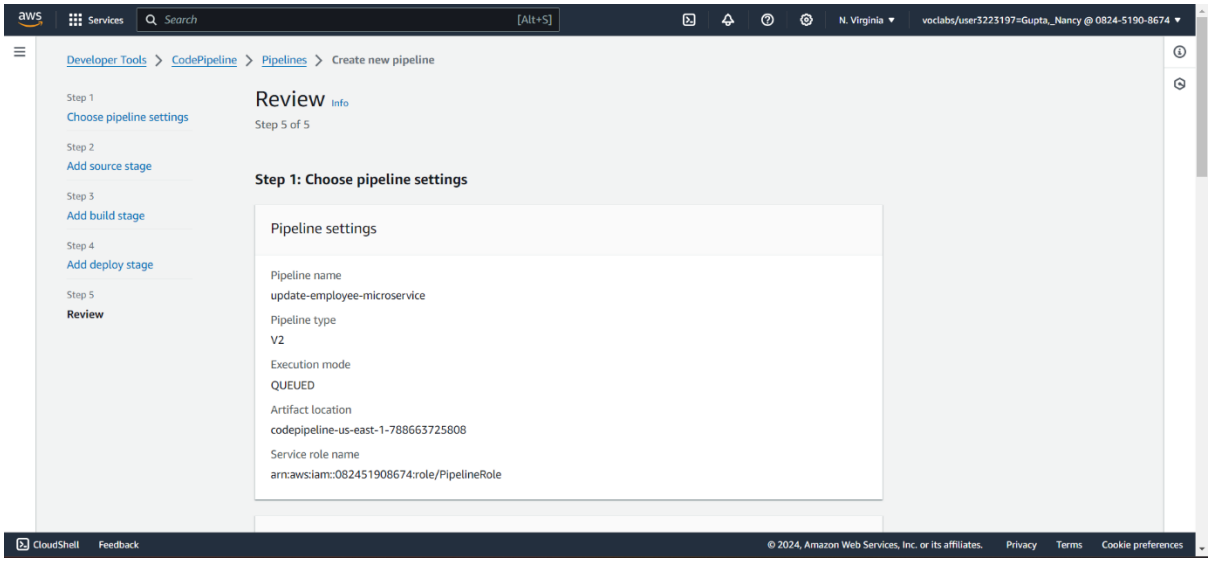


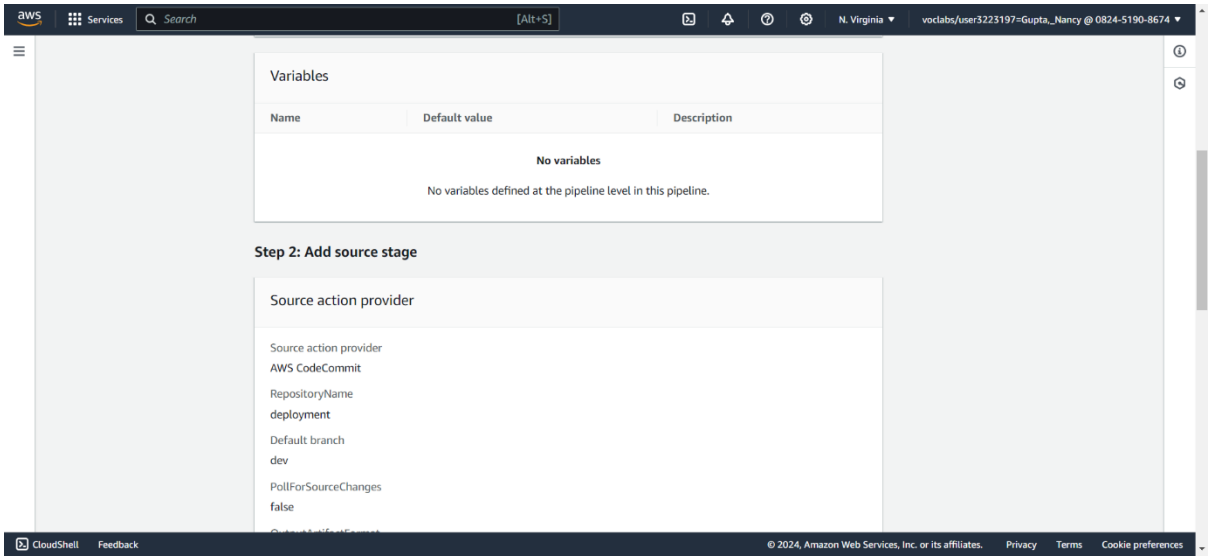
- Deploy provider: **Amazon ECS (Blue/Green)**
- AWS CodeDeploy application name: **microservices**
- AWS CodeDeploy deployment group: **microservices-employee**
- Amazon ECS task definition: **SourceArtifact**
  - Path: taskdef-employee.json
  - AWS CodeDeploy AppSpec file: **SourceArtifact**
  - Path: appspec-employee.yaml

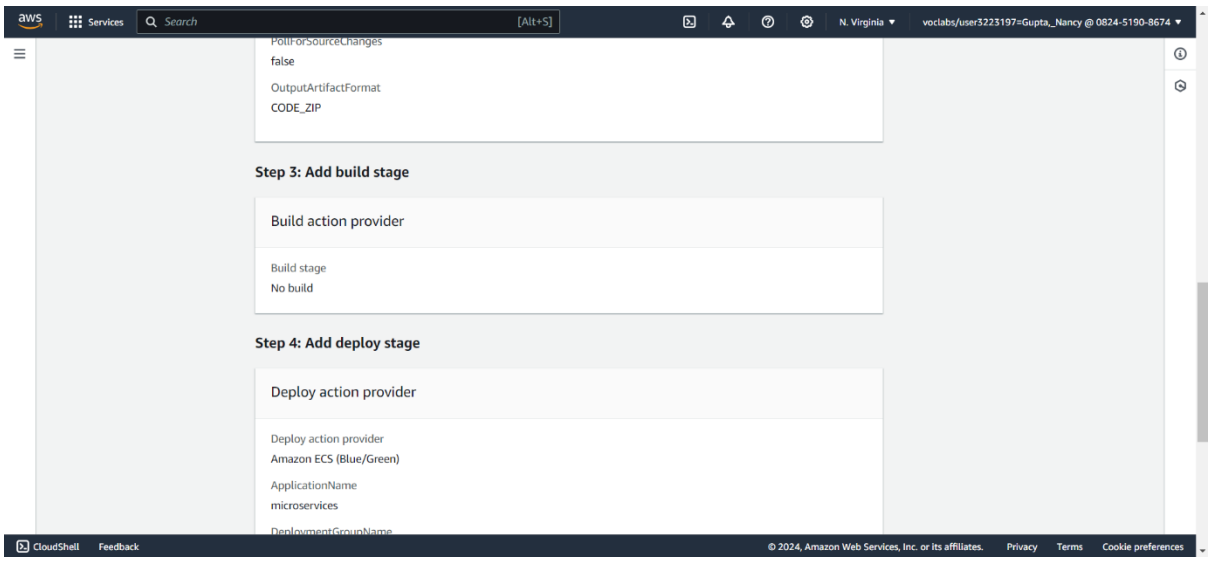

Services
Search
[Alt+S]
N. Virginia
voclabs/user3223197=Gupta,_Nancy @ 0824-5190-8674
Developer Tools
CodePipeline
Pipelines
Create new pipeline
Step 1
Choose pipeline settings
Step 2
Add source stage
Step 3
Add build stage
Step 4
Add deploy stage
Step 5
Review
Add deploy stage
Info
Step 4 of 5
You cannot skip this stage
Pipelines must have at least two stages. Your second stage must be either a build or deployment stage. Choose a provider for either the build stage or deployment stage.
Deploy
Deploy provider
Choose how you deploy to instances. Choose the provider, and then provide the configuration details for that provider.
Amazon ECS (Blue/Green)
Region
US East (N. Virginia)
AWS CodeDeploy application name
Choose one of your existing applications, or create a new one in AWS CodeDeploy.
microservices
Create application
AWS CodeDeploy deployment group
Choose one of your existing deployment groups, or create a new one in AWS CodeDeploy.
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

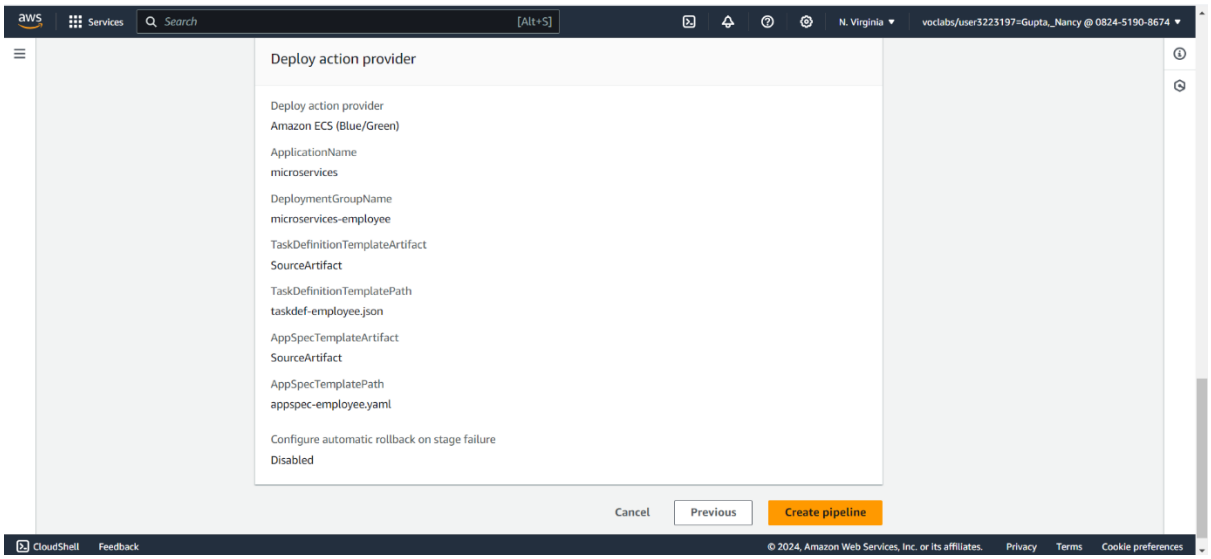

Services
Search
[Alt+S]
N. Virginia
voclabs/user3223197=Gupta,_Nancy @ 0824-5190-8674
Review
Deploy provider
Choose how you deploy to instances. Choose the provider, and then provide the configuration details for that provider.
Amazon ECS (Blue/Green)
Region
US East (N. Virginia)
AWS CodeDeploy application name
Choose one of your existing applications, or create a new one in AWS CodeDeploy.
microservices
Create application
AWS CodeDeploy deployment group
Choose one of your existing deployment groups, or create a new one in AWS CodeDeploy.
microservices-employee
Amazon ECS task definition
Choose the input artifact where your Amazon ECS task definition file is stored. If other than the default file path, specify the path and filename of your task definition file.
SourceArtifact
taskdef-employee.json
The default path is taskdef.json.
AWS CodeDeploy AppSpec file
Choose the input artifact where your AWS CodeDeploy AppSpec file is stored. If other than the default file path, specify the path and filename of your AppSpec file.
SourceArtifact
appspec-employee.yaml
Dynamically update task definition image - optional
You can provide an input artifact and a placeholder name for the container definition image that will be used to dynamically update a task definition. You can specify multiple input artifacts and placeholders.
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

