React App Deployment with Docker, CI/CD, and CloudWatch Documentation

1. Project Overview

This project demonstrates a **full CI/CD workflow** for deploying a React application to **staging** and **production** environments on AWS EC2 using Docker. It also includes **log monitoring** via Amazon CloudWatch.

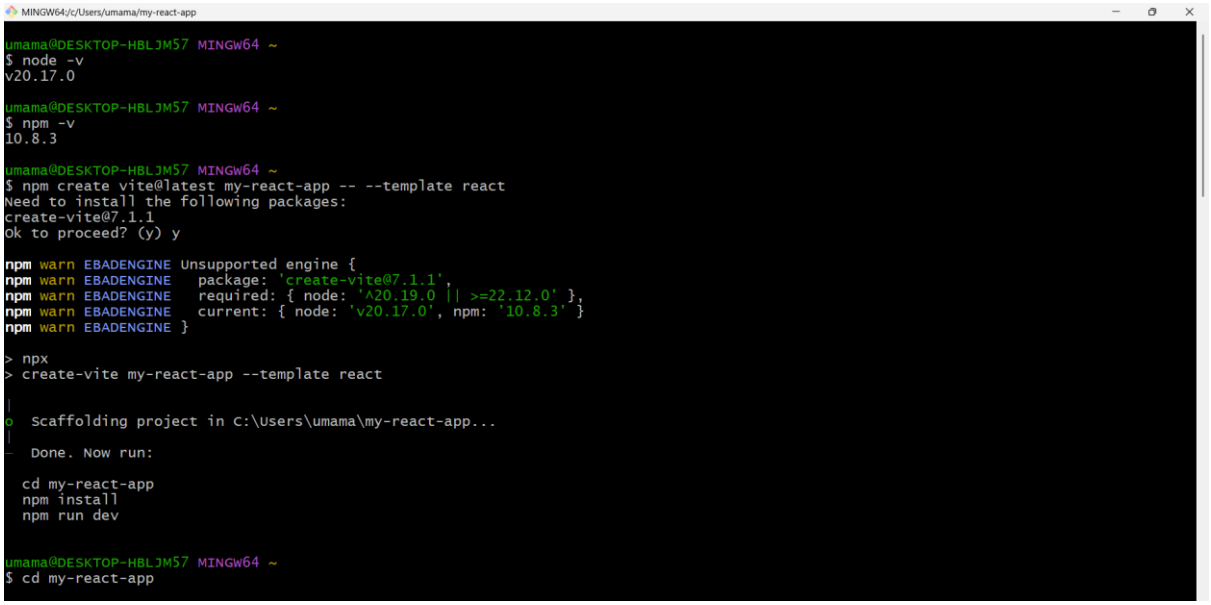
Key Features:

- **Separate Staging & Production environments** on EC2.
- **CI/CD Pipelines** implemented using GitHub Actions.
- **Docker** used for containerized deployments.
- **Nginx** serves the React application.
- **CloudWatch Agent** collects logs from containers into AWS CloudWatch Logs.

2. Prerequisites

- AWS account with **EC2 access**.
- IAM role with **CloudWatch permissions** attached to EC2.
- GitHub repository with your React project.
- **GitHub secrets**:
  - STAGING_IP → Staging EC2 public IP
  - STAGING_KEY → Staging EC2 private key
  - EC2_IP → Production EC2 public IP
  - EC2_SSH_KEY → Production EC2 private key
- Docker installed on EC2.

