CI/CD Pipeline for the Hybrid Workloads on ECS Anywhere by using AWS CDK and GitLab

Dr. Rahul Sharad Gaikwad
Lead DevOps Consultant, AWS


```
"app": "npx ts-node --prefer-ts-exts bin/rahul-intro.ts",
"context": {
  "myname": "Dr.Rahul Sharad Gaikwad",
  "designation": "Lead Devops Consultant",
  "company": "AWS",
  "address": {
    "city": "Pune",
    "state": "Maharastra",
    "country": "India"
  },
  "education": "Ph.D in AIOPS",
  "aboutme": "I helps customers to migrate and modernize workloads to AWS Cloud.",
  "intrests": [
    "reading books",
    "travelling new places",
    "spend time with my family"
  ]
}
```


1 Use Cases | 2 Challenges | 3 Benefits | 4 Solution | 5 Demo | 6 Conclusion

Use cases

Hybrid workloads

Customer's who need to run containerized workloads on both cloud and on-premises.

- Data Gravity
- Compliance
- Latency

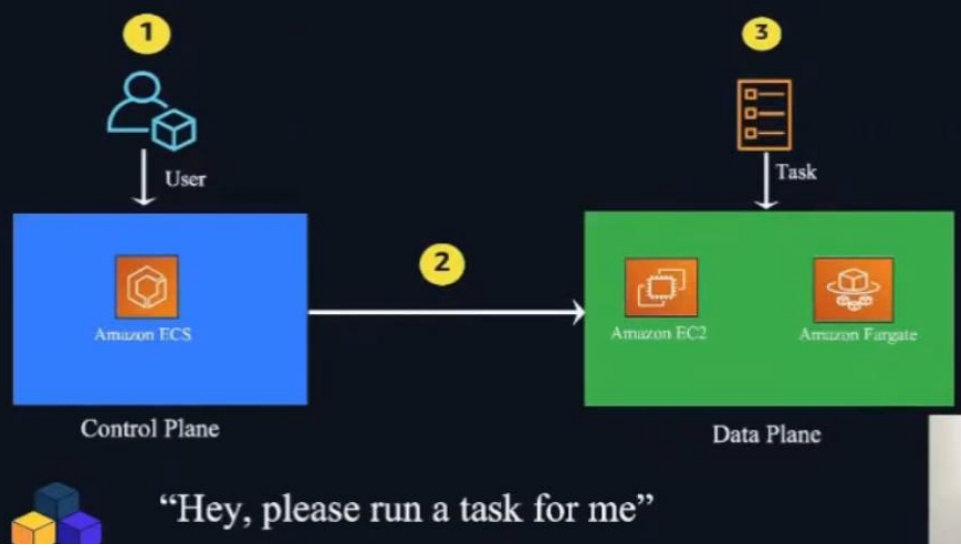
Use cases

On-premise Investment

Customer's who want to continue to utilize their on-premises infrastructure until their investments have fully pay off.

- On-prem investment
- Compute resources
- Skills / expertise

How does Amazon ECS Works?

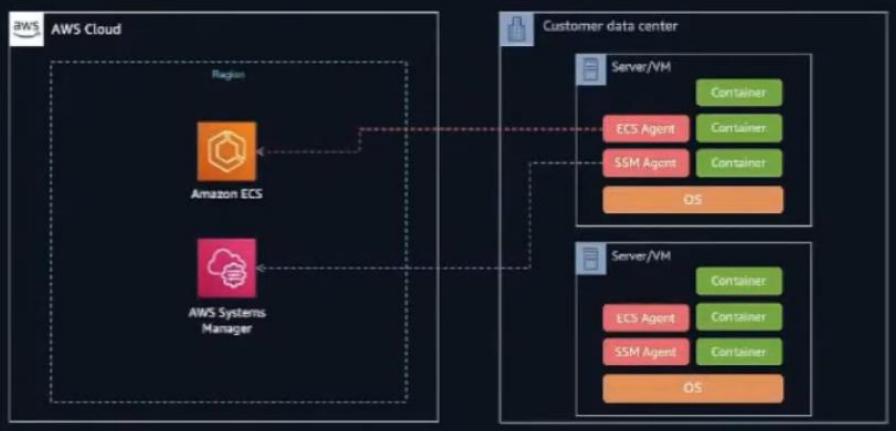


"Hey, please run a task for me"

Challenges / Considerations

- To avoid Multiple Operational models
- To protect the capital investment in their data center
- To orchestrate the containers & management in compliance infrastructure
- To deploy, manage and orchestrate the containers near to the business industries

How does Amazon ECS Anywhere works?