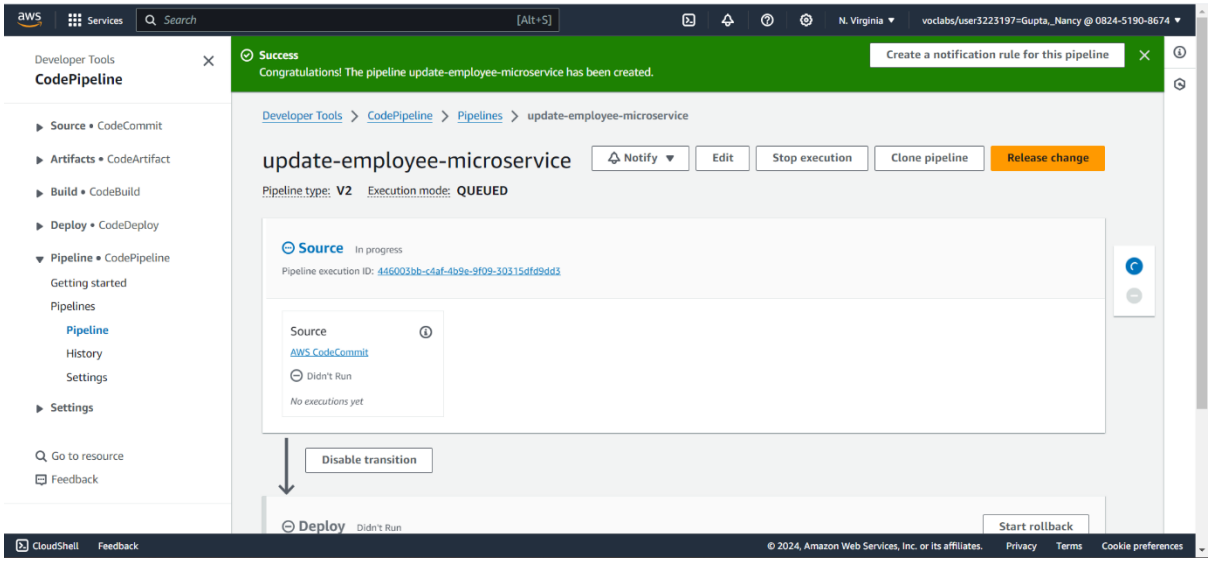

Services
Search
[Alt+S]
N. Virginia
voclabs/user3223197=Gupta,_Nancy @ 0824-5190-8674
microservices-employee
Amazon ECS task definition
Choose the input artifact where your Amazon ECS task definition file is stored. If other than the default file path, specify the path and filename of your task definition file.
SourceArtifact
taskdef-employee.json
The default path is taskdef.json.
AWS CodeDeploy AppSpec file
Choose the input artifact where your AWS CodeDeploy AppSpec file is stored. If other than the default file path, specify the path and filename of your AppSpec file.
SourceArtifact
appspec-employee.yaml
Dynamically update task definition image - optional
You can provide an input artifact and a placeholder name for the container definition image that will be used to dynamically update a task definition. You can specify multiple input artifacts and placeholders.
Input artifact with image details
Select input artifact
Placeholder text in the task definition
IMAGE
Configure automatic rollback on stage failure
Cancel
Previous
Next
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

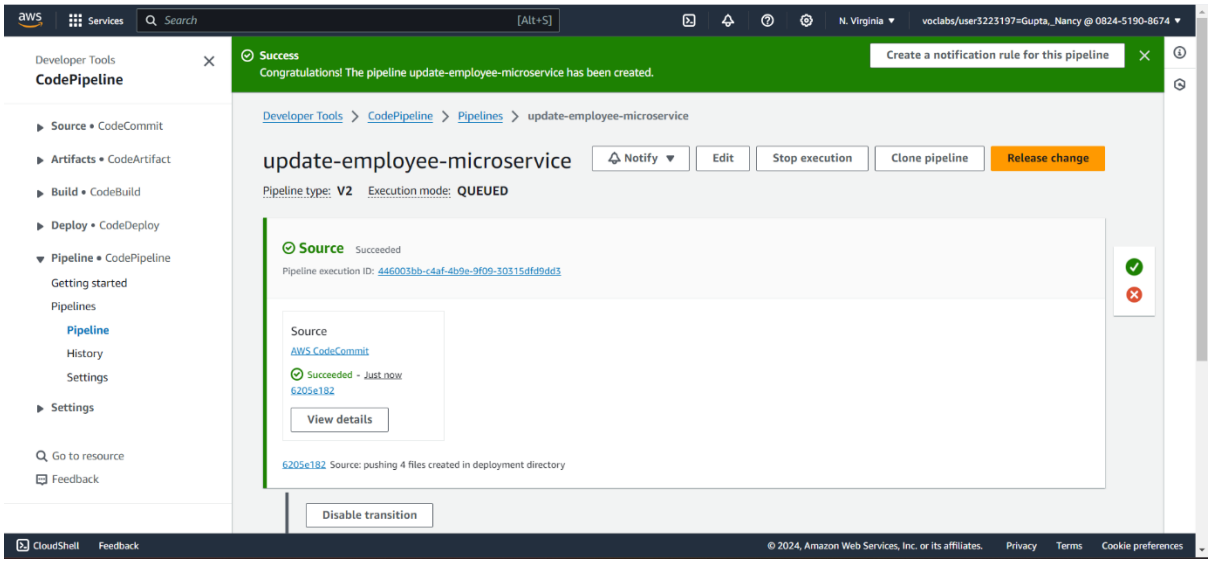

Developer Tools > CodePipeline > Pipelines > Create new pipeline
Step 1
Choose pipeline settings
Step 2
Add source stage
Step 3
Add build stage
Step 4
Add deploy stage
Step 5
Review
Review Info
Step 5 of 5
Step 1: Choose pipeline settings
Pipeline settings
Pipeline name
update-employee-microservice
Pipeline type
V2
Execution mode
QUEUED
Artifact location
codepipeline-us-east-1-788663725808
Service role name
arn:aws:iam::082451908674:role/PipelineRole

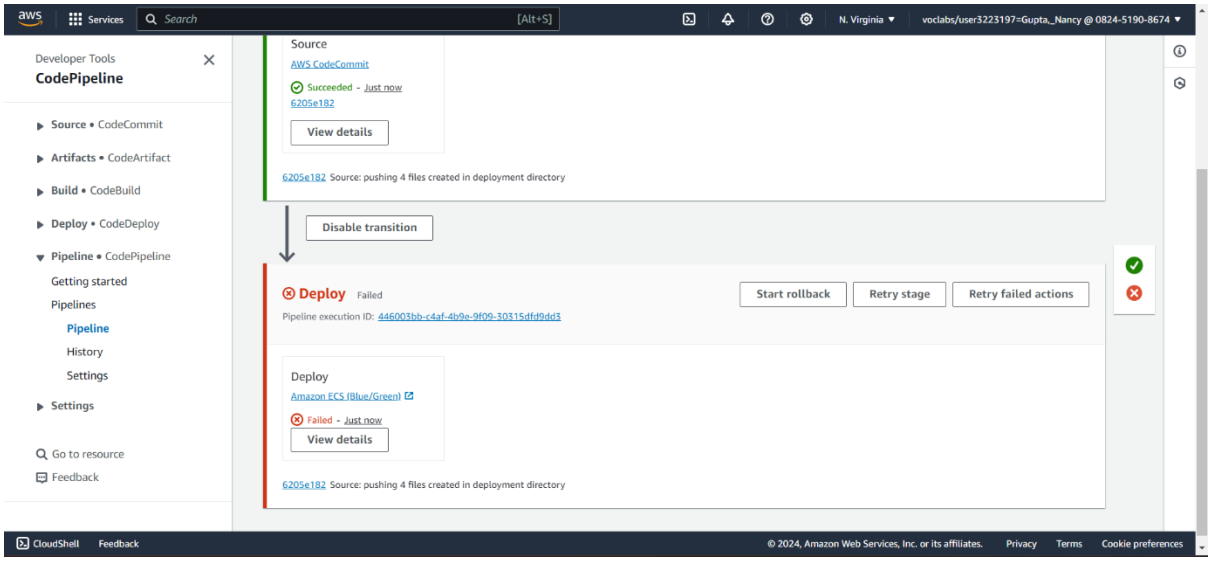

Variables
Name
Default value
Description
No variables
No variables defined at the pipeline level in this pipeline.
Step 2: Add source stage
Source action provider
Source action provider
AWS CodeCommit
RepositoryName
deployment
Default branch
dev
PollForSourceChanges
false


PollForSourceChanges
false
OutputArtifactFormat
CODE_ZIP
Step 3: Add build stage
Build action provider
Build stage
No build
Step 4: Add deploy stage
Deploy action provider
Deploy action provider
Amazon ECS (Blue/Green)
ApplicationName
microservices

