

Deployment of the PHP Data Entry Application can be performed in 2 stages.

1. Deployment of the backend MySQL database server

The DB Server docker file generates a MySQL Docker container with custom schema and data given in `CRUD.SQL`. `Dockerfile` and `buildspec.yml` are stored in CodeCommit. The CodeBuild generates the custom MySQL docker image and stores in ECR. A two-stage pipeline has been implemented to connect the CodeCommit repository with the CodeBuild project.

A user-data startup script (`db_ec2_inst_prereq.sh`) has been implemented to install Docker and CLI tools during the DB Server EC2 instance initialization. The script also pulls the docker image from the ECR and builds the MySQL container with the ‘always’ restart policy and port mapping of 3603:3603. A boot-time script `copy_db_config` is also created to copy the DB Environment information (`DB_HOST`, `DB_USER`, `DB_PASSWORD`) to an S3 object. (`s3://balagurusamy/db-server.env`). These environment variables are passed to the PHP application containers by the ECS service while creating tasks. This S3 object is specified during the task definition of PHP App Server Containers.

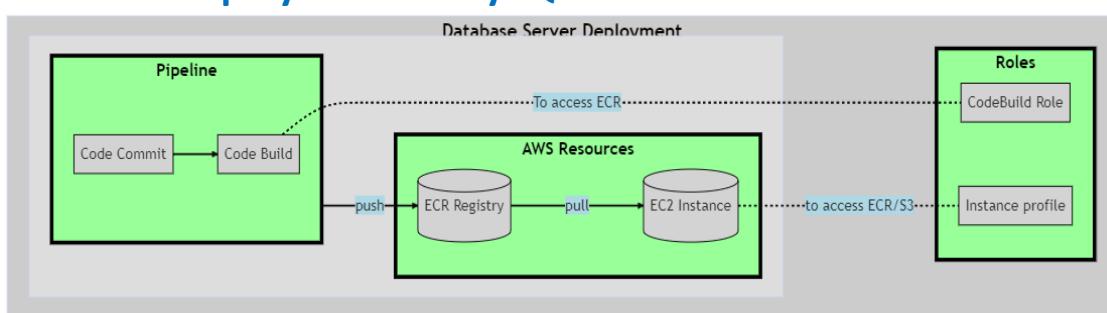
2. Deployment of the PHP Application server

The app-server docker file pulls the PHP image from the docker hub, installs and enables the `mysqli` extension, configures the `php.ini` file. The `backend.php` file has been modified to get DB credentials from the environment variables `DB_HOST`, `DB_USER`, `DB_PASSWORD`. `Dockerfile`, `buildspec.yml` and modified PHP application source files are

checked in to CodeCommit. The CodeBuild stage generates the docker image and stores it in ECR and creates an artifact file: **imagedefinitions.json** This artifact file is used by the CodeDeploy service to identify the containers for the generated images. A Three-Stage CodePipeline has been implemented to automate the build and deployment process.

A load balancer is created with a target group with port mapping 80:80. A task definition has been created for the PHP container with the environment file: **s3://balagurusamy/db-server.env (DB_HOST, DB_USER, DB_PASSWORD)** generated by Database Server Instance. An ECS cluster has been created along with a FARGATE service to use the load balancer and its target group for running the tasks.

STAGE 1: Deployment of MySQL Database Container



Step 1: Implementation of Dockerfile

Contents of Dockerfile

```
FROM mysql:8.0.32
```

```
COPY ./crud.sql /docker-entrypoint-initdb.d
```

This pulls mysql:8.0.32, copies the **crud.sql** to **/docker-entrypoint-initdb.d**. The files from this folder will be executed when the container is run very first time.

Step 2: Implementation of buildspec.yml

```
version: 0.2

phases:
  pre_build:
    commands:
      - echo Logging in to Amazon ECR...
#This logs in to AWS ECR
      - aws ecr get-login-password --region $AWS_DEFAULT_REGION | docker login --username AWS --password-stdin $AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com
  build:
    commands:
      - echo Build started on `date`
      - echo Building the Docker image...
#To build the docker container and store it in ECR
      - docker build -t $IMAGE_REPO_NAME:$IMAGE_TAG .
      - docker tag $IMAGE_REPO_NAME:$IMAGE_TAG
$AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com/$IMAGE_REPO_NAME:$IMAGE_TAG
      - printf '[{"name":"project2-db-docker","imageUri":"%s"}]'
```

\$AWS_ACCOUNT_ID.dkr.ecr.\$AWS_DEFAULT_REGION.amazonaws.com/\$IMAGE_REPO_NAME:\$IMAGE_TAG
> imagedefinitions.json

```
  post_build:
    commands:
      - echo Build completed on `date`
      - echo Pushing the Docker image...
      - docker push
$AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com/$IMAGE_REPO_NAME:$IMAGE_TAG
artifacts:
  files:
    - imagedefinitions.json
```

Step 3: Create a CodeCommit repository project2-dbserver and check in Dockerfile, buildspec.yml, crud.sql ,db_ec2_inst_prereq.sh (startup script for DB Server EC2 instance)

The screenshot shows the AWS CodeCommit interface. On the left, the navigation pane is visible with sections like Developer Tools, CodeCommit, Source, Code, Pull requests, Commits, Branches, Git tags, Settings, Approval rule templates, Artifacts, and Build. The main content area displays the 'project2-dbserver' repository. The 'Info' tab is selected, showing a list of files: buildspec.yml, crud.sql, db_ec2_inst_prereq.sh, and Dockerfile. At the top right, there are buttons for Notify, master, Create pull request, and Clone URL.

Step 4: Create a Repository in Elastic Container Registry: project2-dbserver

Screenshot of the project2-dbserver ECR registry

The screenshot shows the AWS Amazon Elastic Container Registry (ECR) interface. The left sidebar includes sections for Amazon Elastic Container Registry, Private registry, Public registry, and Repositories. Under Repositories, there are links for Getting started, Documentation, and Public gallery. The main area displays the 'Private repositories (5)' section. A search bar shows 'project2-dbse'. A table lists the repository details:

Repository name	URI	Created at	Tag immutability	Scan frequency	Encryption type	Pull through cache
project2-dbserver	506160996768.dkr.ecr.us-east-1.amazonaws.com/project2-dbserver	February 24, 2023, 21:44:41 (UTC-05)	Disabled	Manual	AES-256	Inactive

Step 5: Create a CodePipeline: project2-db-pipeline

Screenshots of pipeline and its details

The screenshot shows the AWS CodePipeline Pipelines list. A search bar at the top contains the text 'db'. Below it is a table with columns: Name, Most recent execution, Latest source revisions, and Last executed. One row is visible, showing 'project2-db-pipeline' with a green 'Succeeded' status, the latest source revision 'Source - 2B2ae455: initial version of Dockerfile and buildspec.yml for MySQL container', and '2 days ago' as the last execution time.

The screenshot shows the 'Edit action' configuration for the 'Source' stage. It includes fields for Action name ('Source'), Action provider ('AWS CodeCommit'), Repository name ('Q project2-dbserver'), Branch name ('Q master'), Change detection options (selected 'Amazon CloudWatch Events (recommended)'), Output artifact format (selected 'CodePipeline default'), Variable namespace ('SourceVariables'), and Output artifacts ('SourceArtifact'). Buttons for 'Cancel' and 'Done' are at the bottom right.

CodeBuild Stage

Action name
Choose a name for your action
Build
No more than 100 characters

Action provider
AWS CodeBuild

Region
US East (N. Virginia)

Input artifacts
Choose an input artifact for this action. [Learn more](#)
SourceArtifact
Add
No more than 100 characters

Project name
Choose a build project that you have already created in the AWS CodeBuild console. Or create a build project in the AWS CodeBuild console and then return to this task.
project2-dbserver or **Create project**

Environment variables - optional
Choose the key, value, and type for your CodeBuild environment variables. In the value field, you can reference variables generated by CodePipeline. [Learn more](#)
Add environment variable

Build type
 Single build
Triggers a single build.
 Batch build
Triggers multiple builds as a single execution.