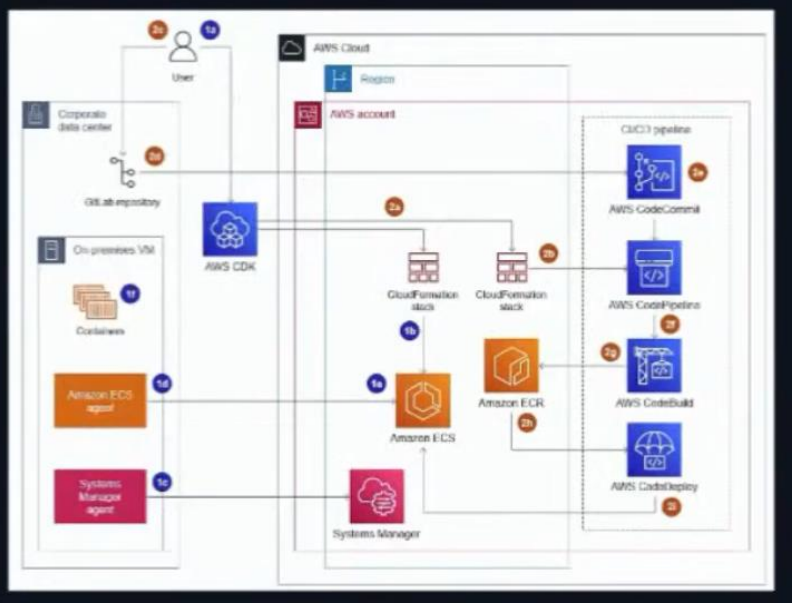
Customer Benefits

- Unified, fully-managed control plane
- Single management interface and API
- Reusable Task definitions
- Consistent tooling and governance
- AWS security, scalability, and reliability

Problem Statement

Customers who are already running their container applications using on-premise infrastructure and controlling codebase using GitLab repositories, but now want to manage their workload using AWS Cloud services without disturbing and sacrificing existing on-premise infrastructure and investments.

Solution



Demo Time

```
{
  "app": "sarat-intro.ts",
  "context": {
    "myname": "Sarat Chandra Pothula",
    "designation": "Devops Consultant",
    "company": "AWS Professional Services",
    "address": {
      "city": "Hyderabad",
      "state": "Telangana",
      "country": "India"
    },
    "skills": {
      "skill1": "strong in devops processes",
      "skill2": "cloud enablement"
    },
    "areaofexpertise": "Information Technology",
    "intrests": [
      "reading books",
      "travelling new places",
      "spend time with my family"
    ]
  }
}
```

SARAT CHANDRA POTHULA
DevOps Consultant, AWS

Pre-requisites:

- ✓ An active AWS account.
- ✓ AWS Command Line Interface (AWS CLI), installed and configured.
- ✓ AWS CDK Toolkit, installed and configured.
- ✓ Node package manager (npm), installed and configured for the AWS CDK in TypeScript.

Product Versions:

- ✓ AWS CDK Version 2.94.0 or later
- ✓ Node Version 18.16.0 or later
- ✓ NPM Version 9.5.1 or later

Source Code:

- ✓ https://github.com/aws-samples/amazon-ecs-anywhere-cicd-pipeline-cdk-sample.git