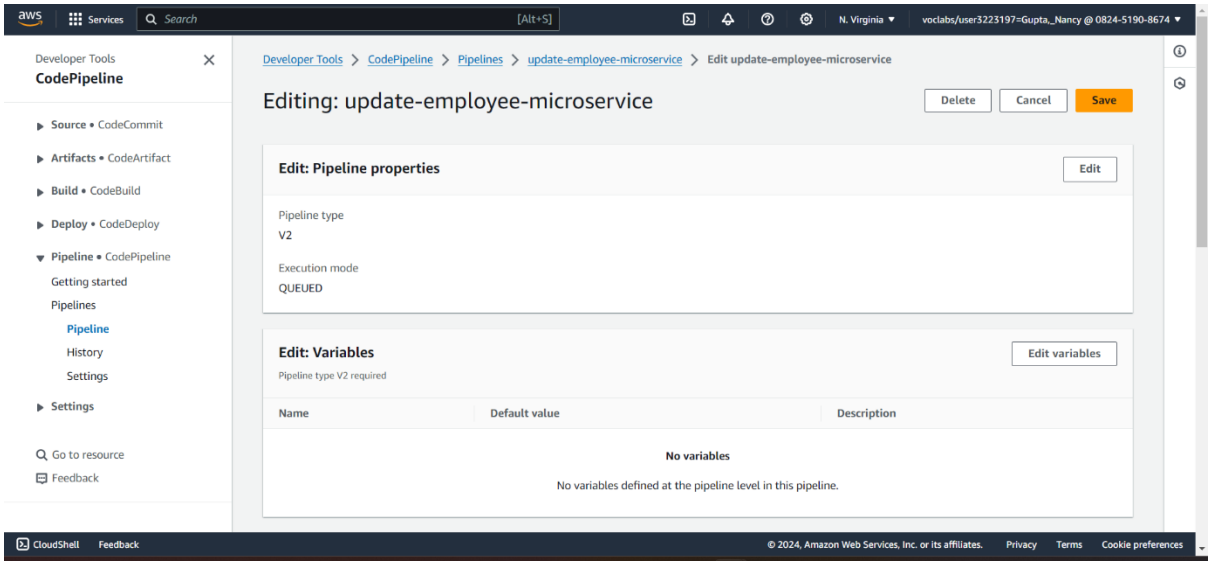

Deploy action provider
Deploy action provider
Amazon ECS (Blue/Green)
ApplicationName
microservices
DeploymentGroupName
microservices-employee
TaskDefinitionTemplateArtifact
SourceArtifact
TaskDefinitionTemplatePath
taskdef-employee.json
AppSpecTemplateArtifact
SourceArtifact
AppSpecTemplatePath
appspec-employee.yaml
Configure automatic rollback on stage failure
Disabled
Cancel
Previous
Create pipeline

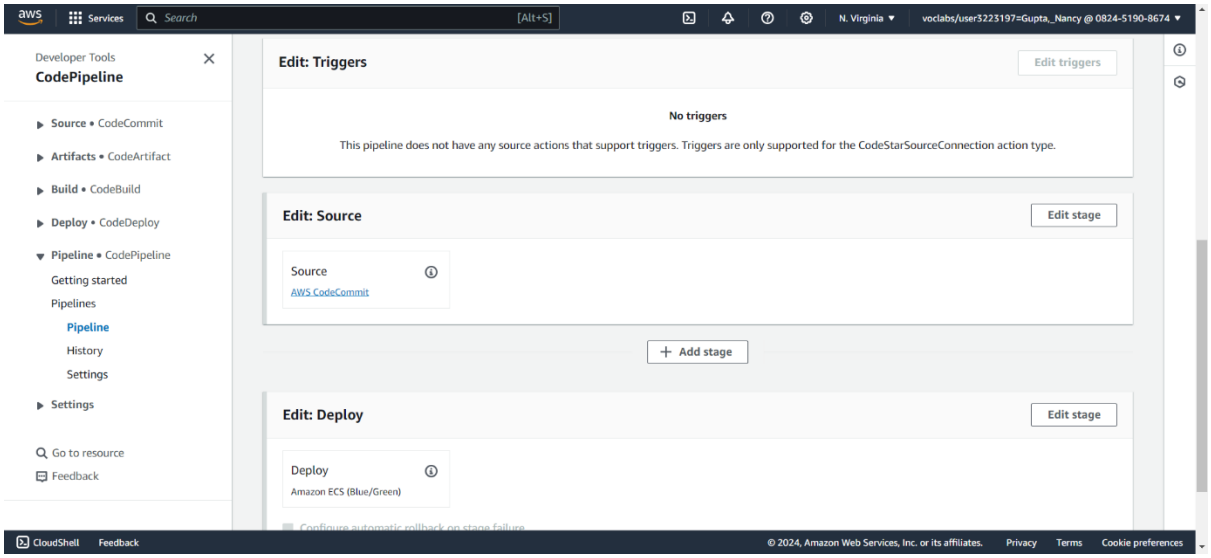

Success
Congratulations! The pipeline update-employee-microservice has been created.
Create a notification rule for this pipeline
Developer Tools
CodePipeline
Developer Tools > CodePipeline > Pipelines > update-employee-microservice
update-employee-microservice
Notify
Edit
Stop execution
Clone pipeline
Release change
Pipeline type: V2
Execution mode: QUEUED
Source In progress
Pipeline execution ID: 446003bb-c4af-4b9e-9f09-30315dfd9dd3
Source
AWS CodeCommit
Didn't Run
No executions yet
Disable transition
Deploy Didn't Run
Start rollback

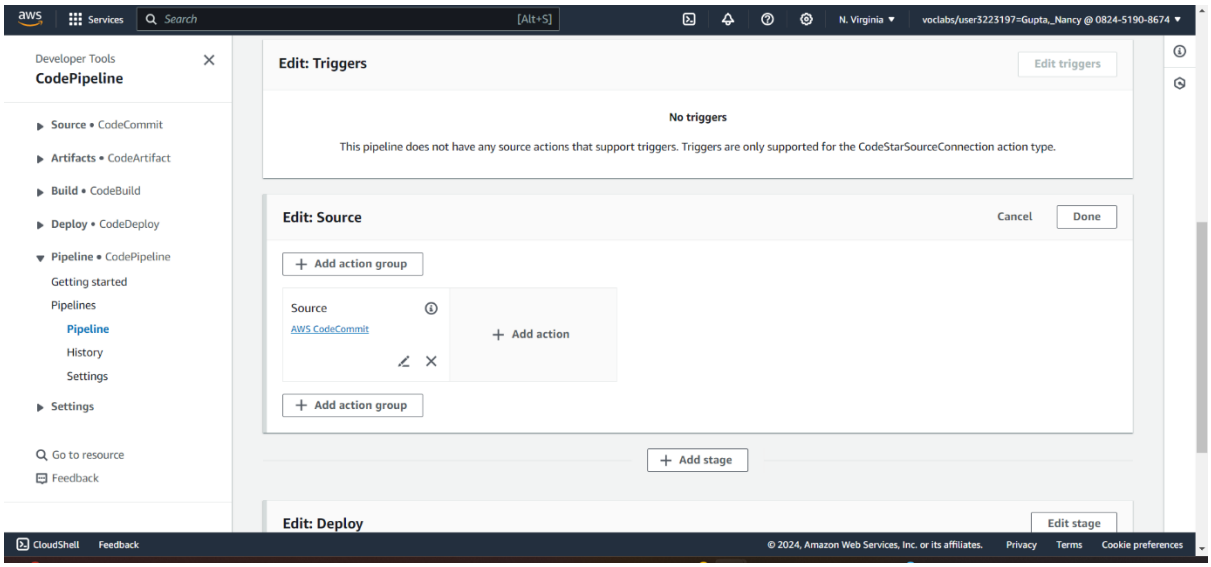

Success
Congratulations! The pipeline update-employee-microservice has been created.
Create a notification rule for this pipeline
Developer Tools
CodePipeline
Developer Tools > CodePipeline > Pipelines > update-employee-microservice
update-employee-microservice
Notify
Edit
Stop execution
Clone pipeline
Release change
Pipeline type: V2
Execution mode: QUEUED
Source Succeeded
Pipeline execution ID: 446003bb-c4af-4b9e-9f09-30315dfd9dd3
Source
AWS CodeCommit
Succeeded - Just now
6205e182
View details
6205e182 Source: pushing 4 files created in deployment directory
Disable transition



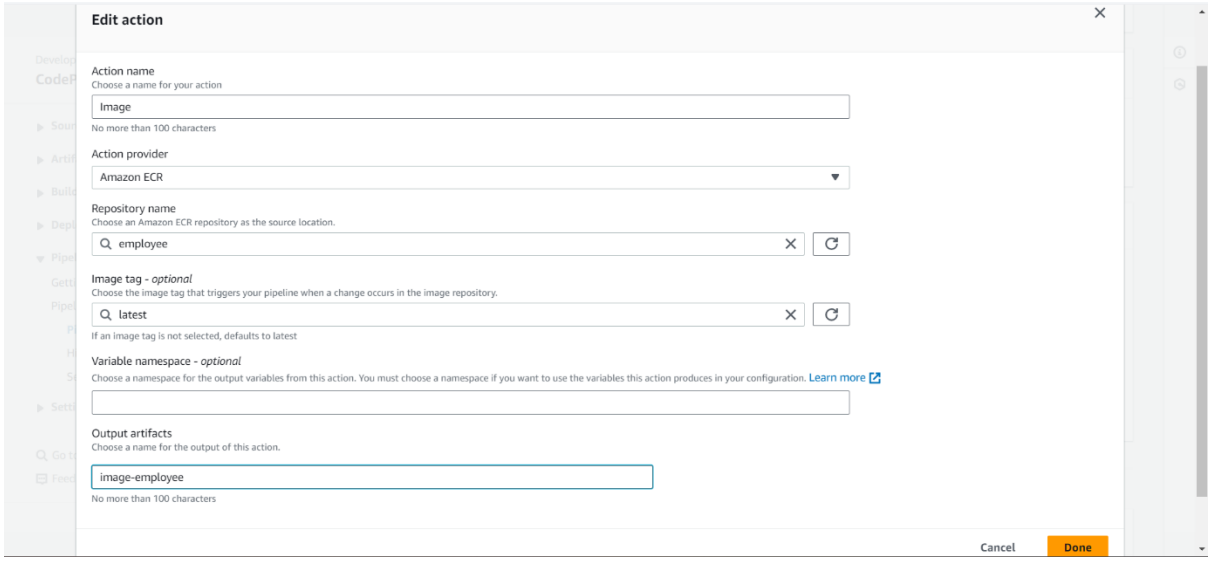
2. Add another *source* to the employee microservice pipeline. Add an action with the following details:

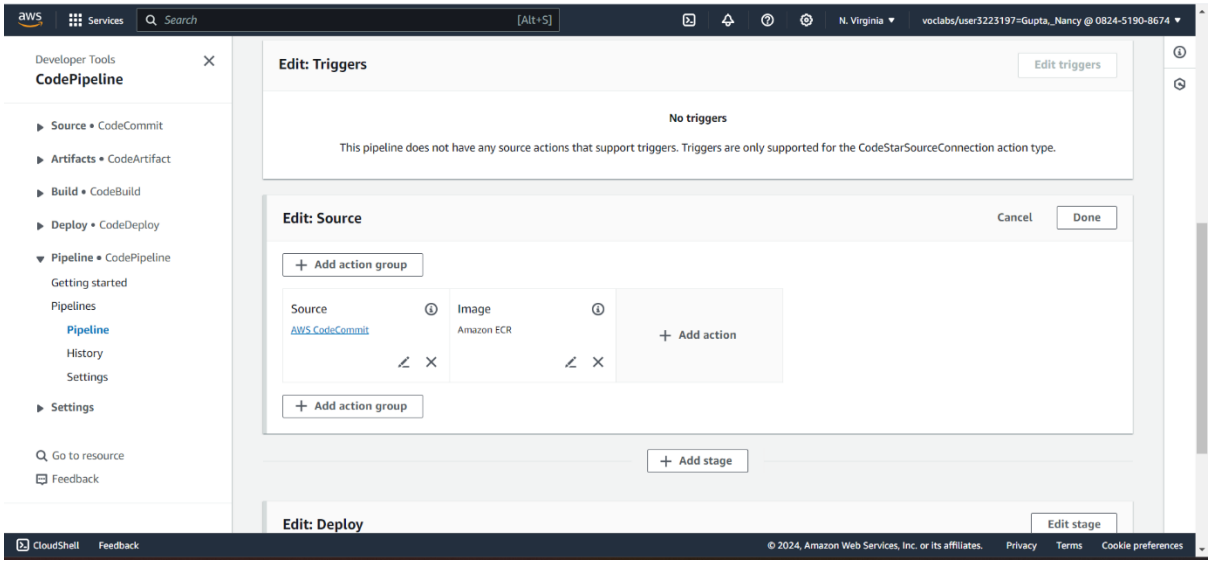




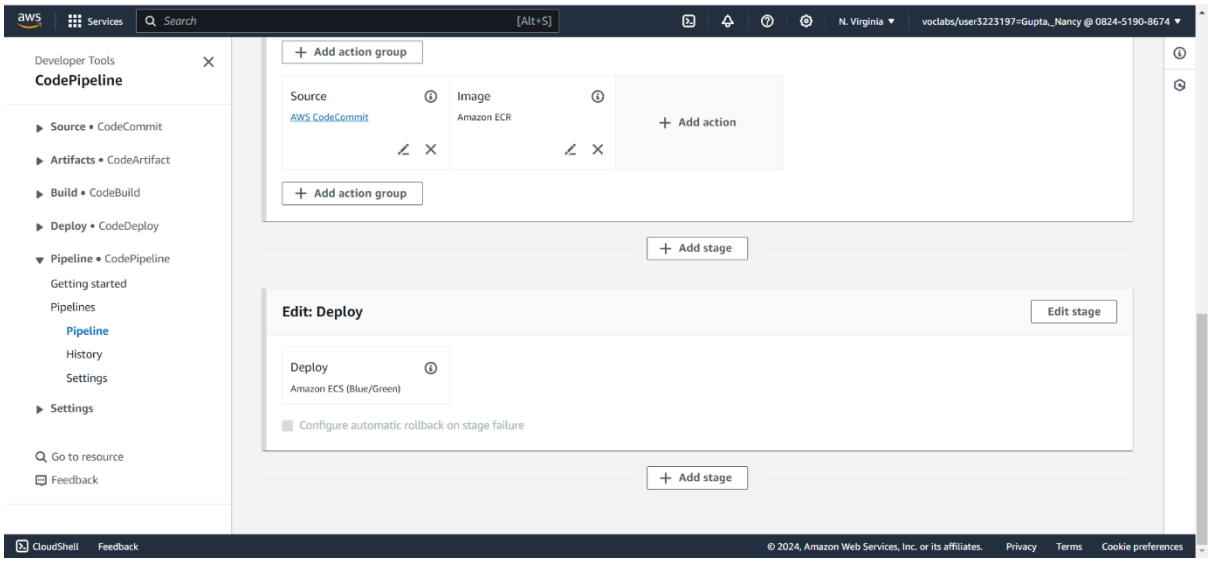


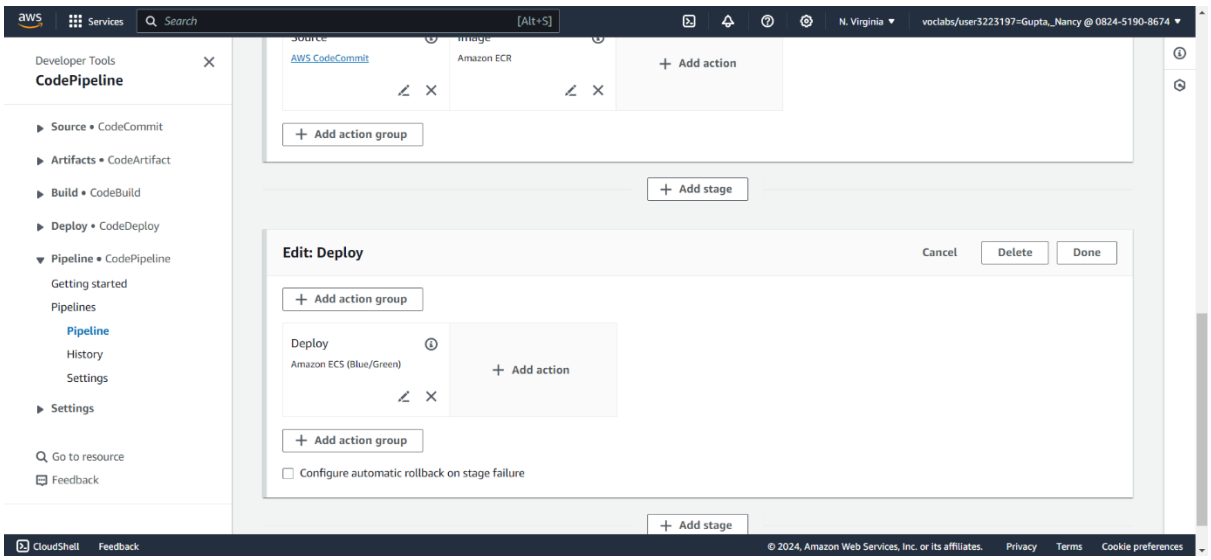
- Action name: Image
- Action provider: **Amazon ECR**
- Repository name: **employee**
- Image tag: **latest**
- Output artifacts: image-employee