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](#)
BuildVariables

Output artifacts
Choose a name for the output of this action.
BuildArtifact
Add
No more than 100 characters

Cancel **Done**

Successful pipeline workflow. This creates MySQL docker image in the project2-dbserver repo in ECR

Developer Tools **CodePipeline** [Alt+S]

Services N. Virginia | sudhakar @ 5061-6099-6768

project2-db-pipeline

Source **Succeeded**
Pipeline execution ID: [8f7d2cfb-1ffa-4f1a-8249-642ec3d30a54](#)

Source
AWS CodeCommit
Succeeded - 2 days ago
282ae455

Disable transition

Build **Succeeded**
Pipeline execution ID: [8f7d2cfb-1ffa-4f1a-8249-642ec3d30a54](#)

Build
AWS CodeBuild
Succeeded - 2 days ago
Details

Step 6: Create an IAM Role for the DB Server EC2 Instance (to access ECR, S3,CodeDeploy and other services)

The screenshot shows the AWS IAM Roles page. The left sidebar is titled "Identity and Access Management (IAM)" and includes sections for "Access management" (User groups, Users, Roles, Policies, Identity providers, Account settings), "Access reports" (Access analyzer, Archive rules, Analyzers, Settings, Credential report, Organization activity, Service control policies (SCPs)), and "Related consoles" (IAM Identity Center). The main content area shows the "Proj2CodeDeploy-EC2-Instance-Profile" role under "Roles". The "Summary" tab is selected, displaying details like Creation date (February 20, 2023, 14:15 UTC-05:00), ARN (arn:aws:iam:506160996768:role/Proj2CodeDeploy-EC2-Instance-Profile), Last activity (12 hours ago), and Maximum session duration (1 hour). Below the summary is a "Permissions" tab, which lists "Permissions policies (11) Info" (You can attach up to 10 managed policies). A table shows the attached policies, all of which are AWS managed:

Policy name	Type	Description
AmazonEC2FullAccess	AWS managed	Provides full access to Amazon EC2
AmazonS3FullAccess	AWS managed	Provides full access to Amazon S3
AmazonEC2ContainerRegistryFullAccess	AWS managed	Provides administrative access to Amazon ECR
AmazonS3FullAccess	AWS managed	Provides full access to all buckets via S3
AWSCodeDeployFullAccess	AWS managed	Provides full access to CodeDeploy
CloudWatchLogsFullAccess	AWS managed	Provides full access to CloudWatch Logs

Step 7: Create Security Group: ssh-db-port (to grant access to SSH 22 & MySQL 3306 ports)

Screenshot of Security Group

The screenshot shows the AWS Security Groups page. The left sidebar includes sections for "EC2 Dashboard", "EC2 Global View", "Events", "Tags", "Limits", "Instances" (Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Scheduled Instances, Capacity Reservations), "Images" (AMIs, AMI Catalog), and "Elastic Block Store" (Volumes, Snapshots, Lifecycle Manager). The main content area shows the "Security Groups (1/1) info" section. It displays a table for the "ssh-db-port" security group, which has a VPC ID of vpc-0776b69936ebe8a6b, an owner of 506160996768, and 2 permission entries. Below this is a detailed view for the "sg-00e444a05094bc594 - ssh-db-port" group, showing the "Inbound rules" tab with two rules:

Name	Security group rule...	IP version	Type	Protocol	Port range	Source
-	sgr-010c5582b32268...	IPv4	SSH	TCP	22	0.0.0.0/0
-	sgr-01de4a31fed7106d	IPv4	MySQL/Aurora	TCP	3306	0.0.0.0/0

Step 8: Implementation of startup script for the DB Server EC2 instance

This script will be executed when the EC2 instance is created. This installs the docker, AWS CLI, creates **copy_db_config** script to copy the DB **environment variables** (**DB_HOST, DB_USER, DB_PASSWORD**) to an S3 object. (s3://balagurusamy/db-server.env). This will be used by the PHP application container.

Contents of the script: **db_ec2_inst_prereq.sh**

```
#!/bin/bash
yum update -y
#install docker
amazon-linux-extras install -y docker
systemctl enable docker
usermod -aG docker ec2-user
#install CLI
yum install awscli -y
#build the docker container with restart policy-always
docker run --name db-docker --restart always -e MYSQL_ROOT_PASSWORD="bluehills" -d -p 3306:3306 ACCTNO.dkr.ecr.us-east-1.amazonaws.com/project2-dbserver:latest
#Create a boot-time script: copy_db_config to store the db credentials in S3
cat << EOM > /var/lib/cloud/scripts/per-boot/copy_db_config
#!/bin/bash
public_ip=$(curl -s http://169.254.169.254/latest/meta-data/public-ipv4)
echo "DB_HOST=$public_ip" > /tmp/db-server.env
echo "DB_USER=root" >> /tmp/db-server.env
echo "DB_PASSWORD=bluehills" >> /tmp/db-server.env
echo "DB_DATABASE=user" >> /tmp/db-server.env
aws s3 cp /tmp/db-server.env s3://balagurusamy
EOM
#Execute the copy_db_config
chmod +x /var/lib/cloud/scripts/per-boot/copy_db_config
/var/lib/cloud/scripts/per-boot/copy_db_config
```

Step 9: Launch an EC2 instance for DB Server with ssh/mysql ports enabled and initialize the instance with the user-data script: **db_ec2_inst_prereq.sh**

Instance summary for i-0aac59c6fd81884bb (proj2-dbserver) [Info](#)
Updated less than a minute ago

Instance ID	i-0aac59c6fd81884bb (proj2-dbserver)	Public IPv4 address	44.204.117.51 open address	Private IPv4 addresses	172.31.12.221
IPv6 address	-	Instance state	Running	Public IPv4 DNS	ec2-44-204-117-51.compute-1.amazonaws.com open address
Hostname type	IP name: ip-172-31-12-221.ec2.internal	Private IP DNS name (IPv4 only)	ip-172-31-12-221.ec2.internal	Elastic IP addresses	-
IP name:	ip-172-31-12-221.ec2.internal	Instance type	t2.micro	AWS Compute Optimizer finding	Opt-in to AWS Compute Optimizer for recommendations. Learn more
Answer private resource DNS name	IPv4 (A)	VPC ID	vpc-0776b69936ebe8a6b (default-vpc)	Auto Scaling Group name	-
Auto-assigned IP address	44.204.117.51 [Public IP]	Subnet ID	subnet-0ad1b2c637fb0c42	State transition reason	-
IAM Role	Proj2CodeDeploy-EC2-Instance-Profile	Kernel ID	-	State transition message	Client.UserInitiatedShutdown: User initiated shutdown

Details **Security** **Networking** **Storage** **Status checks** **Monitoring** **Tags**

Instance details [Info](#)

Platform	Amazon Linux (Inferred)	AMI ID	ami-0dfcb1ef8550277af	Monitoring	disabled
Platform details	Linux/UNIX	AMI name	amzn2-ami-kernel-5.10-hvm-2.0.20230207.0-x86_64-gp2	Termination protection	Disabled
Stop protection	Disabled	Launch time	Mon Feb 27 2023 14:43:28 GMT-0500 (Eastern Standard Time) (1 minute)	AMI location	amazon/amzn2-ami-kernel-5.10-hvm-2.0.20230207.0-x86_64-gp2
Instance auto-recovery	Default	Lifecycle	normal	Stop-hibernate behavior	disabled
AMI Launch index	0	Key pair name	my-ssh-key	State transition reason	-
Credit specification	standard	Kernel ID	-	State transition message	Client.UserInitiatedShutdown: User initiated shutdown

Security

Inbound rules

Name	Security group rule ID	Port range	Protocol	Source	Security groups
-	sgr-010c5582b3226884b	22	TCP	0.0.0.0/0	ssh-db-port
-	sgr-01de4a31fec7106d	3306	TCP	0.0.0.0/0	ssh-db-port