Steps to follow

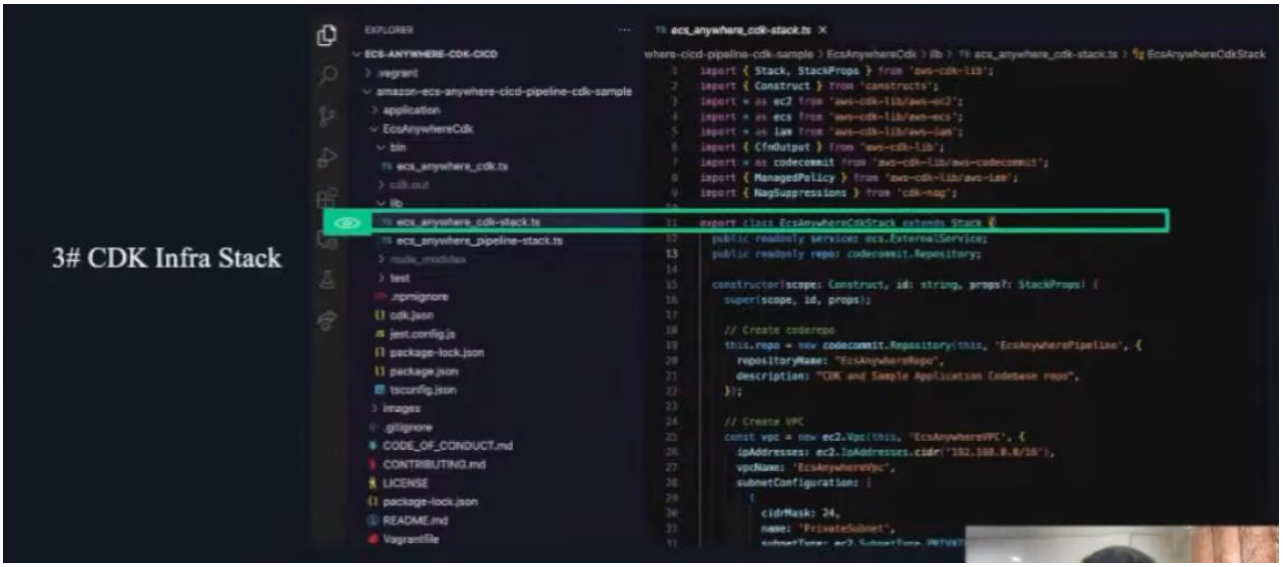
1# Clone Git Repository

```
bash-3.2$ git clone https://github.com/aws-samples/amazon-ecs-anywhere-cicd-pipeline-cdk-sample.git
Cloning into 'amazon-ecs-anywhere-cicd-pipeline-cdk-sample'...
remote: Enumerating objects: 83, done.
remote: Counting objects: 100% (83/83), done.
remote: Compressing objects: 100% (70/70), done.
remote: Total 83 (delta 29), reused 52 (delta 10), pack-reused 0
Receiving objects: 100% (83/83), 323.74 KiB | 830.00 KiB/s, done.
Resolving deltas: 100% (29/29), done.
```

2# Setup AWS Profile

```
bash-3.2$ aws configure
AWS Access Key ID [****************4LOI]: ********************
AWS Secret Access Key [****************OoOv]: ****************
Default region name [us-east-1]: us-east-1
Default output format [None]:
```

3# CDK Infra Stack



4# Install latest CDK

```
bash-3.2$ npm install -g aws-cdk@latest
```

5# Install Node Packages

```
bash-3.2$ sudo npm ci

added 222 packages, and audited 907 packages in 47s

3 packages are looking for funding
  run `npm fund` for details

41 moderate severity vulnerabilities

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
```

6# Bootstrap CDK with AWS Account

```
bash-3.2$ cdk bootstrap
⏳  Bootstrapping environment
```

7# List CDK Stack

```
bash-3.2$ cdk ls
EcsAnywhereInfraStack
EcsAnywherePipelineStack
```

8# Synthesizes a stack

```
bash-3.2$ cdk synth EcsAnywhereInfraStack
```

9# cdk deploy
EcsAnywhereInfraStack

IAM Statement Changes

	Resource	Effect	Action	Principal	Condition
+	${EcsAnywhereInstanceRole.Arn}	Allow	sts:AssumeRole	Service:ssm.amazonaws.com	
+	${ExternalTaskDefinition/EcsAnywhereContainer/LogGroup.Arn}	Allow	logs:CreateLogStream logs:PutLogEvents	AWS:${ExternalTaskDefinition/ExecutionRole}	
+	${ExternalTaskDefinition/ExecutionRole.Arn}	Allow	sts:AssumeRole	Service:ecs-tasks.amazonaws.com	
+	${ecs-taskRole-EcsAnywhereInfraStack.Arn}	Allow	sts:AssumeRole	Service:ecs-tasks.amazonaws.com	
+	*	Allow	ecr:BatchCheckLayerAvailability ecr:BatchGetImage ecr:GetAuthorizationToken ecr:GetDownloadUrlForLayer logs:CreateLogStream logs:PutLogEvents	AWS:${ExternalTaskDefinition/ExecutionRole}	
+	arn:aws:logs:*:*:*	Allow	logs:CreateLogGroup logs:CreateLogStream logs:DescribeLogStreams logs:PutLogEvents	AWS:${ecs-taskRole-EcsAnywhereInfraStack}	