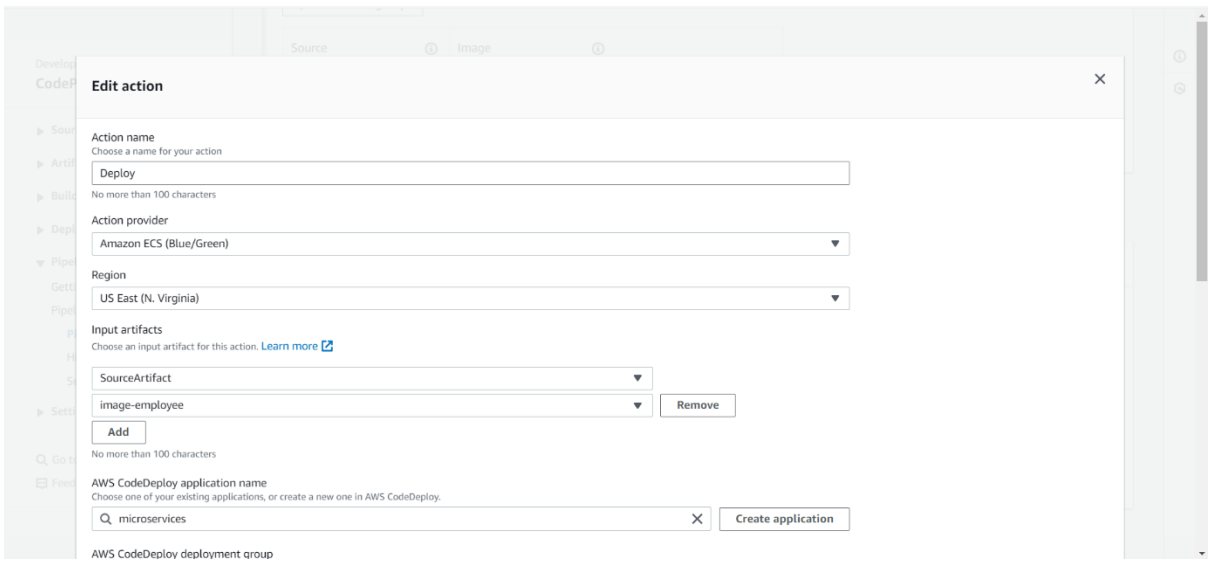


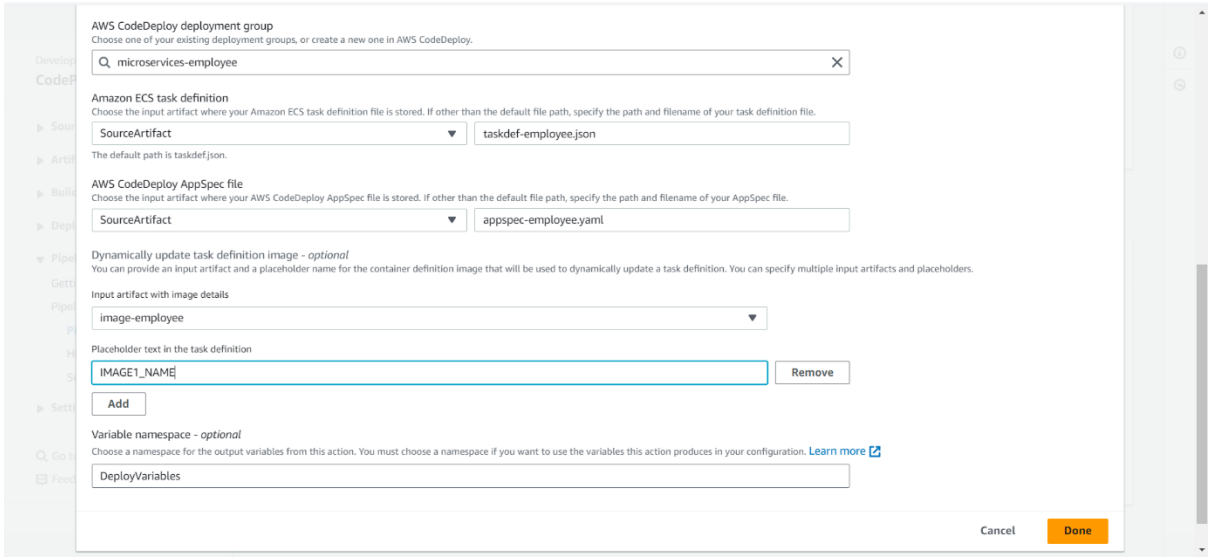


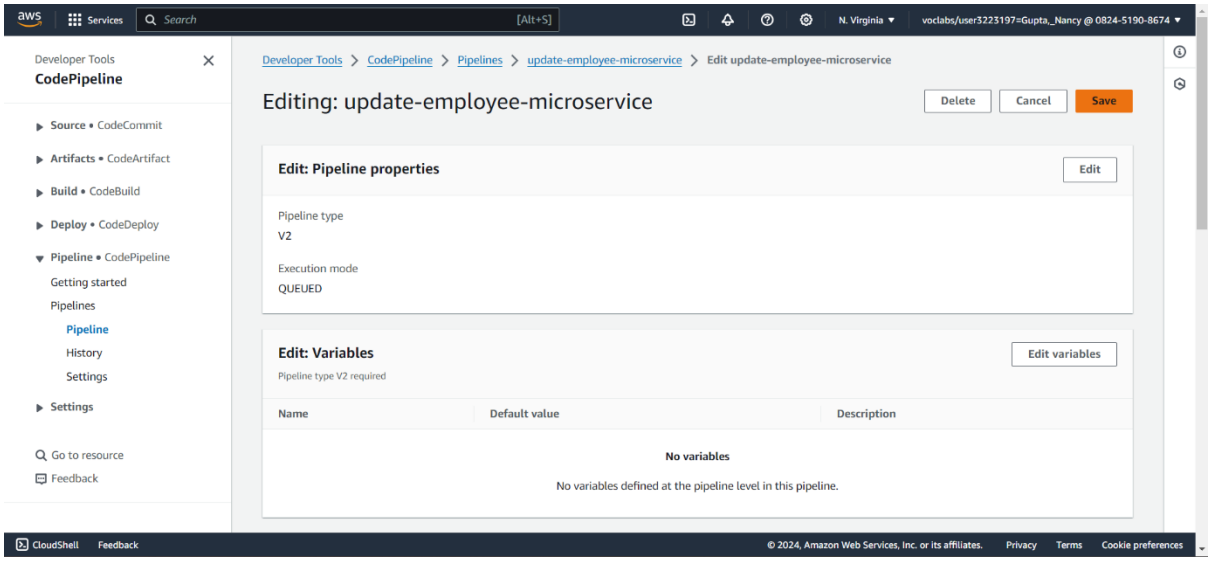
3. Edit the Amazon ECS (Blue/Green) action in the deploy stage:
    - Add another *input artifact* and choose **image-employee**.
    - Under **Dynamically update task definition image**, for **Input artifact with image details**, choose **image-employee**.
    - For **Placeholder text in the task definition**, enter IMAGE1_NAME

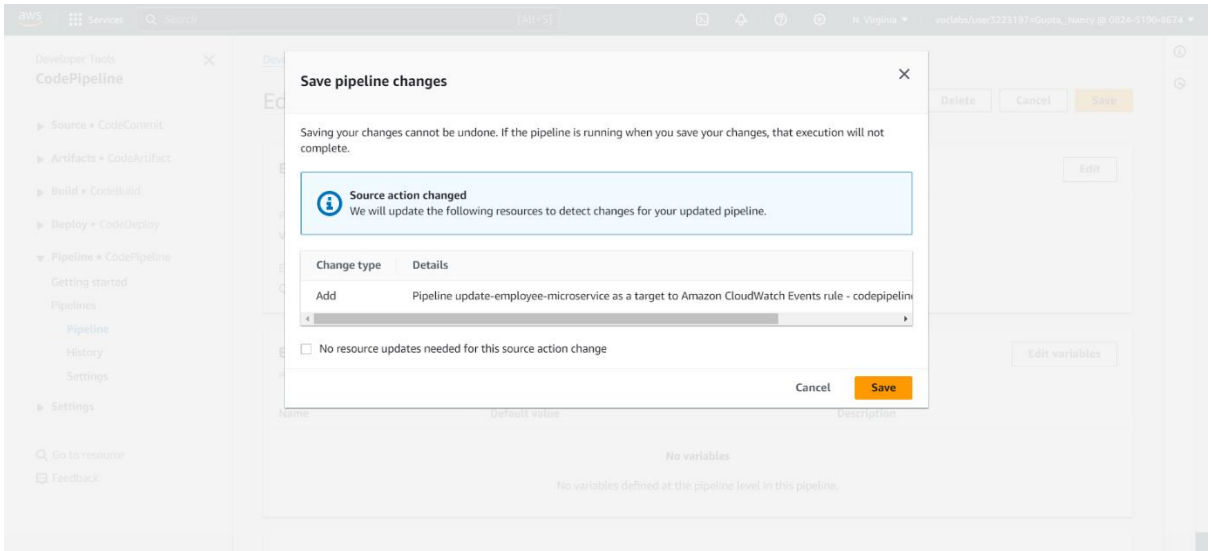

Services
Search
[Alt+S]
N. Virginia
voclabs/user3223197=Gupta,_Nancy @ 0824-5190-8674
Developer Tools
CodePipeline
Source • CodeCommit
Artifacts • CodeArtifact
Build • CodeBuild
Deploy • CodeDeploy
Pipeline • CodePipeline
Getting started
Pipelines
Pipeline
History
Settings
Settings
Go to resource
Feedback
Add action group
Source
AWS CodeCommit
Image
Amazon ECR
Add action
Add action group
Add stage
Edit: Deploy
Edit stage
Deploy
Amazon ECS (Blue/Green)
Configure automatic rollback on stage failure
Add stage
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

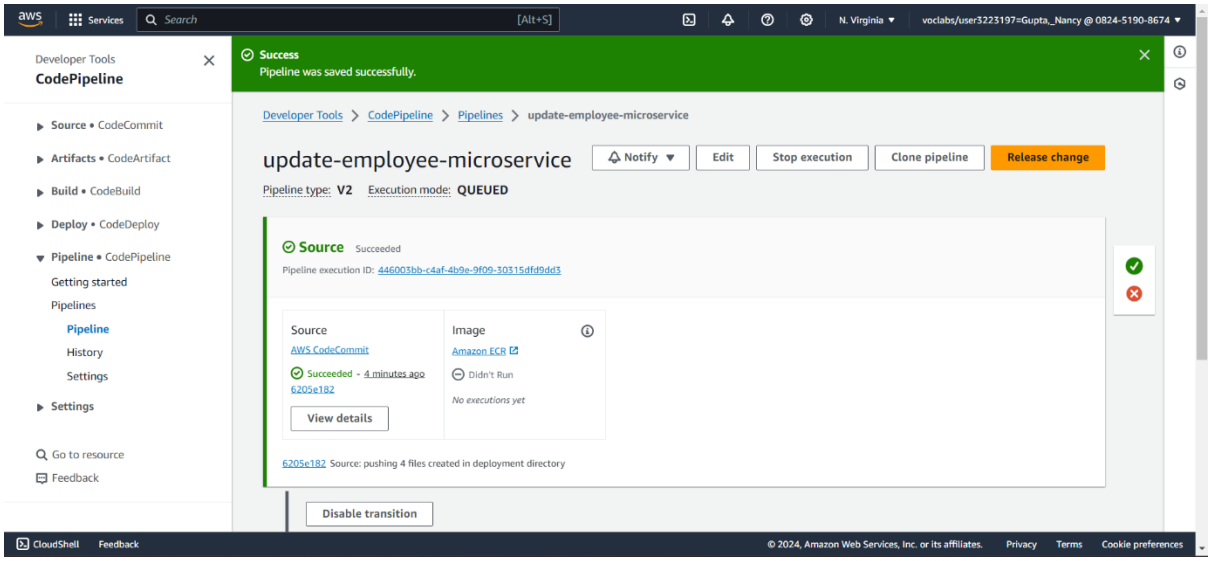

Services
Search
[Alt+S]
N. Virginia
voclabs/user3223197=Gupta,_Nancy @ 0824-5190-8674
Developer Tools
CodePipeline
Source • CodeCommit
Artifacts • CodeArtifact
Build • CodeBuild
Deploy • CodeDeploy
Pipeline • CodePipeline
Getting started
Pipelines
Pipeline
History
Settings
Settings
Go to resource
Feedback
AWS CodeCommit
Amazon ECR
Add action
Add action group
Add stage
Edit: Deploy
Cancel
Delete
Done
Add action group
Deploy
Amazon ECS (Blue/Green)
Add action
Add action group
Configure automatic rollback on stage failure
Add stage
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Edit action
Action name
Choose a name for your action
Deploy
No more than 100 characters
Action provider
Amazon ECS (Blue/Green)
Region
US East (N. Virginia)
Input artifacts
Choose an input artifact for this action. Learn more
SourceArtifact
image-employee
Remove
Add
No more than 100 characters
AWS CodeDeploy application name
Choose one of your existing applications, or create a new one in AWS CodeDeploy.
microservices
Create application
AWS CodeDeploy deployment group