Outbound rules

**Step 10: Log in to DB server and verify the MySQL container installation.
Login to EC2 instance**

```

root@UrsaMajor:~ x root@ip-172-31-12-221:/home x + | v
(mypython) sudhakar@UrsaMajor:~$ ssh -i ".ssh/my-ssh-key.pem" ec2-user@ec2-44-204-117-51.compute-1.amazonaws.com
Last login: Mon Feb 27 20:08:49 2023 from c-73-159-171-229.hsd1.ma.comcast.net
--| --|_) 
_|| ( / Amazon Linux 2 AMI
__\_\_|___|
https://aws.amazon.com/amazon-linux-2/
[ec2-user@ip-172-31-12-221 ~]$ sudo -s
Check the installation status using grep /var/log/cloud-init-output.log for docker, ecr and db-server.env
[root@ip-172-31-12-221 log]# grep docker cloud-init-output.log
Cleaning repos: amzn2-core amzn2extra-docker amzn2extra-kernel-5.10
--> Package docker.x86_64 0:20.10.17-1.amzn2.0.2 will be installed
--> Processing Dependency: runc >= 1.0.0 for package: docker-20.10.17-1.amzn2.0.2.x86_64
--> Processing Dependency: libcgroup >= 0.40.rc1-5.15 for package: docker-20.10.17-1.amzn2.0.2.x86_64
--> Processing Dependency: containerd >= 1.3.2 for package: docker-20.10.17-1.amzn2.0.2.x86_64
--> Processing Dependency: pigz for package: docker-20.10.17-1.amzn2.0.2.x86_64
docker x86_64 20.10.17-1.amzn2.0.2 amzn2extra-docker 39 M
containerd x86_64 1.6.8-1.amzn2.0.1 amzn2extra-docker 27 M
runc x86_64 1.1.4-1.amzn2.0.1 amzn2extra-docker 2.9 M
Installing : docker-20.10.17-1.amzn2.0.2.x86_64 5/5
Verifying : docker-20.10.17-1.amzn2.0.2.x86_64 4/5
docker.x86_64 0:20.10.17-1.amzn2.0.2
Installing docker
20 docker=latest enabled \
Created symlink from /etc/systemd/system/multi-user.target.wants/docker.service to /usr/lib/systemd/system/docker.service.
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
https://docs.docker.com/engine/reference/commandline/login/#credentials-store
[root@ip-172-31-12-221 log]# grep ecr cloud-init-output.log
Unable to find image '506160996768.dkr.ecr.us-east-1.amazonaws.com/project2-dbserver:latest' locally
Status: Downloaded newer image for 506160996768.dkr.ecr.us-east-1.amazonaws.com/project2-dbserver:latest
[root@ip-172-31-12-221 log]# grep db-server cloud-init-output.log
upload: tmp/db-server.env to s3://balagurusamy/db-server.env
upload: tmp/db-server.env to s3://balagurusamy/db-server.env
upload: tmp/db-server.env to s3://balagurusamy/db-server.env
upload: tmp/db-server.env to s3://balagurusamy/db-server.env
[root@ip-172-31-12-221 log]#
Execute bash script on the MySQL container to check the execution status of crud.sql
[root@ip-172-31-12-221 ec2-user]# docker exec -it db-docker bash
bash-4.4# mysql -pbluehills
mysql: [Warning] Using a password on the command line interface can be insecure.
Welcome to the MySQL monitor. Commands end with ; or \g.
Your MySQL connection id is 10
Server version: 8.0.32 MySQL Community Server - GPL

Copyright (c) 2000, 2023, Oracle and/or its affiliates.

Oracle is a registered trademark of Oracle Corporation and/or its
affiliates. Other names may be trademarks of their respective
owners.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

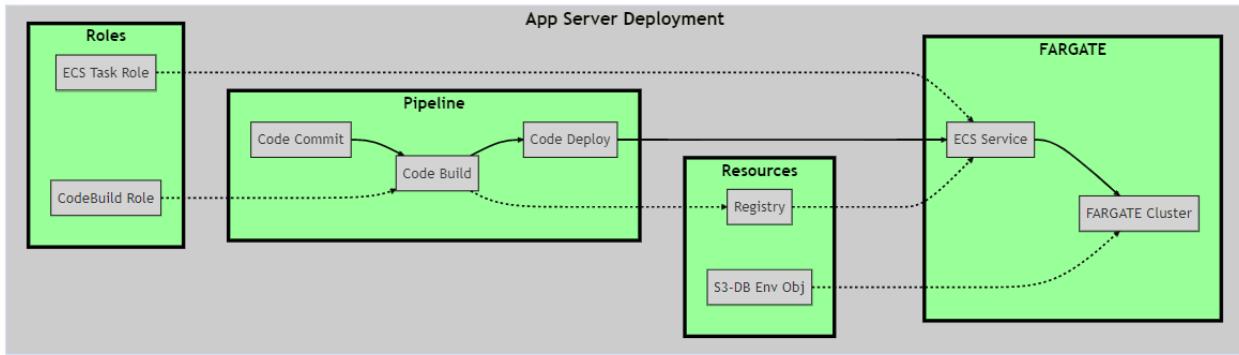
mysql> show databases;
+-----+
| Database |
+-----+
| information_schema |
| mysql |
| performance_schema |
| sys |
| user |
+-----+
5 rows in set (0.00 sec)

mysql> select * from user.crud;
+----+-----+-----+-----+-----+
| id | name | email | phone | city |
+----+-----+-----+-----+-----+
| 40 | Mahatma gandhi | gandhi@gandhi.com | 9191919191 | New Delhi |
| 42 | Issac Newton | newton@newton.com | 44444444 | London |
| 44 | sudhakar balagurusamy | whitesnow@aol.com | 7814921488 | South Weymouth |
+----+-----+-----+-----+-----+
3 rows in set (0.00 sec)

mysql>

```

Stage 2: Deployment of the Data Entry PHP application server



Step 1: Implementation of Dockerfile

Implement a **Dockerfile** to pull the latest PHP image, configure the php.ini, install the MYSQLI extension, add the PHP pages

Contents of Dockerfile

```
FROM php:8.1.16-apache
# Copy the Application files to temporary folder /tmp/app
COPY ./configure ./index.php /tmp/app/
COPY ./ajax/ /tmp/app/ajax/
COPY ./backend/ /tmp/app/backend/
COPY ./css/ /tmp/app/css/
# Run the configure script to enable and install mysqli extension
# configure script also copies the application files to appropriate html
folders
RUN /bin/bash /tmp/app/configure
```

This pulls the php:8.1.16, copies the application files and the configure script to temporary folder and runs the configure script.

Contents of the script: configure.

```
# Copy the application files to /var/www/html folder
cp -r /tmp/app/ajax/ /var/www/html
cp -r /tmp/app/backend/ /var/www/html
cp -r /tmp/app/css/ /var/www/html
# Copy the modified index.php
cp /tmp/app/index.php /var/www/html/
# Create the php.ini configuration file
cp /var/www/html/php.ini-production /var/www/html/php.ini
# Enable mysqli extension to access the MySQL database server
sed -i 's/^;extension=mysqli/extension=mysqli/g' /var/www/html/php.ini
# Install the mysqli extension
docker-php-ext-install mysqli
```

This copies the application files to html folder and enables and installs mysqli extension module.

Contents of backend/database.php

```
<?php  
  
// Create connection  
$conn = mysqli_connect(getenv("DB_HOST"), getenv("DB_USER"),  
getenv("DB_PASSWORD"), getenv("DB_DATABASE"));  
// Check connection  
if (!$conn) {  
    die("Connection failed: " . mysqli_connect_error());  
}  
?>
```

Database.php has been modified to retrieve DB INFO from the environment variables: DB_HOST, DB_USER, DB_PASSWORD.

Step 2: Implementation of buildspec.yml

Contents of buildspec.yml

This is used by CodeBuild application to build the container and store it in the ECR. The **IAM role** used for the CodeBuild application needs to have write access to ECR (Elastic Container Registry). This also generates an artifact file **imagedefinitions.json** used by the Elastic Container Service to deploy the generated image to the appropriate FARGATE cluster containers.

```
version: 0.2
phases:
  pre_build:
    commands:
      - echo Logging in to Amazon ECR...
      - aws ecr get-login-password --region $AWS_DEFAULT_REGION | docker login --username AWS --
password-stdin $AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com
  build:
    commands:
      - echo Build started on `date`
      - echo Building the Docker image...
      - docker build -t $IMAGE_REPO_NAME:$IMAGE_TAG .
      - docker tag $IMAGE_REPO_NAME:$IMAGE_TAG
$AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com/$IMAGE_REPO_NAME:$IMAGE_TAG
      - printf '[{"name":"project2-docker","imageUri":"%s"}]'
$AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com/$IMAGE_REPO_NAME:$IMAGE_TAG
> imagedefinitions.json
  post_build:
    commands:
      - echo Build completed on `date`
      - echo Pushing the Docker image...
      - docker push
$AWS_ACCOUNT_ID.dkr.ecr.$AWS_DEFAULT_REGION.amazonaws.com/$IMAGE_REPO_NAME:$IMAGE_TAG
artifacts:
  files:
    - imagedefinitions.json
```

Step 3: Create a CodeCommit repository and check in the Application files, Dockerfile and buildspec.yml

The screenshot shows two views of the AWS CodeCommit interface. The top view is a list of repositories under the 'Repositories' tab, showing a single repository named 'project2-appserver'. The bottom view is a detailed view of the 'project2-appserver' repository, showing its contents. The contents include several files and folders: 'ajax', 'backend', 'css', 'buildspec.yml', 'configure', 'Dockerfile', and 'index.php'. The 'Code' tab is selected in the sidebar.