3. First created locally the react app



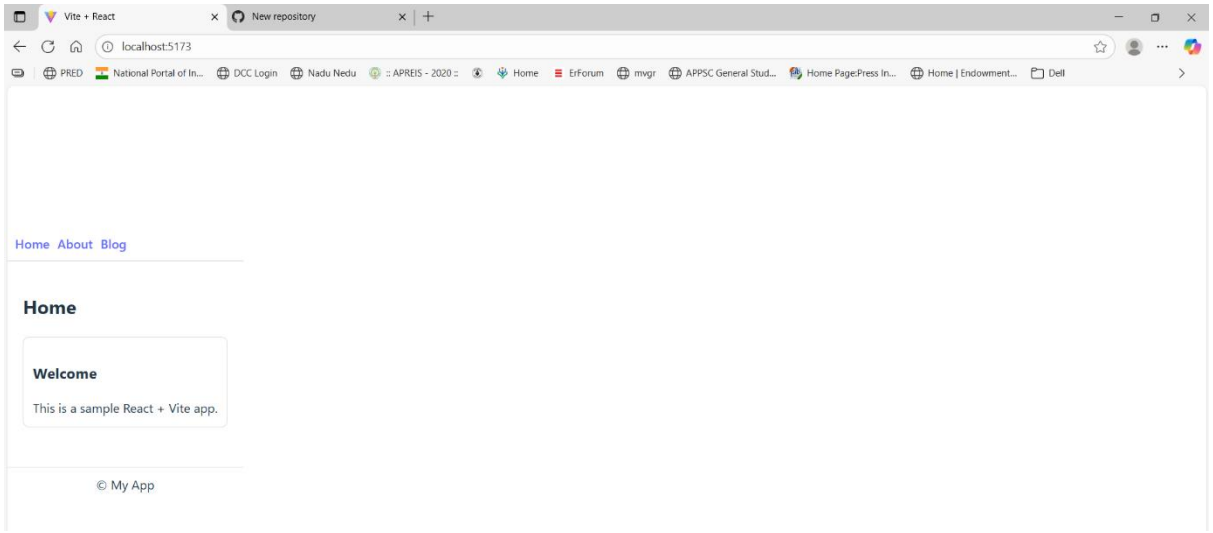
```
MINGW64:/c/Users/umama/my-react-app

umama@DESKTOP-HBLJM57 MINGW64 ~/my-react-app
$ npm install
npm warn EBADENGINE Unsupported engine {
npm warn EBADENGINE   package: '@vitejs/plugin-react@5.0.2',
npm warn EBADENGINE   required: { node: '^20.19.0 || >=22.12.0' },
npm warn EBADENGINE   current: { node: 'v20.17.0', npm: '10.8.3' }
npm warn EBADENGINE }
npm warn EBADENGINE Unsupported engine {
npm warn EBADENGINE   package: 'vite@7.1.3',
npm warn EBADENGINE   required: { node: '^20.19.0 || >=22.12.0' },
npm warn EBADENGINE   current: { node: 'v20.17.0', npm: '10.8.3' }
npm warn EBADENGINE }

added 152 packages, and audited 153 packages in 41s

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

found 0 vulnerabilities

umama@DESKTOP-HBLJM57 MINGW64 ~/my-react-app
$ npm i react-router-dom
npm warn EBADENGINE Unsupported engine {
npm warn EBADENGINE   package: '@vitejs/plugin-react@5.0.2',
npm warn EBADENGINE   required: { node: '^20.19.0 || >=22.12.0' },
npm warn EBADENGINE   current: { node: 'v20.17.0', npm: '10.8.3' }
npm warn EBADENGINE }
npm warn EBADENGINE Unsupported engine {
npm warn EBADENGINE   package: 'vite@7.1.3',
npm warn EBADENGINE   required: { node: '^20.19.0 || >=22.12.0' },
npm warn EBADENGINE   current: { node: 'v20.17.0', npm: '10.8.3' }
npm warn EBADENGINE }

added 4 packages, and audited 157 packages in 5s

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

Adjusting the package.json file according to our needs

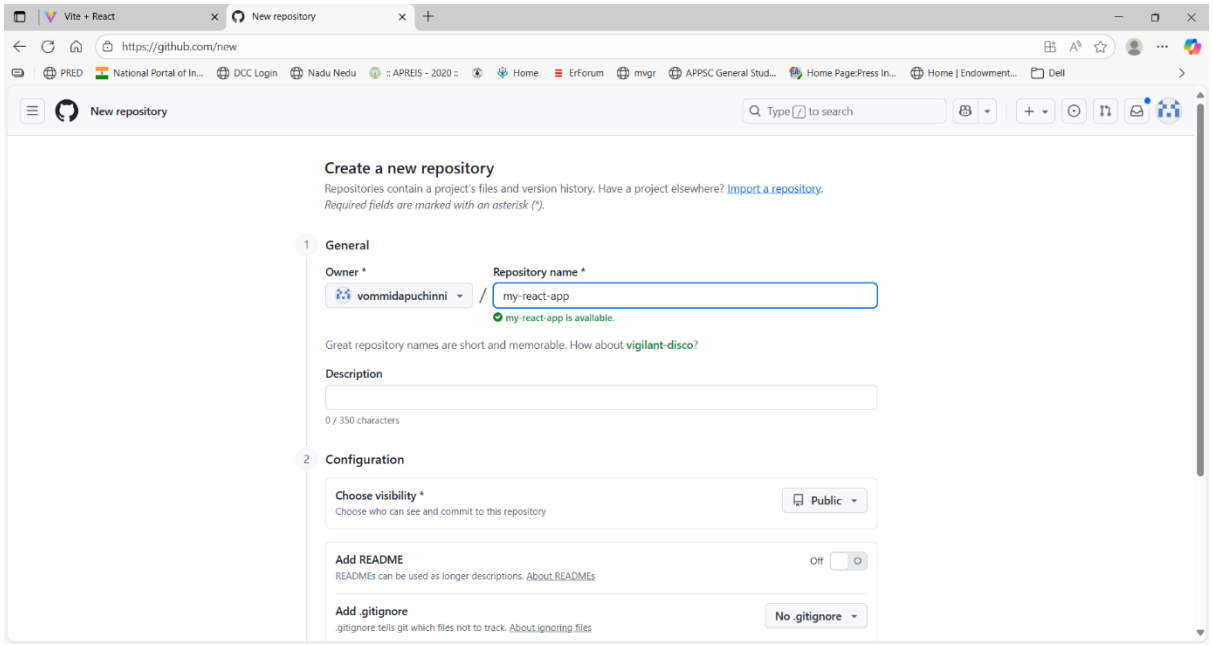
```
MINGW64:/c/Users/umama/my-react-app