AWS CodeDeploy deployment group
Choose one of your existing deployment groups, or create a new one in AWS CodeDeploy.
microservices-employee
Amazon ECS task definition
Choose the input artifact where your Amazon ECS task definition file is stored. If other than the default file path, specify the path and filename of your task definition file.
SourceArtifact
taskdef-employee.json
The default path is taskdef.json.
AWS CodeDeploy AppSpec file
Choose the input artifact where your AWS CodeDeploy AppSpec file is stored. If other than the default file path, specify the path and filename of your AppSpec file.
SourceArtifact
appspec-employee.yaml
Dynamically update task definition image - optional
You can provide an input artifact and a placeholder name for the container definition image that will be used to dynamically update a task definition. You can specify multiple input artifacts and placeholders.
Input artifact with image details
image-employee
Placeholder text in the task definition
IMAGE1_NAME
Remove
Add
Variable namespace - optional
Choose a namespace for the output variables from this action. You must choose a namespace if you want to use the variables this action produces in your configuration. Learn more
DeployVariables
Cancel
Done


Services
Search
[Alt+S]
N. Virginia
Developer Tools
CodePipeline
Source • CodeCommit
Artifacts • CodeArtifact
Build • CodeBuild
Deploy • CodeDeploy
Pipeline • CodePipeline
Getting started
Pipelines
Pipeline
History
Settings
Settings
Go to resource
Feedback
Developer Tools > CodePipeline > Pipelines > update-employee-microservice > Edit update-employee-microservice
Editing: update-employee-microservice
Delete
Cancel
Save
Edit: Pipeline properties
Edit
Pipeline type
V2
Execution mode
QUEUED
Edit: Variables
Edit variables
Pipeline type V2 required
Name
Default value
Description
No variables
No variables defined at the pipeline level in this pipeline.
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Save pipeline changes
Saving your changes cannot be undone. If the pipeline is running when you save your changes, that execution will not complete.
Source action changed
We will update the following resources to detect changes for your updated pipeline.
Change type
Details
Add
Pipeline update-employee-microservice as a target to Amazon CloudWatch Events rule - codepipelin
No resource updates needed for this source action change
Cancel
Save

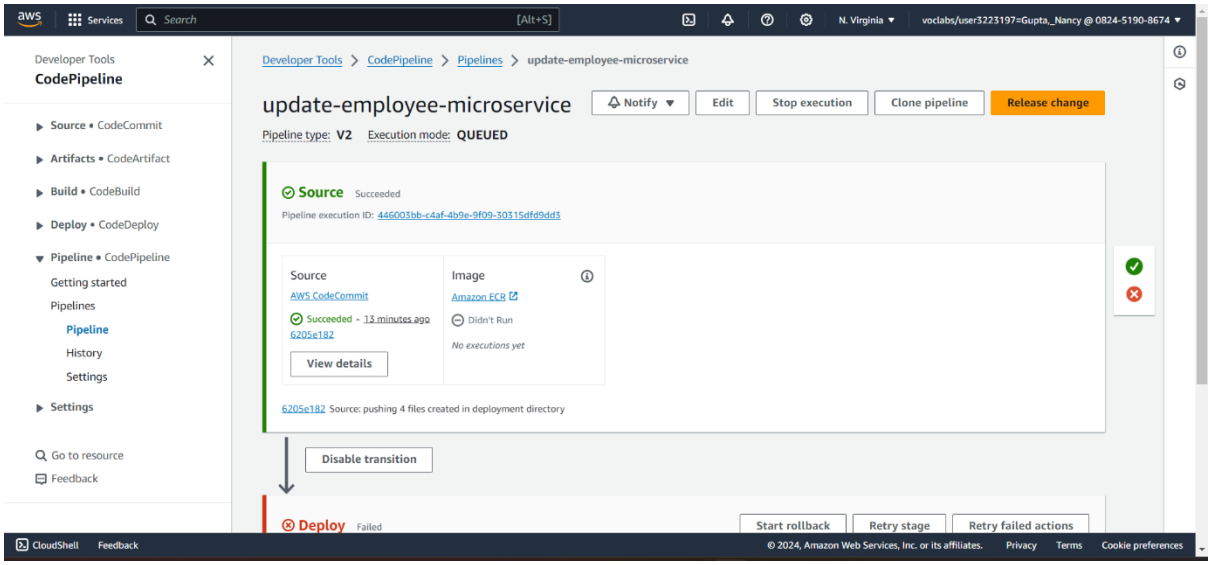


Task 8.5: Test the CI/CD pipeline for the *employee* microservice

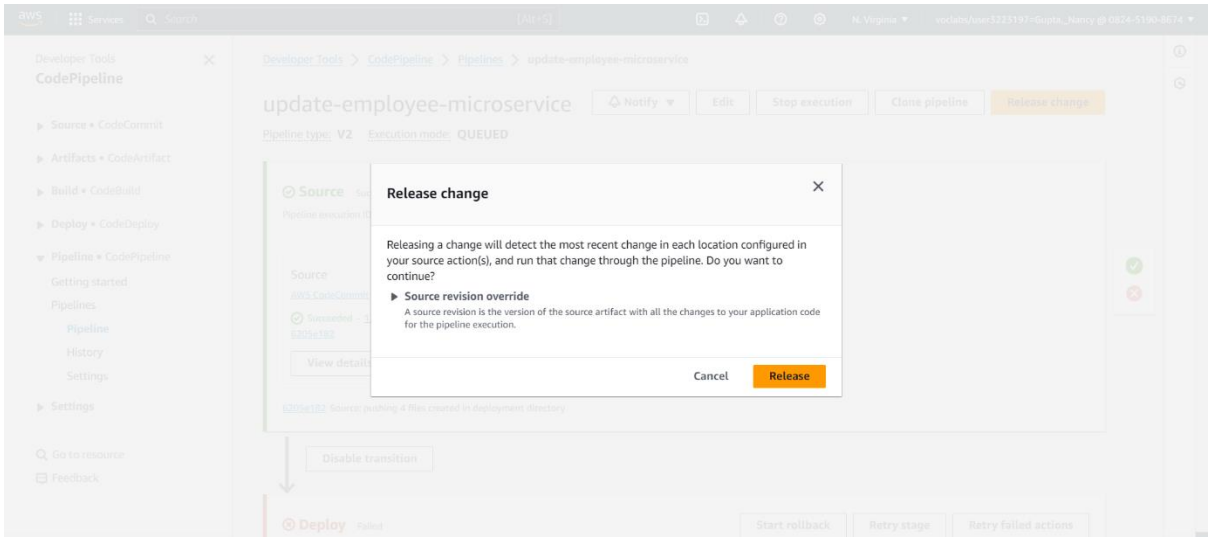
In this task, you will test the pipeline that you just defined for the *employee* microservice.

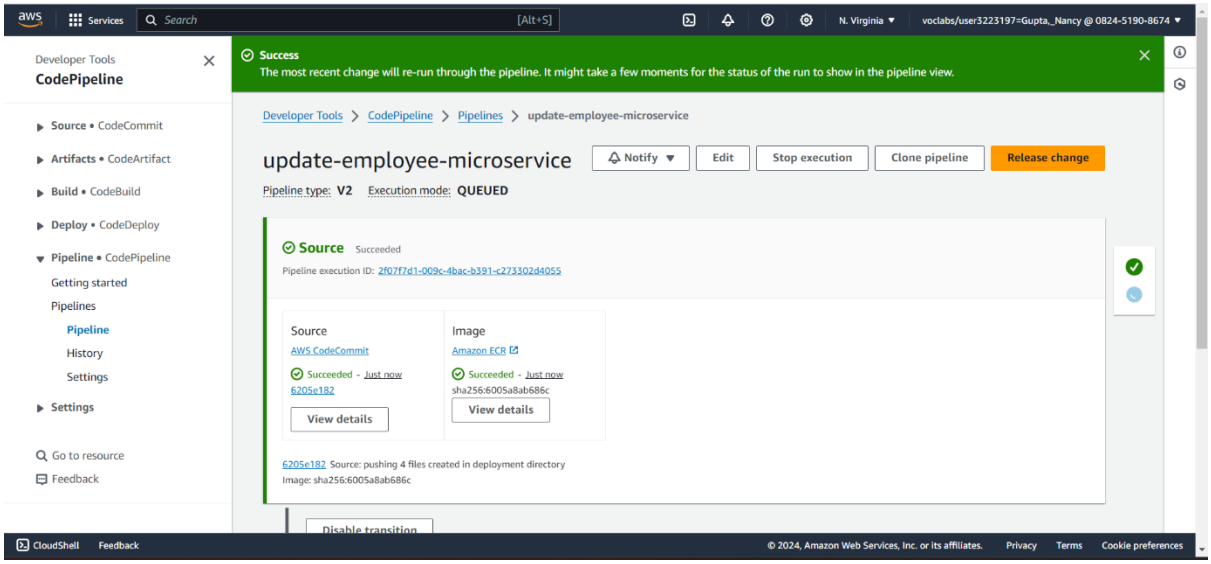
Important: If you are repeating this task, confirm that all target groups are still associated with the Application Load Balancer. The Reassociate Target Groups with Load Balancer section of the appendix provides detail if needed.