Repositories info

Name	Description	Last modified
project2-appserver	-	1 day ago

Clone URL: [HTTPS](#) [SSH](#) [HTTPS \(GR\)](#)

project2-appserver

Notify: master Create pull request Clone URL

project2-appserver Info Add file

Name
ajax
backend
css
buildspec.yml
configure
Dockerfile
index.php

Step 4: Create a CodeBuild Project: project2-appserver.

Source Repo: project2-appserver, Environment variables : AWS_ACCOUNT_ID, IMAGE_TAG, IMAGE_REPO_NAME

The screenshot shows the AWS CodeBuild console interface. On the left, there is a navigation sidebar with the following structure:

- Developer Tools**
- CodeBuild** (selected)
- Source** • CodeCommit
- Artifacts** • CodeArtifact
- Build** • CodeBuild
 - Getting started
 - Build projects
 - Build project** (selected)
 - Settings
- Build history
- Report groups
- Report history
- Account metrics
- Deploy** • CodeDeploy
- Pipeline** • CodePipeline
- Settings
- Go to resource
- Feedback

The main content area displays the configuration for the "project2-appserver" build project. The "Build details" tab is selected. The configuration includes:

- Configuration** section:

Source provider: AWS CodeCommit	Primary repository: project2-appserver	Artifacts upload location: -	Build badge: Disabled
Public builds: Disabled			
- Project configuration** section:

Name: project2-appserver	Description: GL Project2 Deploying a data entry application on Containers
Project ARN: arn:aws:codebuild:us-east-1:506160996768:project/project2-appserver	Concurrent build limit: -
Tags: -	
- Source** section:

Source provider: AWS CodeCommit	Source identifier: -	Repository: project2-appserver	Source version: refs/heads/master
Git clone depth: 1	Git submodules: False		
- Build project** section (expanded):

Image: aws/codebuild/standard:4.0	Environment type: Linux	Compute: 3 GB memory, 2 vCPUs	Privileged: True
Service role: arn:aws:iam::506160996768:role/service-role/codebuild-project2-appserver-service-role	Timeout: 1 hour 0 minutes	Queued timeout: 8 hours 0 minutes	Certificate: -
Registry credential: -			
VPC			
Environment variables			
Name	Value	Type	
AWS_ACCOUNT_ID	506160996768	PLAINTEXT	
AWS_DEFAULT_REGION	us-east-1	PLAINTEXT	
IMAGE_TAG	latest	PLAINTEXT	
IMAGE_REPO_NAME	project2-appserver	PLAINTEXT	
File systems			
Buildspec			
Using the buildspec.yml in the source code root directory			
Batch configuration			
Batch service role: -	Allowed compute type(s) for batch: -	Maximum builds allowed in batch: -	Batch timeout: -
Combine artifacts: -			

At the bottom of the page, there are links for Feedback and Language, and a copyright notice: © 2023, Amazon Web Services.

Step 5: Create an Elastic Container Registry Repository: project2-appserver.

ACCTNO.dkr.ecr.us-east-1.amazonaws.com/project2-appserver

The screenshot shows the AWS ECR (Amazon Elastic Container Registry) interface. On the left, there's a sidebar with links like 'Private registry', 'Public registry', and 'Repositories'. The main area is titled 'Amazon ECR > Repositories' and has tabs for 'Private' and 'Public'. A search bar at the top right contains the text 'appser'. Below it, a table lists 'Private repositories (5)'. One row is highlighted, showing details for the repository 'project2-appserver'. The table columns include Repository name, URI, Created at, Tag immutability, Scan frequency, Encryption type, and Pull through cache. The 'Created at' column shows 'February 22, 2023, 16:44:10 (UTC-05)'. The 'Tag immutability' column shows 'Disabled'. The 'Scan frequency' column shows 'Manual'. The 'Encryption type' column shows 'AES-256'. The 'Pull through cache' column shows 'Inactive'.

Repository name	URI	Created at	Tag immutability	Scan frequency	Encryption type	Pull through cache
project2-appserver	506160996768.dkr.ecr.us-east-1.amazonaws.com/project2-appserver	February 22, 2023, 16:44:10 (UTC-05)	Disabled	Manual	AES-256	Inactive

Step 6: Create a target group: project2-appserver-tg

Create a load balancer in 2 Availability zones: project2-appserver-lb

Target Group Screenshot

The screenshot shows the AWS EC2 Target Groups interface. On the left, a navigation sidebar lists various EC2 services like Instance Types, Launch Templates, and Auto Scaling. The main content area is titled 'project2-appserver-tg'. It displays 'Details' for the target group, including its ARN, target type (Instance), protocol (HTTP: 80), and VPC configuration (VPC ID: vpc-0776b69936ebe8a6b). Below this, a table shows target counts: Total targets (0), Healthy (0), Unhealthy (0), Unused (0), Initial (0), and Draining (0). A tab bar at the bottom includes 'Targets' (selected), Monitoring, Health checks, Attributes, and Tags. The 'Registered targets' section below shows 'No registered targets' and a 'Register targets' button.

Load Balancer Screenshot

The screenshot shows the AWS EC2 Load Balancers interface. The left sidebar includes the same EC2 services as the previous screenshot. The main content area is titled 'Load balancers (1)'. It shows a table with one entry: 'project2-appserver-lb' (DNS name: project2-appserver-lb-201..., State: Active, VPC ID: vpc-0776b69936ebe8a6b, Availability Zones: 2, Type: application, Date created: February 23, 2023, 16 (UTC-05:00)). Below the table, a message says '0 load balancers selected' and 'Select a load balancer above.'

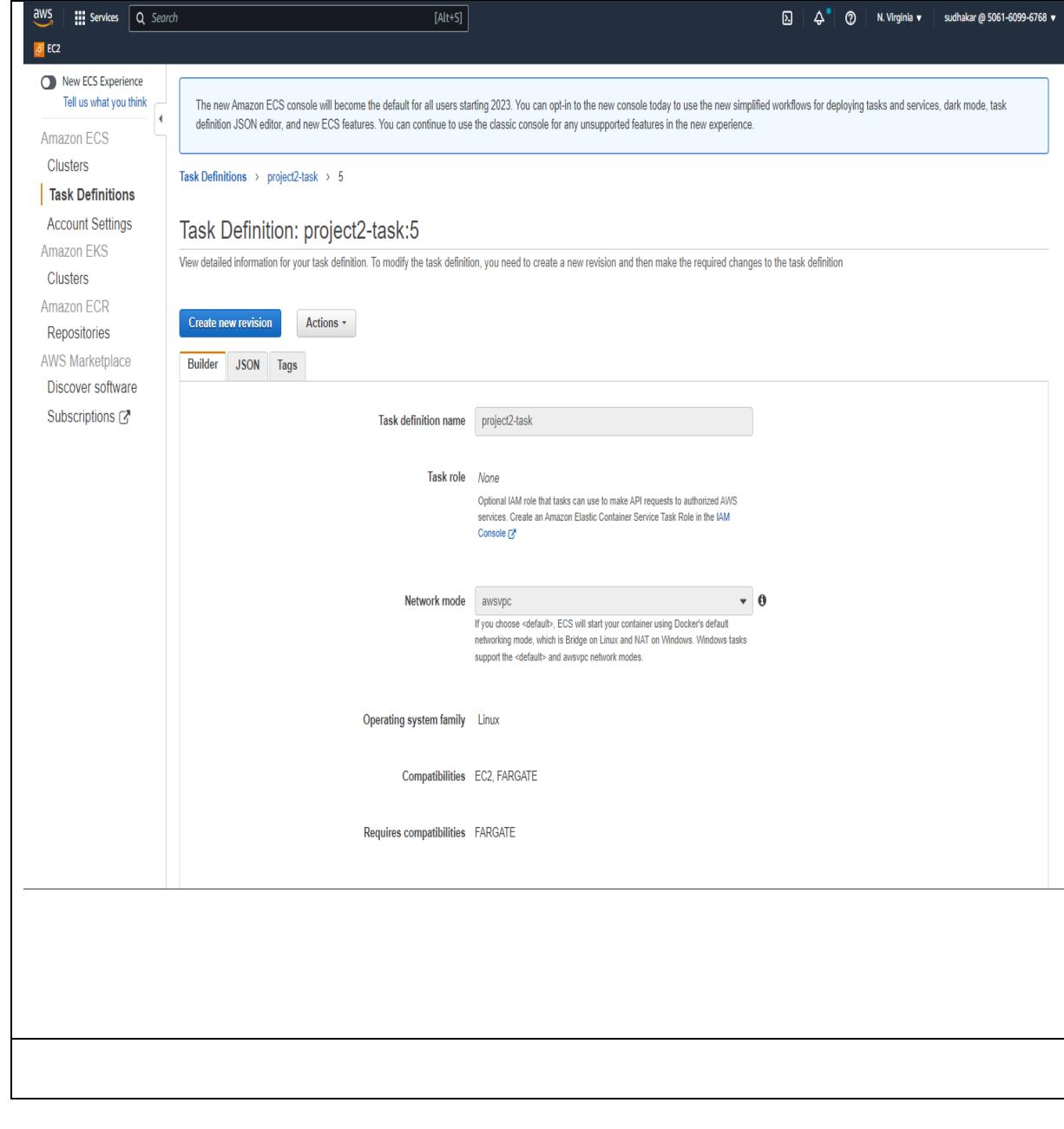
The screenshot shows the AWS EC2 Load Balancers console. On the left, a navigation sidebar lists various services like Instance Types, Launch Templates, and Auto Scaling. The main content area displays the details for a load balancer named 'project2-appserver-lb'. The 'Details' section shows the ARN, Load balancer type (Application), DNS name (project2-appserver-lb-2012189478.us-east-1.elb.amazonaws.com), Status (Active), VPC (vpc-0776b69936be8a6b), IP address type (IPv4), Scheme (Internet-facing), Availability Zones (us-east-1b, us-east-1a), Hosted zone (Z355XDOTRQ7X7K), and Date created (February 23, 2023, 16:33 (UTC-05:00)). Below this, the 'Listeners' tab is selected, showing one listener for port 80 (Protocol: HTTP) with a Forward rule pointing to a target group named 'project2-tg'. Other tabs include Network mapping, Security, Monitoring, Integrations, Attributes, and Tags.