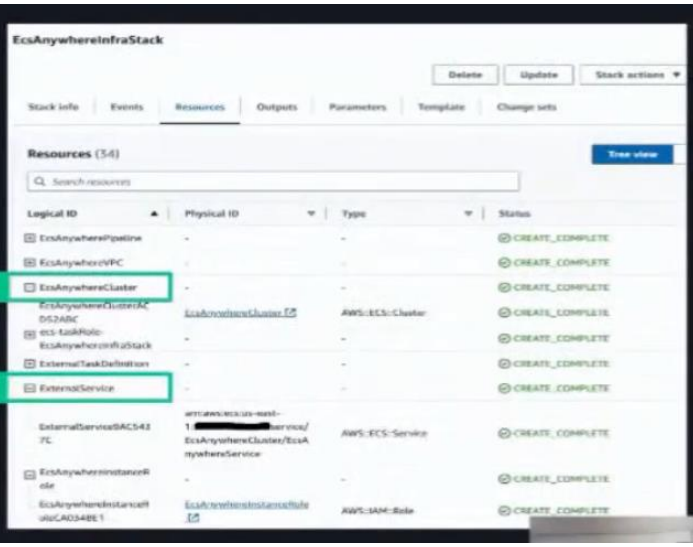
IAM Policy Changes

	Resource	Managed Policy ARN
+	${EcsAnywhereInstanceRole}	arn:${AWS::Partition}:iam::aws:policy/AmazonSSMManagedInstanceCore
+	${EcsAnywhereInstanceRole}	arn:aws:iam::aws:policy/service-role/AmazonEC2ContainerServiceforEC2Role
+	${ecs-taskRole-EcsAnywhereInfraStack}	arn:${AWS::Partition}:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
+	${ecs-taskRole-EcsAnywhereInfraStack}	arn:${AWS::Partition}:iam::aws:policy/AWSXRayDaemonWriteAccess

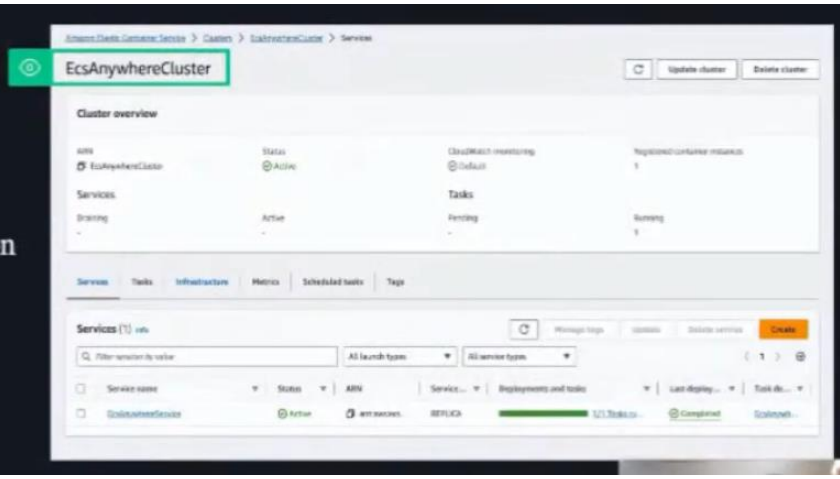
(NOTE: There may be security-related changes not in this list. See https://github.com/aws/aws-cdk/issues/1299)

```
Do you wish to deploy these changes (y/n)? y
EcsAnywhereInfraStack: deploying... [1/1]
EcsAnywhereInfraStack: creating CloudFormation changeset...

 ✅  EcsAnywhereInfraStack

✨  Deployment time: 183.54s
```