umama@DESKTOP-HBLJM57 MINGW64 ~/my-react-app
$ npm install -D eslint vitest @testing-library/react @testing-library/jest-dom
npm warn EBADENGINE Unsupported engine {
npm warn EBADENGINE   package: '@vitejs/plugin-react@5.0.2',
npm warn EBADENGINE   required: { node: '^20.19.0 || >=22.12.0' },
npm warn EBADENGINE   current: { node: 'v20.17.0', npm: '10.8.3' }
npm warn EBADENGINE }
npm warn EBADENGINE Unsupported engine {
npm warn EBADENGINE   package: 'vite@7.1.3',
npm warn EBADENGINE   required: { node: '^20.19.0 || >=22.12.0' },
npm warn EBADENGINE   current: { node: 'v20.17.0', npm: '10.8.3' }
npm warn EBADENGINE }

added 54 packages, and audited 211 packages in 22s

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

found 0 vulnerabilities

umama@DESKTOP-HBLJM57 MINGW64 ~/my-react-app
$ ls
README.md  eslint.config.js  index.html  node_modules/  package-lock.json  package.json  public/  src/  vite.config.js

umama@DESKTOP-HBLJM57 MINGW64 ~/my-react-app
$ vi package.json

umama@DESKTOP-HBLJM57 MINGW64 ~/my-react-app
$ cat package.json
{
  "name": "my-react-app",
  "private": true,
  "version": "0.0.0",
  "type": "module",
  "scripts": {
    "dev": "vite",
    "build": "vite build",
    "lint": "eslint .",
```