Step 7: Create a cluster: project2-cluster

The screenshot shows the AWS ECS Clusters console. On the left, a navigation sidebar lists services like Amazon ECS, Clusters, Task Definitions, Account Settings, Amazon EKS, Amazon ECR, Amazon ECR Repositories, AWS Marketplace, Discover software, and Subscriptions. The main content area displays the details for a cluster named 'project2-cluster'. It shows the Cluster ARN (arn:aws:ecs:us-east-1:1506160996768:cluster/project2-cluster), Status (ACTIVE), Registered container instances (0), Pending tasks count (0 Fargate, 0 EC2, 0 External), Running tasks count (0 Fargate, 0 EC2, 0 External), Active service count (1 Fargate, 0 EC2, 0 External), and Draining service count (0 Fargate, 0 EC2, 0 External). Below this, there are tabs for Services, Tasks, ECS Instances, Metrics, Scheduled Tasks, Tags, and Capacity Providers. A 'Create' button is available at the bottom left, and a note indicates the new ECS console will become the default starting in 2023. The last update was on February 25, 2023, at 11:13:23 PM.

Step 8: Create a Task Definition: project2-task with
Create a Container definition : project2-docker with port mapping 80:80 with
environment file

Create a task definition with FARGATE in awsbatch mode



The screenshot shows the AWS ECS Task Definitions page. A modal window is open for creating a new task definition named "project2-task".

Task Definition: project2-task:5

Task definition name: project2-task

Task role: None

Network mode: awsvpc

Operating system family: Linux

Compatibilities: EC2, FARGATE

Requires compatibility: FARGATE

A message at the top of the modal states: "The new Amazon ECS console will become the default for all users starting 2023. You can opt-in to the new console today to use the new simplified workflows for deploying tasks and services, dark mode, task definition JSON editor, and new ECS features. You can continue to use the classic console for any unsupported features in the new experience."

aws Services Search [Alt+S] N. Virginia sudhakar @ 5061-6099-6768 ▾

EC2

Task execution IAM role

This role is required by tasks to pull container images and publish container logs to Amazon CloudWatch on your behalf. If you do not have the `ecsTaskExecutionRole` already, we can create one for you.

Task execution role `ecsTaskExecutionRole`

Task size

The task size allows you to specify a fixed size for your task. Task size is required for tasks using the Fargate launch type and is optional for the EC2 or External launch type. Container level memory settings are optional when task size is set. Task size is not supported for Windows containers.

Task memory (MiB) 512

Task CPU (unit) 256

Task memory maximum allocation for container memory reservation



0 512 shared of 512 MiB

Task CPU maximum allocation for containers



0 256 shared of 256 CPU units

Task Placement

Constraint No constraints

Container definitions

AWS Services Search [Alt+S] N. Virginia sudhakar @ 5061-6095

EC2

Container definitions

Container Name	Image	CPU Units	GPU	Inference Accelerator	Hard/Soft memory limits (MiB)	Essential
project2-docker	506160996768.dkr.ecr...	0	-/-	-/-	-/-	true

Details

Port Mappings

Host Port	Container Port	Protocol
80	80	tcp

Environment Variables

Key	Value/ValueFrom
No environment variables	

Environment Files

Source	Location
S3 ARN	arn:aws:s3:::balagurusamy/db-server.env

Container Ordering

Container Name	Condition
No container ordering	

Container Timeouts

Start timeout:
Stop timeout:

Docker labels

Key	Value
No docker labels	

Extra hosts

Hostname	IP address

Mount Points

Container Path	Source Volume	Read only
No mount points		

Volumes from

Source Container	Read only
No volumes from	

Ulimits

Name	Soft limit	Hard limit
No ulimit		

Elastic Inference

Accelerator

None

Log Configuration

Log driver: awslogs

Key	Value
awslogs-group	/ecs/project2-task
awslogs-region	us-east-1
awslogs-stream-prefix	ecs

Step 9: Create a Service in the cluster: project2-app-service

Number of Desired Tasks: 2

The screenshot shows the AWS Lambda service configuration interface. At the top, there's a navigation bar with the AWS logo, 'Services' (highlighted), a search bar, and a 'Cluster' dropdown set to 'project2-cluster'. Below the navigation, the main content area has a title 'Update Service'.

Step 4: Review (highlighted)

Review section details:

- Cluster: project2-cluster
- Launch type: FARGATE
- Operating system family: Linux
- Task Definition: project2-task:6
- Platform version: LATEST
- Service name: project2-app-service
- Service type: REPLICA
- Number of tasks: 2
- Minimum healthy percent: 100
- Maximum percent: 200
- Deployment circuit breaker: Disabled

Configure network

Configure network section details:

- Health check grace period: 0
- Allowed VPC: vpc-0776b69936ebe8a6b
- Allowed subnets: subnet-00019920ca2b47b3c,subnet-011ec34aa842b672f
- Security groups*: sg-05c00e63661d0e400
- Auto-assign public IP: ENABLED

Load balancing settings can only be set on service creation.

AWS Services Search [Alt+S] N. Virginia ▾ sudhakar@5061-6099-6768 ▾

EC2

Minimum healthy percent 100

Maximum percent 200

Deployment circuit breaker Disabled

Configure network Edit

Health check grace period 0

Allowed VPC [vpc-0776b69936ebe8a6b](#)

Allowed subnets [subnet-00019920ca2b47b3c](#), [subnet-011ec34aa842b672f](#)

Security groups* [sg-05c00e63661d0e400](#)

Auto-assign public IP ENABLED

Load balancing settings can only be set on service creation.

Load Balancer Name project2-appserver-lb

Container Name: project2-docker

Container Port: 80

Target Group: [arn:aws:elasticloadbalancing:us-east-1:506160996768:targetgroup:ecs-project2-app-service/b6ac3f294a7e468b](#)

Set Auto Scaling (optional) Edit

not configured

Cancel Previous Update Service

The new Amazon ECS console will become the default for all users starting 2023. You can opt-in to the new console today to use the new simplified workflows for deploying tasks and services. You can continue to use the classic console for any unsupported features in the new experience.