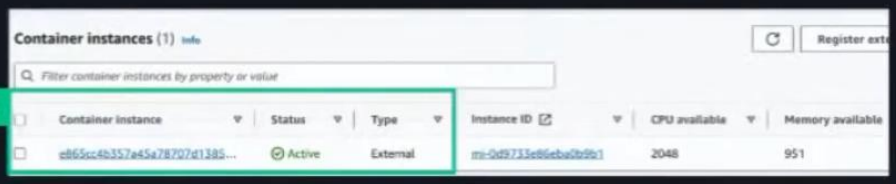
10# CloudFormation Stack



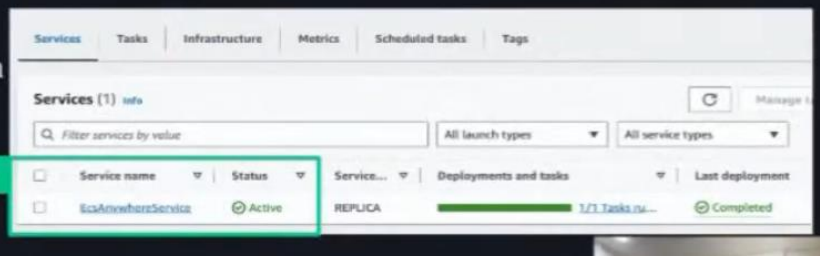
11# ECS Cluster Creation



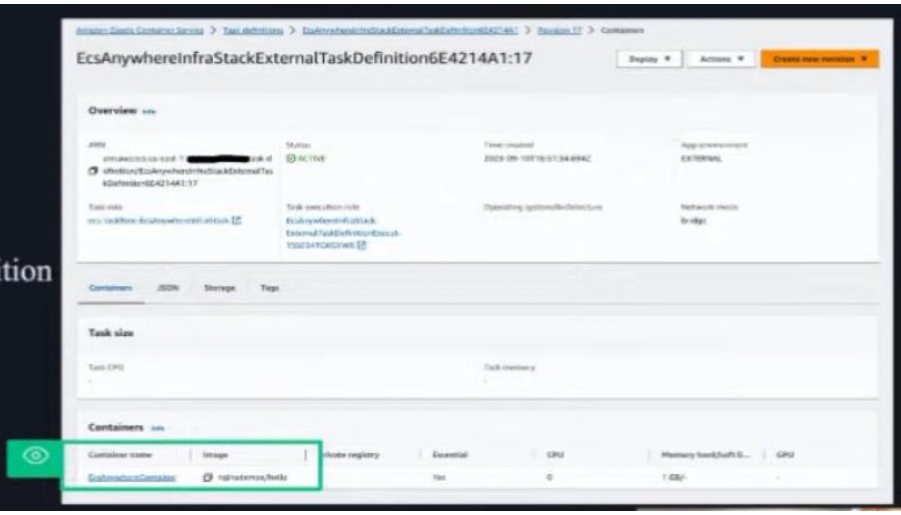
12# Configure External Instance(On-Premises)



13# ECS Service Creation



14# ECS Task Definition



Registering External Instance(On-Premises)

15# Install Vagrant-VMWare-Desktop

```
brew install vagrant-vmware-utility
vagrant global-status
vagrant plugin install vagrant-vmware-desktop
```

16# Vagrant file

```
Vagrant.configure("2") do |config|
  config.vm.box = "bento/ubuntu-20.04-arm64"
  config.vm.network "forwarded_port", guest: 80, host: 80
  config.vm.provider "vmware_desktop" do |vb|
    vb.memory = 2048
    vb.cpus = 2
    vb.gui = true
  end
end
```

17# Vagrant up

```
Ubuntu 20.04.6 LTS vagrant tty1

vagrant login: vagrant
Password:
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-153-generic aarch64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Mon 11 Sep 2023 05:56:42 PM UTC

  System load:  1.44               Processes:             240
  Usage of /:   11.6% of 29.82GB   Users logged in:       0
  Memory usage: 10%                IPv4 address for eth0: 192.168.234.133
  Swap usage:   0%

This system is built by the Bento project by Chef Software
More information can be found at https://github.com/chef/bento
^C
vagrant@vagrant:~$ pwd
/home/vagrant
vagrant@vagrant:~$ _
```