```
umama@DESKTOP-HBLJM57 MINGW64 ~/my-react-app
$ npm run build

> my-react-app@0.0.0 build
> vite build

You are using Node.js 20.17.0. Vite requires Node.js version 20.19+ or 22.12+. Please upgrade your Node.js version.
vite v7.1.3 building for production...
✓ 47 modules transformed.
dist/index.html                   0.46 kB │ gzip:  0.30 kB
dist/assets/index-Dtn62Xmo.css    0.91 kB │ gzip:  0.49 kB
dist/assets/index-BMOMM_uG.js   220.73 kB │ gzip: 70.96 kB
✓ built in 2.92s
```

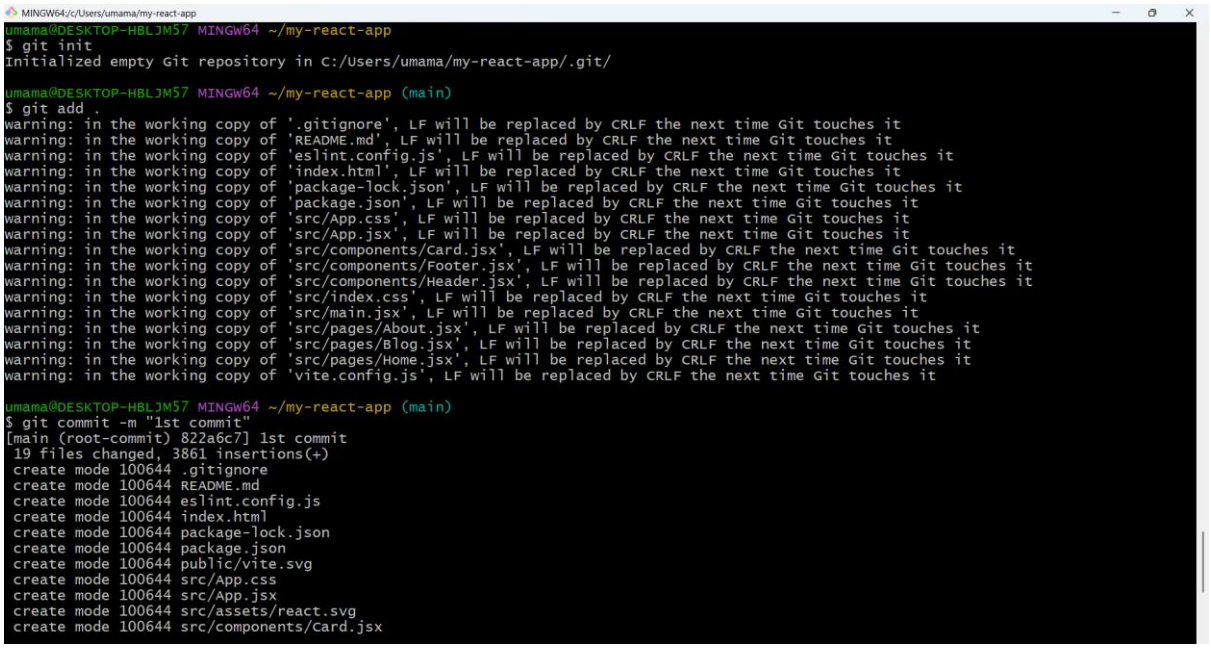
Accessing the file locally



Create git repo



Push all code to the git repo




```
umama@DESKTOP-HBLJM57 MINGW64 ~/my-react-app (main)
$ git remote add origin https://github.com/vommidapuchinni/my-react-app.git

umama@DESKTOP-HBLJM57 MINGW64 ~/my-react-app (main)
$ git push origin main
Enumerating objects: 26, done.
Counting objects: 100% (26/26), done.
Delta compression using up to 8 threads
Compressing objects: 100% (24/24), done.
Writing objects: 100% (26/26), 35.93 KiB | 1.80 MiB/s, done.
Total 26 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/vommidapuchinni/my-react-app.git
 * [new branch]      main -> main