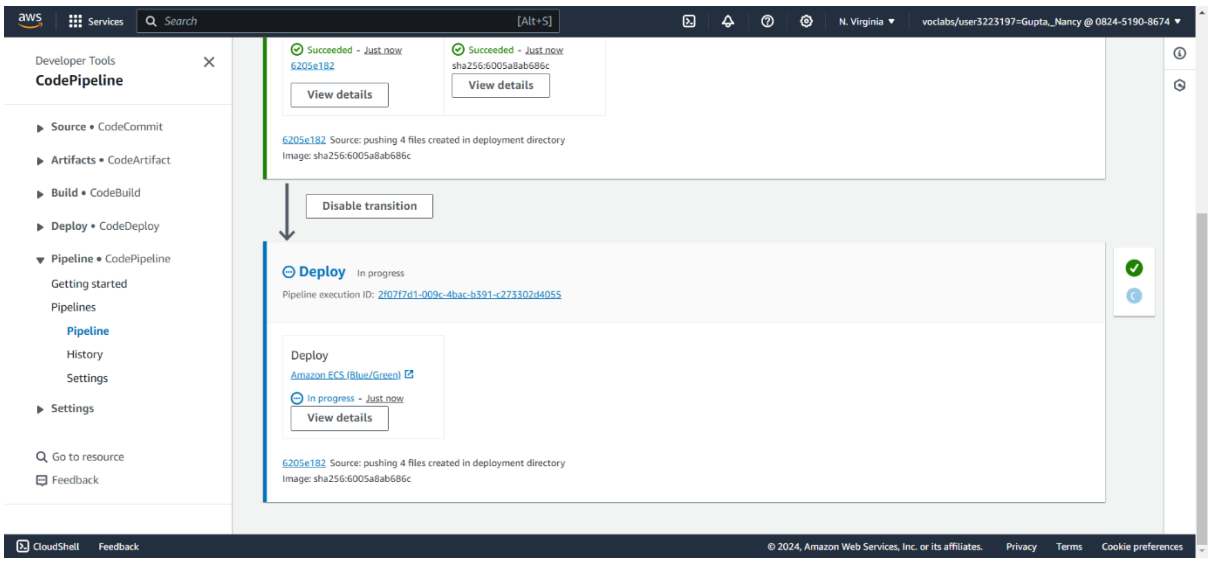
1. Launch a deployment of the *employee* microservice on Amazon ECS on Fargate.



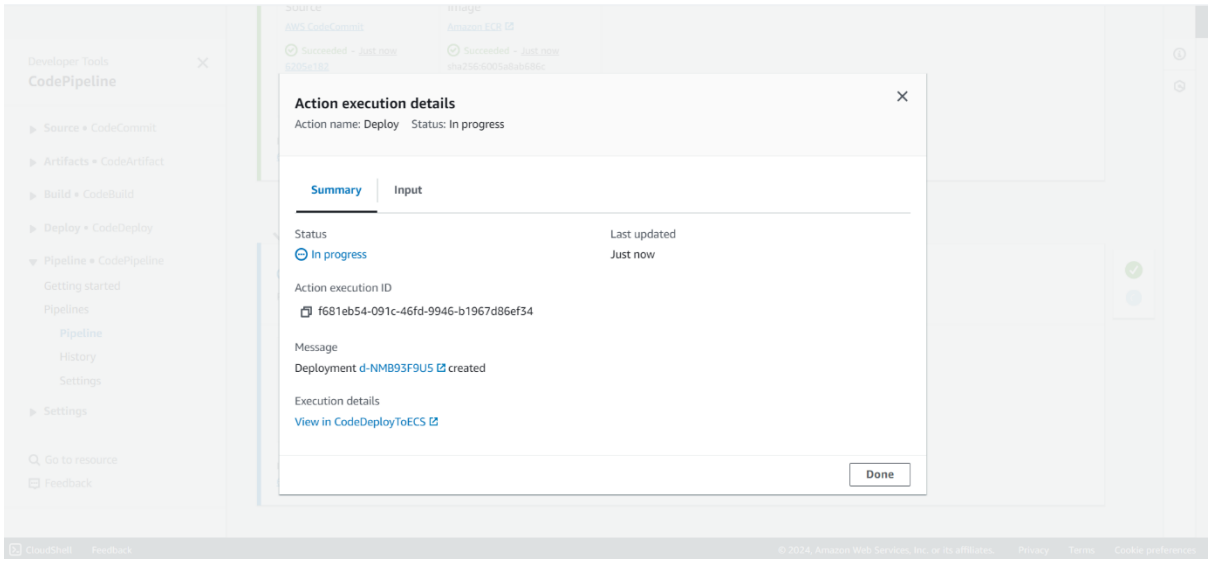
   - Use the *release change* feature to force a test of the pipeline.