Clusters > project2-cluster > Service: project2-app-service

Service : project2-app-service

Cluster	project2-cluster	Desired count	2
Status	ACTIVE	Pending count	2
Task definition	project2-task:5	Running count	0
Service type	REPLICA		
Launch type	FARGATE		
Service role	AWSServiceRoleForECS		
Created By	arn:aws:iam::506160996768:user/sudhakar		

Details Tasks Events Auto Scaling Deployments Metrics Tags Logs

Task status: Running Stopped

Filter in this page

Task	Task Definition	Last status	Desired status	Group
124286107fb242b8922f326ffeb...	project2-task:5	RUNNING	RUNNING	service:project2-app-service
558ccd0ca4364011ade43aacfc3...	project2-task:5	RUNNING	RUNNING	service:project2-app-service

Step 10: Create a CodePipeline: project2-app-pipeline.

The screenshot shows the AWS CodePipeline console interface. The top navigation bar includes the AWS logo, Services (with EC2 selected), a search bar, and user information (N. Virginia, suhakar@50614). On the left, a sidebar titled "CodePipeline" lists various integration points: Source (CodeCommit), Artifacts (CodeArtifact), Build (CodeBuild), Deploy (CodeDeploy), and Pipeline (CodePipeline). Under Pipeline, there are links for "Getting started" and "Pipelines". The main content area shows the "Pipelines" list with one item: "project2-app-pipeline" (Succeeded, 4 hours ago). A prominent orange "Create pipeline" button is located at the top right of the list. The URL in the browser's address bar is [https://console.aws.amazon.com/codestar/pipelines?region=N. Virginia](#).

Name	Most recent execution	Latest source revisions	Last executed
project2-app-pipeline	Succeeded	Source - c3ac0a09: added comments	4 hours ago

CodeCommit Stage

aws Services Search [Alt+S]

Edit action

Action name
Choose a name for your action

Action provider

Repository name
Choose a repository that you have already created where you have pushed your source code.

Branch name
Choose a branch of the repository

Change detection options - *optional*
Choose a detection mode to automatically start your pipeline when a change occurs in the source code.

<input checked="" type="radio"/> Amazon CloudWatch Events (recommended) Use Amazon CloudWatch Events to automatically start my pipeline when a change occurs	<input type="radio"/> AWS CodePipeline Use AWS CodePipeline to check periodically for changes
---	--

Output artifact format - *optional*
Choose the output artifact format.

<input checked="" type="radio"/> CodePipeline default AWS CodePipeline uses the default zip format for artifacts in the pipeline. Does not include git metadata about the repository.	<input type="radio"/> Full clone AWS CodePipeline passes metadata about the repository that allows subsequent clone. Only supported for AWS CodeBuild actions.
--	---

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](#)

Output artifacts
Choose a name for the output of this action.

No more than 100 characters

Feedback

CodeBuild Stage

Action name
Choose a name for your action

Action provider

Region

Input artifacts
Choose an input artifact for this action. [Learn more](#)

SourceArtifact

No more than 100 characters

Project name
Choose a build project that you have already created in the AWS CodeBuild console. Or create a build project in the AWS CodeBuild console and then return to this task.

Environment variables - *optional*
Choose the key, value, and type for your CodeBuild environment variables. In the value field, you can reference variables generated by CodePipeline. [Learn more](#)

Add environment variable

Build type

Single build
Triggers a single build.

Batch build
Triggers multiple builds as a single execution.

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](#)

BuildVariables

Output artifacts
Choose a name for the output of this action.

BuildArtifact

No more than 100 characters

CodeDeploy Stage

AWS Services Search [Alt+S]

EC2

Develop

CodeDeploy

Sources

Artifacts

Builds

Deploys

Pipes

Getters

Pipes

Handlers

Settings

Go to

Feedback

Edit action

Action name
Choose a name for your action

No more than 100 characters

Action provider

Region

Input artifacts
Choose an input artifact for this action. [Learn more](#)

No more than 100 characters

Cluster name
Choose a cluster that you have already created in the Amazon ECS console. Or create a cluster in the Amazon ECS console and then return to this task.

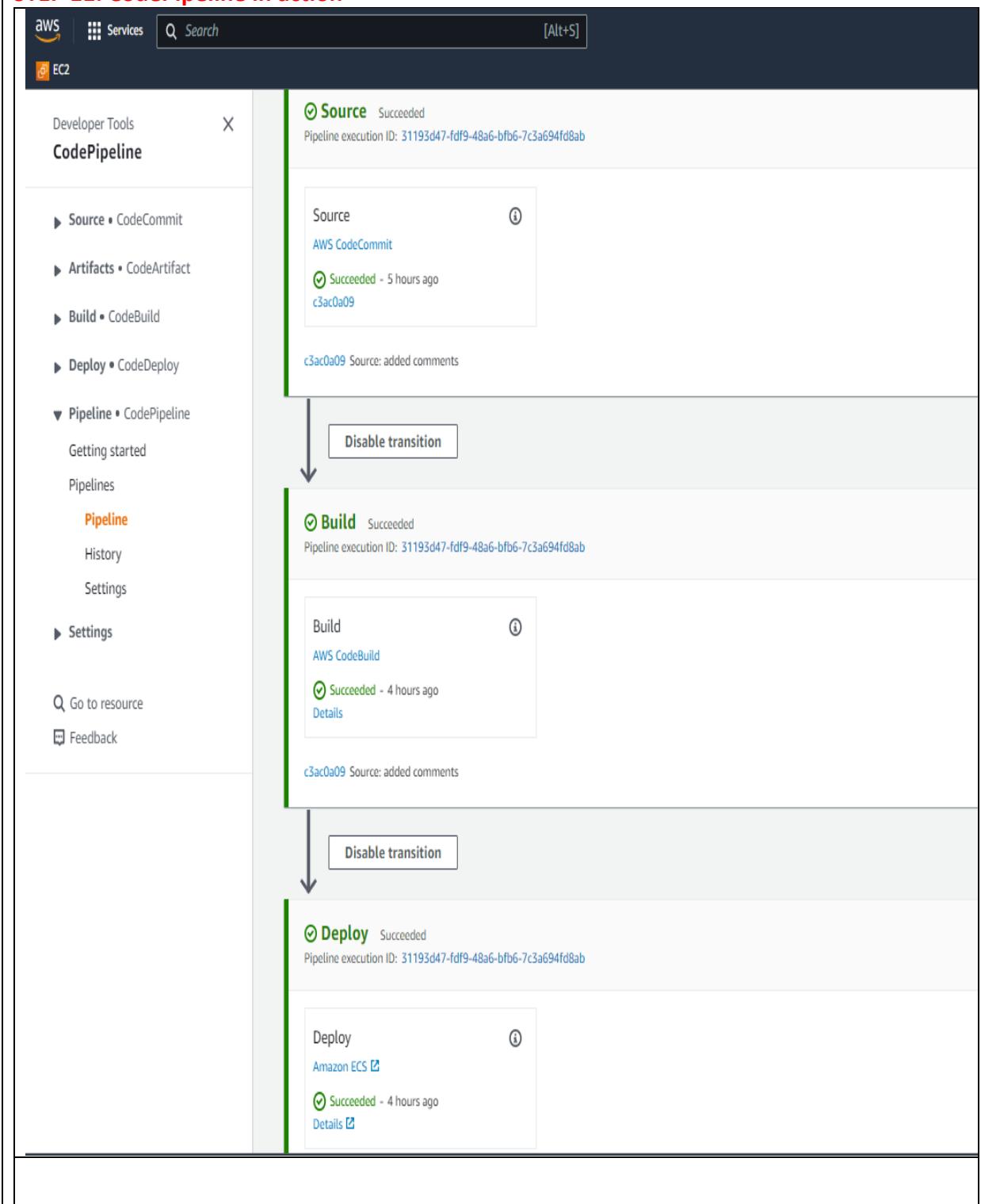
Service name
Choose a service that you have already created in the Amazon ECS console for your cluster. Or create a new service in the Amazon ECS console and then return to this task.

Image definitions file - *optional*
Enter the JSON file that describes your service's container name and the image and tag.

Deployment timeout - *optional*
Enter the timeout in minutes for the deployment action.