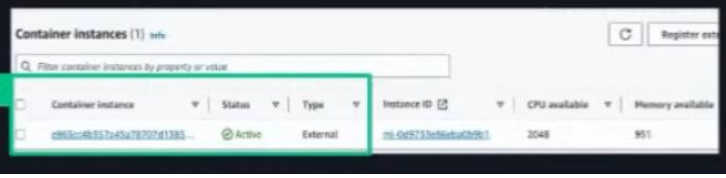
18# Vagrant ssh

```
Welcome to Ubuntu 20.04.6 LTS (GNU/Linux 5.4.0-153-generic aarch64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

  System information as of Sun 10 Sep 2023 05:22:23 PM UTC

  System load:  0.0                Processes:             221
  Usage of /:   16.0% of 29.82GB   Users logged in:       1
  Memory usage: 21%                IPv4 address for docker0: 172.17.0.1
  Swap usage:   0%                 IPv4 address for eth0:    192.168.234.132

This system is built by the Bento project by Chef Software
More information can be found at https://github.com/chef/bento
Last login: Sun Sep 10 22:16:31 2023 from 192.168.234.1
```

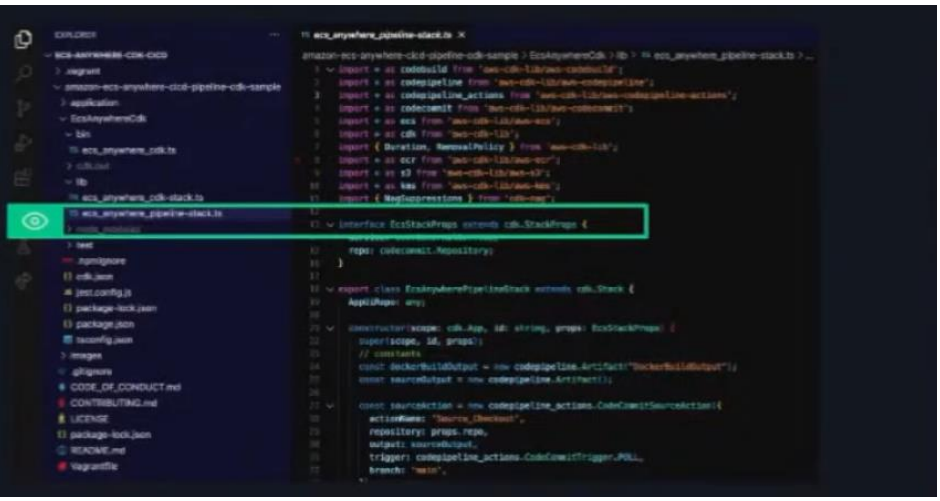
19# Register External Instance

```
curl --proto "https" -o "/tmp/ecs-anywhere-install.sh" "https://amazon-ecs-agent.s3.amazonaws.com/ecs-anywhere-install-latest.sh" && bash /tmp/ecs-anywhere-install.sh --region "us-east-1" --cluster "EcsAnywhereCluster" --activation-id "**********" --activation-code "********"
```