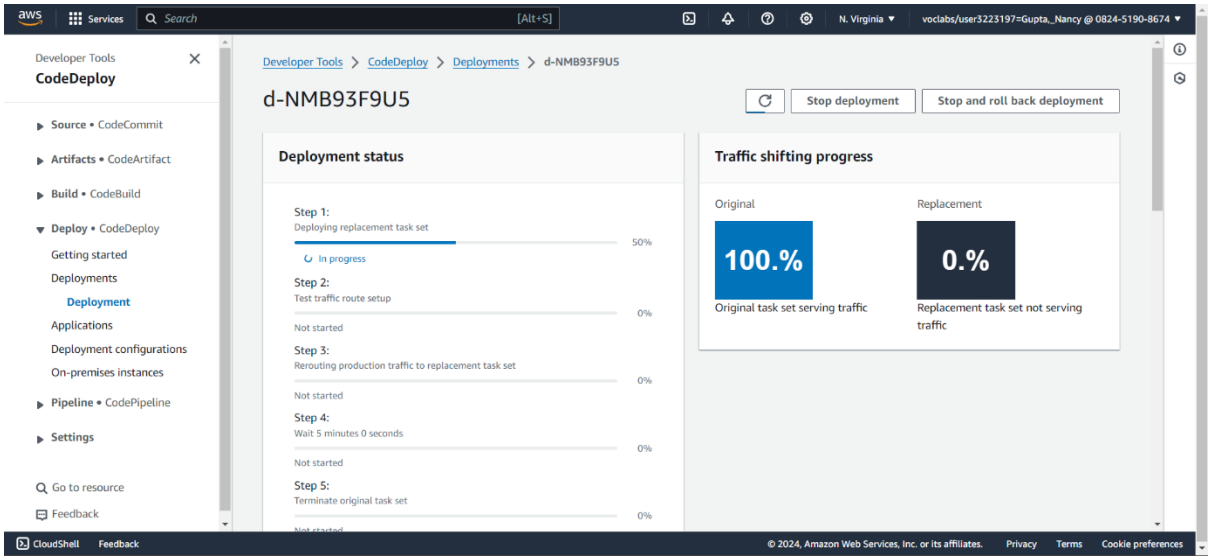


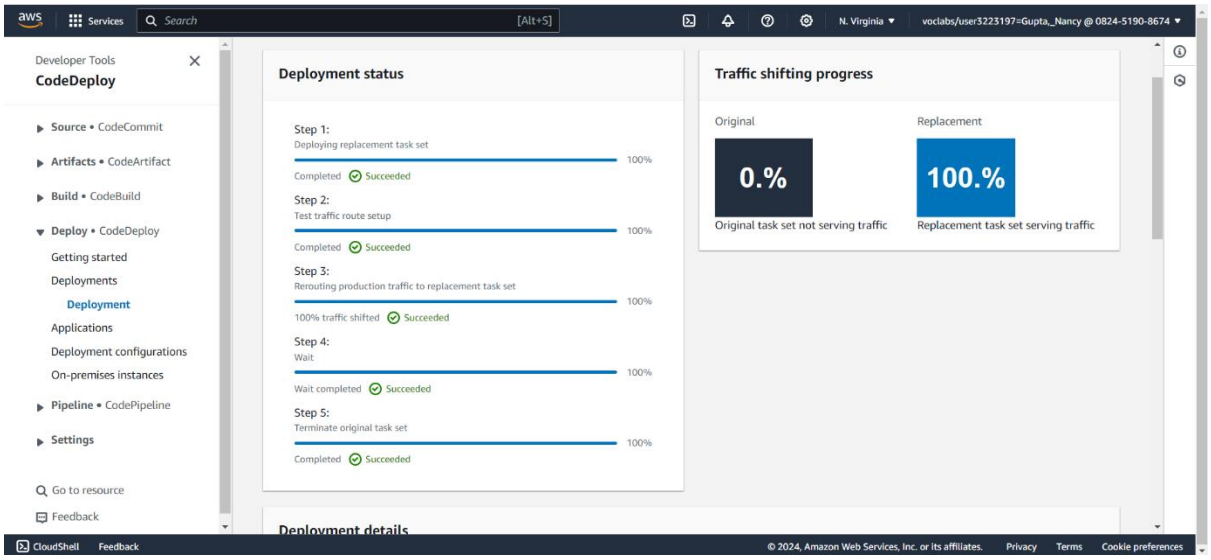


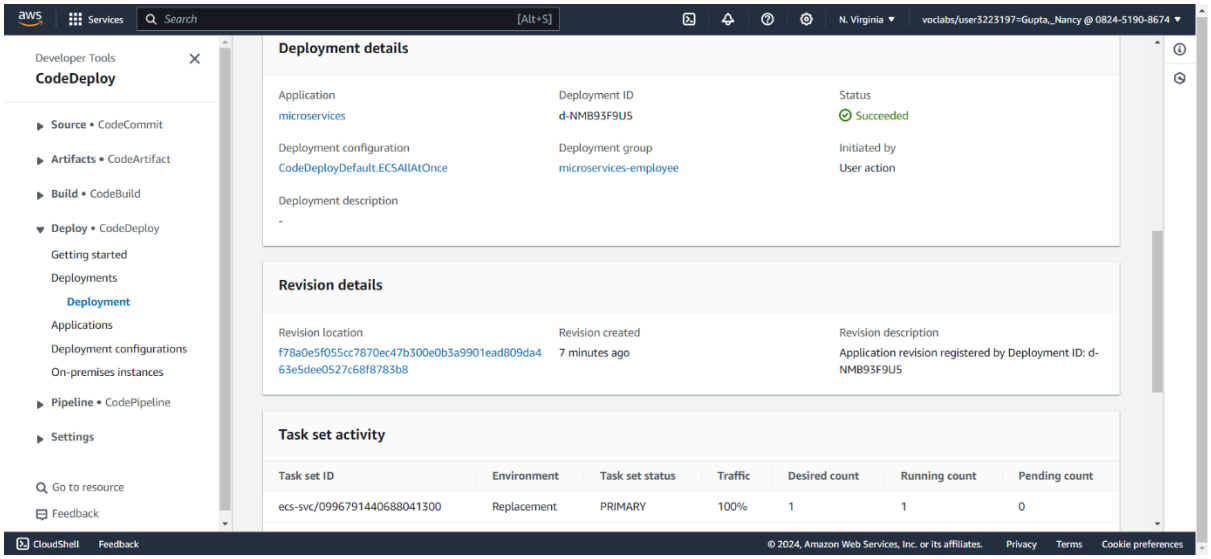


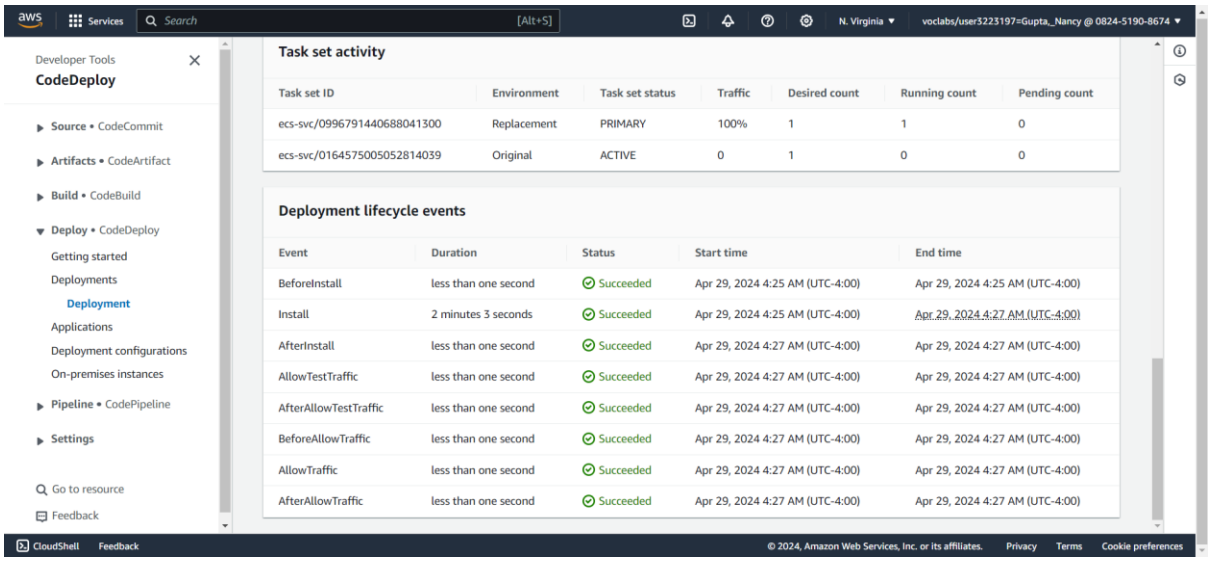
- Follow the progress in CodeDeploy. Within a few minutes, if everything was configured correctly, all of the *Deployment lifecycle events* should succeed.


Action execution details
Action name: Deploy Status: In progress
Summary
Input
Status
In progress
Last updated
Just now
Action execution ID
f681eb54-091c-46fd-9946-b1967d86ef34
Message
Deployment d-NMB93F9U5 created
Execution details
View in CodeDeployToECS
Done


Developer Tools
CodeDeploy
Developer Tools > CodeDeploy > Deployments > d-NMB93F9U5
d-NMB93F9U5
Stop deployment
Stop and roll back deployment
Deployment status
Step 1:
Deploying replacement task set
50%
In progress
Step 2:
Test traffic route setup
0%
Not started
Step 3:
Rerouting production traffic to replacement task set
0%
Not started
Step 4:
Wait 5 minutes 0 seconds
0%
Not started
Step 5:
Terminate original task set
0%
Traffic shifting progress
Original
100.%
Original task set serving traffic
Replacement
0.%
Replacement task set not serving traffic


Developer Tools
CodeDeploy
Deployment status
Step 1:
Deploying replacement task set
100%
Completed Succeeded
Step 2:
Test traffic route setup
100%
Completed Succeeded
Step 3:
Rerouting production traffic to replacement task set
100%
100% traffic shifted Succeeded
Step 4:
Wait
100%
Wait completed Succeeded
Step 5:
Terminate original task set
100%
Completed Succeeded
Traffic shifting progress
Original
0.%
Original task set not serving traffic
Replacement
100.%
Replacement task set serving traffic
Deployment details

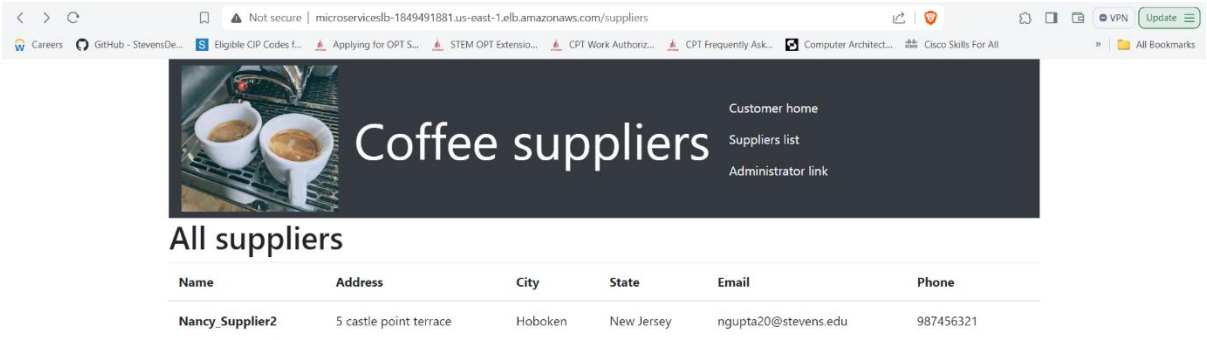


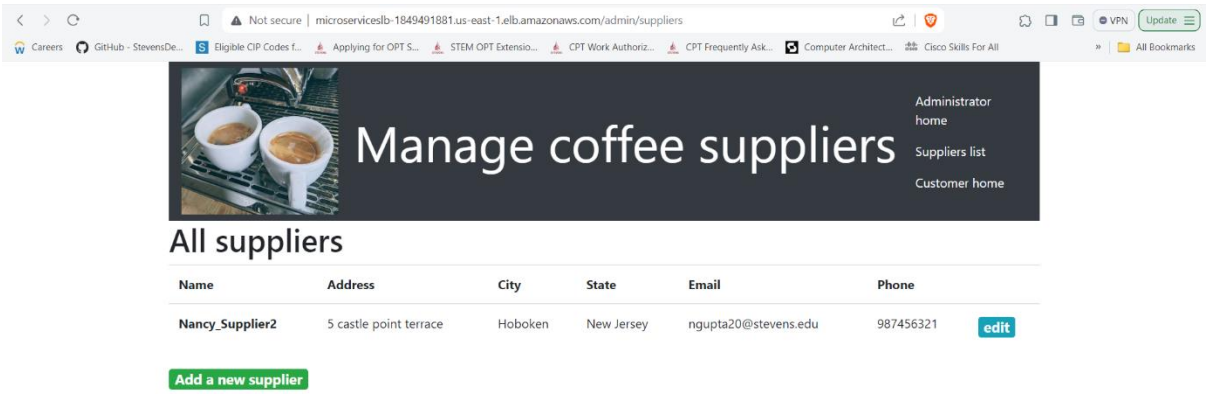


2. Load the *employee* microservice in a browser tab.
    - In the browser tab where the customer microservice is running, choose **Administrator link**.

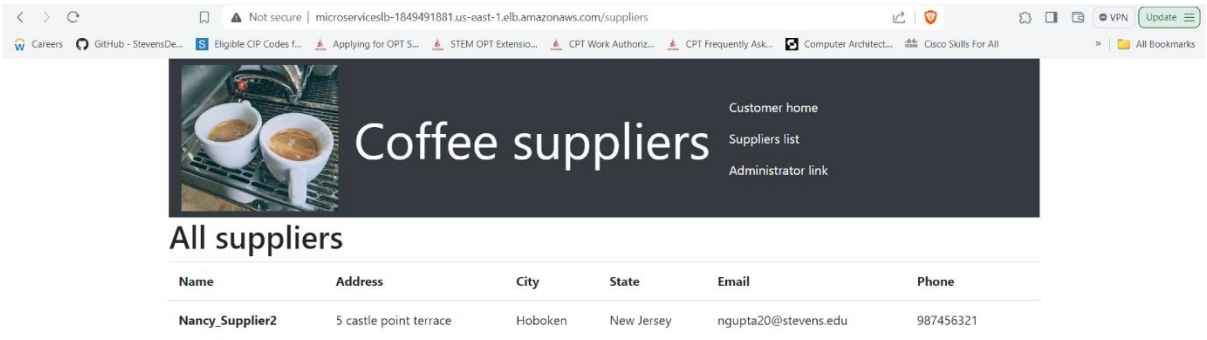
You are directed to a page in the format *http://microserviceslb-UNIQUE-NUMBER.us-east-1.elb.amazonaws.com/admin/suppliers*.

The employee microservice loads. If it doesn't load, try adding :8080 just after amazonaws.com in the URL.





- Choose **List of suppliers** or **Suppliers list**.



The suppliers page should load. This version of the page should *not* have the edit or add supplier buttons. All links in the café web application should now work because you have now deployed both microservices.

3. Observe the running tasks in the Amazon ECS console.

The **Deployments and tasks** status will change as the blue/green deployment advances through its lifecycle events.