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](#)

STEP 11: CodePipeline in action



STEP 12: Cluster Deployment Status, Metrics

Running tasks : 2,

The screenshot shows the AWS ECS console interface. At the top, there's a navigation bar with the AWS logo, a 'Services' dropdown, a search bar, and a keyboard shortcut [Alt+S]. Below the navigation is a banner about the new ECS experience. The left sidebar has a 'Clusters' section selected, along with links for Task Definitions, Account Settings, Amazon EKS, Clusters, Amazon ECR, Repositories, AWS Marketplace, Discover software, and Subscriptions. The main content area shows the 'Service : project2-app-service' details. It lists the Cluster as 'project2-cluster', Status as 'ACTIVE', Task definition as 'project2-task:5', Service type as 'REPLICA', Launch type as 'FARGATE', Service role as 'AWSServiceRoleForECS', and Created By as 'arn:aws:iam:506160996768:user/sudhakar'. Below this, a tab navigation bar includes 'Details' (selected), 'Tasks', 'Events', 'Auto Scaling', 'Deployments', 'Metrics', 'Tags', and 'Logs'. Under the 'Tasks' tab, it says 'Task status: Running Stopped'. A 'Filter in this page' button is available. A table lists two tasks: one with ID 'a311af1efa9c45b6a6075d09f3467fde' and another with ID 'd0106abd008846acbff0d69db7fb354'. Both tasks have a 'Task Definition' of 'project2-task:5', a 'Last status' of 'RUNNING', and a 'Desired status' of 'RUNNING'. The table also includes a 'Group' column which is partially cut off.

Task	Task Definition	Last status	Desired status	Group
a311af1efa9c45b6a6075d09f3467fde	project2-task:5	RUNNING	RUNNING	service:proje
d0106abd008846acbff0d69db7fb354	project2-task:5	RUNNING	RUNNING	service:proje

aws Services Search [Alt+S]

EC2

Amazon ECS

Clusters

- Task Definitions
- Account Settings

Amazon EKS

- Clusters

Amazon ECR

- Repositories

AWS Marketplace

- Discover software
- Subscriptions ↗

classic console for any unsupported features in the new experience.

Clusters > project2-cluster > Service: project2-app-service

Service : project2-app-service

Cluster project2-cluster

Status ACTIVE

Task definition project2-task:5

Service type REPLICA

Launch type FARGATE

Service role AWSServiceRoleForECS

Created By arn:aws:iam::506160996768:user/sudhakar

Details Tasks Events Auto Scaling Deployments Metrics Tags Logs

Task Placement

Strategy No strategies

Constraint No constraints

Service Deployment Options

Minimum healthy percent 100 ⓘ

Maximum percent 200 ⓘ

Deployment circuit breaker Disabled ⓘ

[create pipeline ↗](#) | [view pipelines ↗](#)

Filter in this page

Deployment Id	Status	Desired count	Pending count
ecs-svc/009600332575910...	PRIMARY	2	0

AWS Services Search [Alt+S] N. Virginia sudhakar @ 5061-62

EC2

New ECS Experience Tell us what you think

The new Amazon ECS console will become the default for all users starting 2023. You can opt-in to the new console today to use the new simplified workflows for deploying tasks and services, dark mode, task definition JSON editor, and new ECS features. You can continue to use the classic console for any unsupported features in the new experience.

Amazon ECS

Clusters Task Definitions Account Settings

Amazon EKS Clusters Amazon ECR Repositories AWS Marketplace Discover software Subscriptions

Clusters > project2-cluster > Service: project2-app-service

Service : project2-app-service

Update

Cluster project2-cluster Desired count 2

Status ACTIVE Pending count 0

Task definition project2-task5 Running count 1

Service type REPLICA

Launch type FARGATE

Service role AWSServiceRoleForECS

Created By arn:aws:iam::50610996768:user/sudhakar

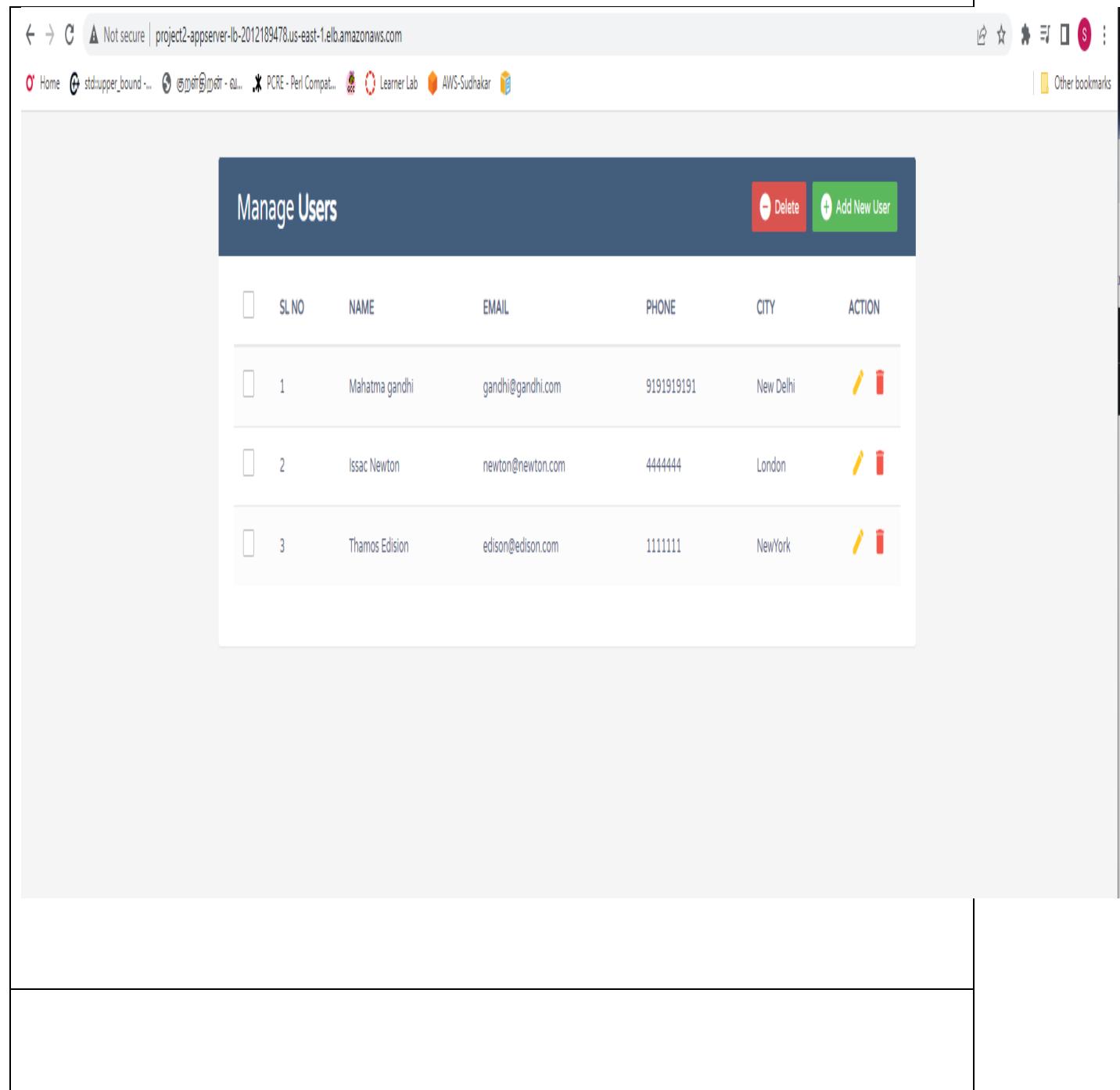
Details Tasks Events Auto Scaling Deployments Metrics Tags Logs

CPUUtilization MemoryUtilization

08:00 16:00 00:00 08:00 16:00 00:00

minimum maximum average

STEP 12: Open the Data Entry Application using the load balancer URL



Not secure | project2-appserver-lb-2012189478.us-east-1.elb.amazonaws.com

Home stdupper_bound ... குறள்கள் - வ... PCRE - Perl Compat... Learner Lab AWS-Sudhakar

Other bookmarks

Manage Users

SL NO	NAME	EMAIL	PHONE	CITY	ACTION
1	Mahatma gandhi	gandhi@gandhi.com	9191919191	New Delhi	 
2	Issac Newton	newton@newton.com	444444	London	 
3	Thamos Edison	edison@edison.com	1111111	New York	 

Delete the User ‘Thomas Edison’ and add a new user ‘sudhakar balagurusamy.’