20# External Instance



21# CDK Pipeline Stack

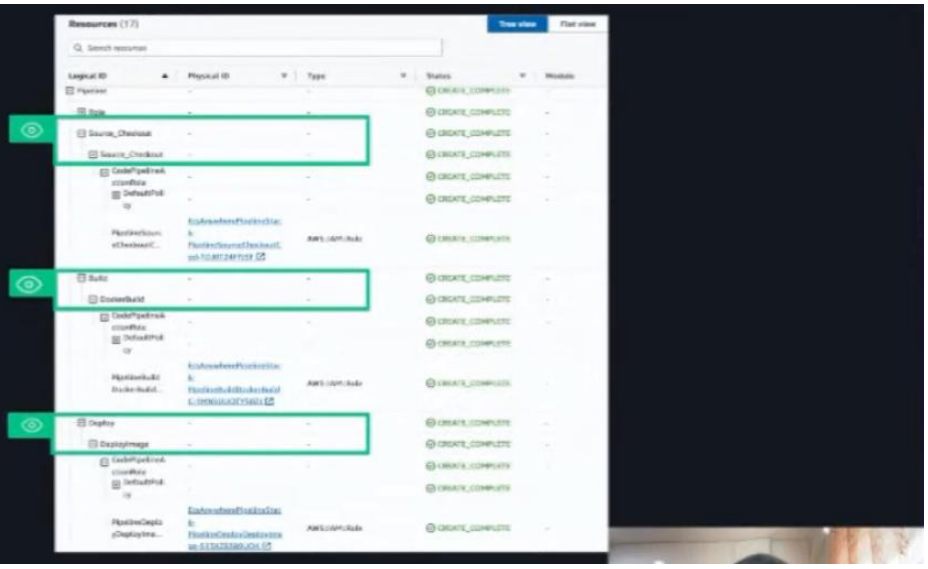


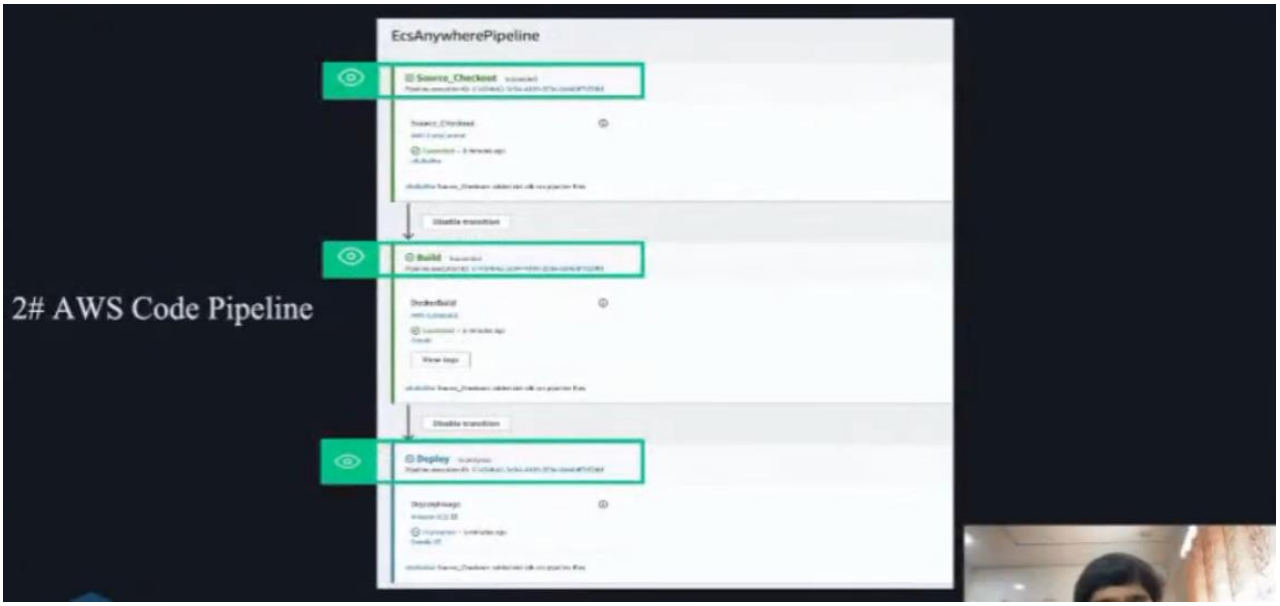
Setup CI/CD Pipeline

22# cdk deploy EcsAnywherePipelineStack

```
✨  Total time: 4.83s

EcsAnywherePipelineStack
EcsAnywherePipelineStack: deploying... [2/2]
EcsAnywherePipelineStack: creating CloudFormation changeset...

 ✅  EcsAnywherePipelineStack

✨  Deployment time: 193.92s
```

23# Pipeline CloudFormation Stack





References / Call for Actions

- https://docs.aws.amazon.com/prescriptive-guidance/latest/patterns/set-up-a-ci-cd-pipeline-for-hybrid-workloads-on-amazon-ecs-anywhere-by-using-aws-cdk-and-gitlab.html?did=pg_card&trk=pg_card
- https://github.com/aws-samples/aws-ecs-anywhere-workshop-samples
- https://docs.aws.amazon.com/cdk/
- https://docs.aws.amazon.com/AmazonECS/latest/developerguide/ecs-anywhere.html#ecs-anywhere-considerations