← → C ⚠ Not secure | project2-appserver-lb-2012189478.us-east-1.elb.amazonaws.com



 Home  stdupper_bound ..  குரும்சிறை - வ...  PCRE - Perl Compat...  Learner Lab  AWS-Sudhakar 

Other bookmarks

Manage Users

 Delete  Add New User

	SL NO	NAME	EMAIL	PHONE	CITY	ACTION
<input type="checkbox"/>	1	Mahatma gandhi	gandhi@gandhi.com	9191919191	New Delhi	 
<input type="checkbox"/>	2	Issac Newton	newton@newton.com	444444	London	 
<input type="checkbox"/>	3	sudhakar balagurusamy	whitesnow@aol.com	7814921488	South Weymouth	 

Cleanup of Resources

Terminate the DB Server EC2 instance proj2-dbserver

The screenshot shows the AWS EC2 Instances page. A green banner at the top indicates "Successfully terminated i-0aac59c6fd81884bb". The main table lists one instance:

Name	Instance ID	Instance state	Instance type	Status check	Ala
proj2-dbserver	i-0aac59c6fd81884bb	Terminated	t2.micro	-	No

Delete the ECS cluster: project2-cluster.

The screenshot shows the AWS ECS Clusters page. A modal dialog titled "Delete Cluster" is open, asking if you want to delete the cluster "project2-cluster" and all its resources. It includes a confirmation message and a text input field with the placeholder "delete me".

Cluster : project2-cluster

Are you sure you want to delete the cluster project2-cluster and all the ECS resources within it?

Enter the phrase "delete me" into the field below to confirm deletion.

delete me

Cancel Delete

Delete the load balancer.

The screenshot shows the AWS EC2 Load Balancers page. On the left, there's a navigation sidebar with options like Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Scheduled Instances, Capacity Reservations, Images, AMIs, AMI Catalog, Elastic Block Store, Volumes, Snapshots, Lifecycle Manager, Network & Security, Security Groups, Elastic IPs, Placement Groups, Key Pairs, Network Interfaces, and Load Balancing. Under Load Balancing, the 'Load balancers' section is selected, showing one item: 'project2-appserver-lb'. A modal window titled 'Delete load balancer' is open over the list, asking for confirmation to delete the load balancer permanently. It includes a warning message about the action being irreversible and a note that target groups will become available for association to another load balancer. Below the modal, the main table shows the load balancer details: Application type, DNS name (project2-appserver-lb-2012189478), Status (Active), and VPC (vpc-0776b69936ebefaa6).

Delete the target group

The screenshot shows the AWS EC2 Target groups page. The left sidebar has the same navigation as the previous screenshot, with the 'Target Groups' section under Load Balancing selected. The main area shows two target groups: 'ecs-project-project2-app-service' and 'project2-appserver-tg'. A modal window titled 'Delete target group?' is open, warning that the action is irreversible and that deleting the target group deletes the group and its registered resources. It also asks if the user is sure they want to delete the target group. The target group 'project2-appserver-tg' is listed as a target. The main table below shows the target group details: Target type (Instance), Protocol : Port (HTTP: 80), Protocol version (HTTP1), and VPC (vpc-0776b69936ebefaa6).

Delete the security group

The screenshot shows the AWS EC2 Security Groups page. A modal dialog box is open, asking for confirmation to delete a security group named "ssh-db-port". The dialog lists the security group's ID and name. At the bottom are "Cancel" and "Delete" buttons.

Successfully deleted 6 security groups

Security Groups (1/5) Info

Name	Security group ID	Security group name	VPC ID	Description	Owner
sg-05e0e63661d0e400	http-port	vpc-0776b69936ebefaa6b	allows http traffic	506160996768	
sg-00e444a05094bc594	ssh-db-port	vpc-0776b69936ebefaa6b	ssh & mysql port	506160996768	
aws-c...	allow-ssh-mysql	vpc-0776b69936ebefaa6b	allow ssh & mysql	506160996768	
aws-c...	allow-ssh-mysql	vpc-0776b69936ebefaa6b	allow ssh & mysql	506160996768	
aws-c...	allow-ssh-mysql	vpc-0776b69936ebefaa6b	allow ssh & mysql	506160996768	

Delete security groups

Are you sure that you want to delete this security group?

- sg-00e444a05094bc594 - ssh-db-port

Cancel Delete

Delete the DB-server pipeline

The screenshot shows the AWS CloudWatch Pipelines page. A modal dialog box is open, asking for confirmation to delete a pipeline named "project2-db-pipeline". It includes a text input field for confirmation and a message about deleting change detection resources. Below the dialog is a table showing pipeline details and a checkbox for resource updates.

Pipelines Info

Delete project2-db-pipeline?

To confirm deletion, type *delete* in the field.

delete

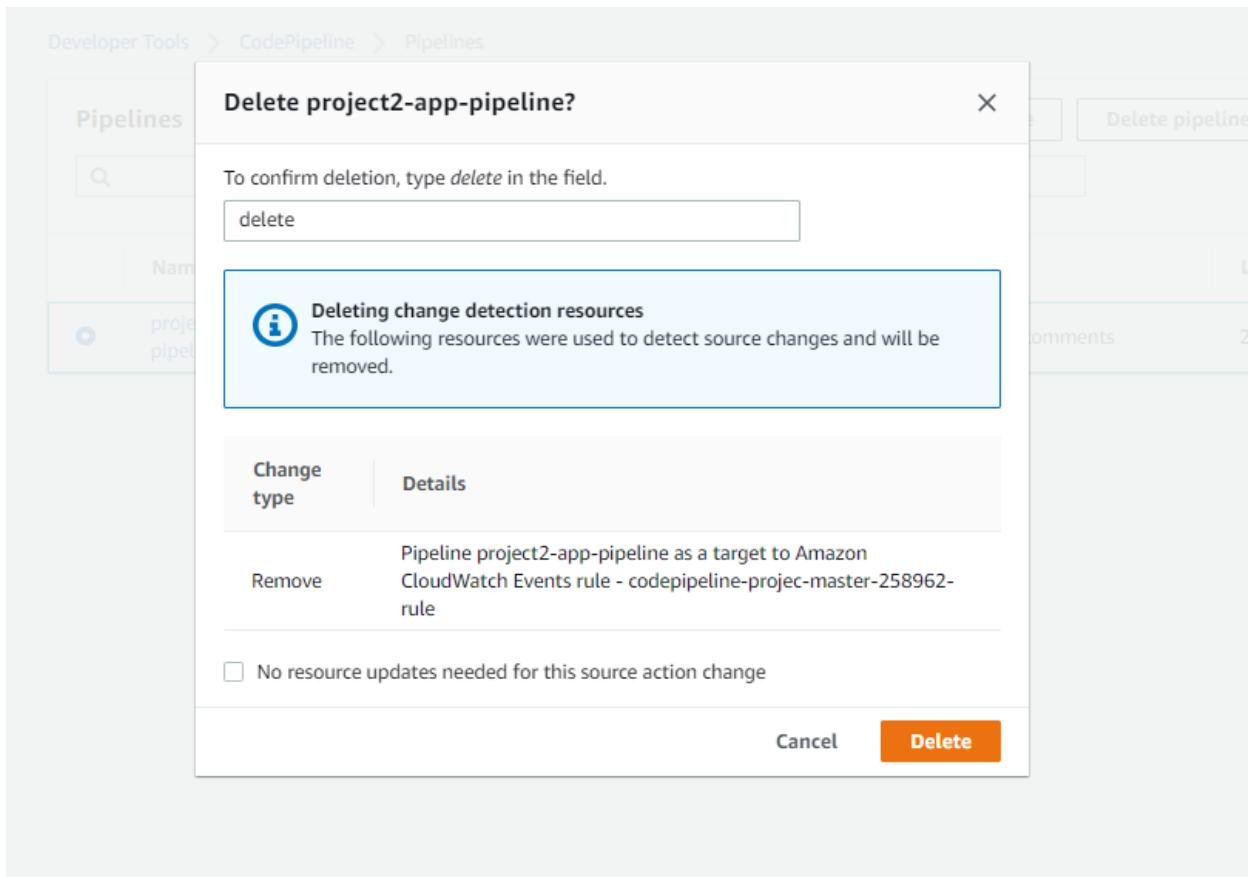
Deleting change detection resources
The following resources were used to detect source changes and will be removed.

Change type	Details
Remove	Pipeline project2-db-pipeline as a target to Amazon CloudWatch Events rule - codepipeline-project-master-191658-rule

No resource updates needed for this source action change

Cancel Delete

Delete the app-server pipeline



Delete the ECR repositories project2-appserver, dbserver