umama@DESKTOP-HBLJM57 MINGW64 ~/my-react-app (main)
$
```

4. Docker Setup

4.1 Dockerfile

4.2 docker-compose-staging.yaml

4.2 docker-compose-staging.yaml

5. CloudWatch Configuration

5.1 cloudwatch-staging.json

5.2 cloudwatch-prod.json

6. GitHub Actions CI/CD Pipelines

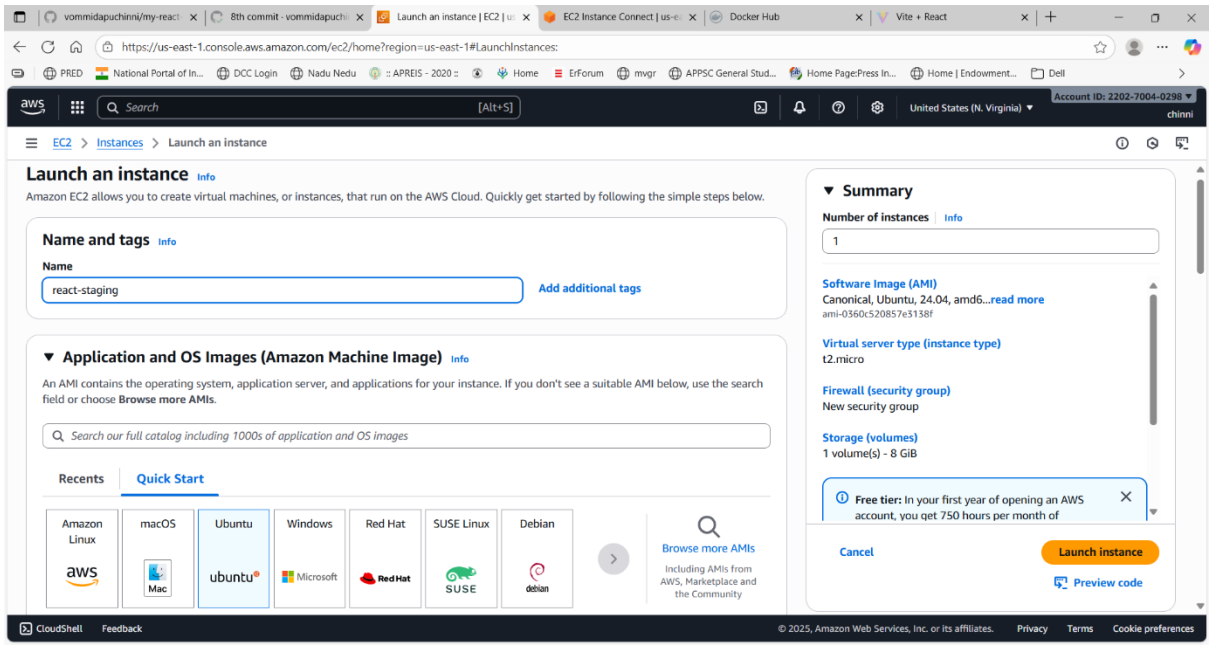
6.1 deploy-staging.yaml

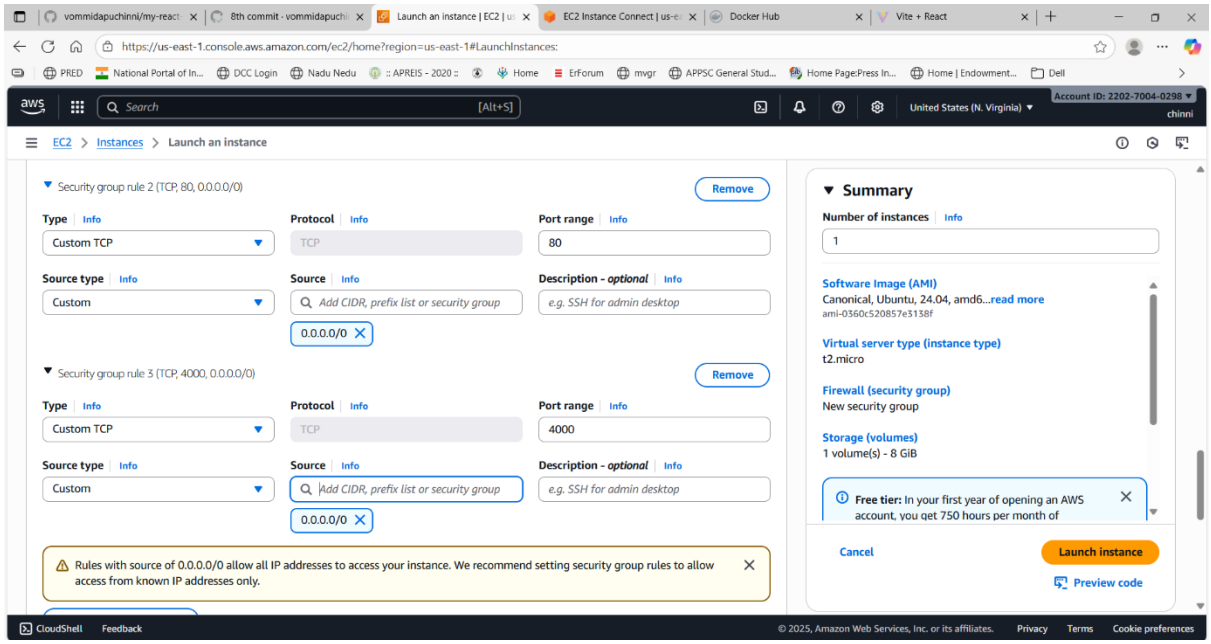
6.2 deploy-prod.yaml

7. Deployment Steps

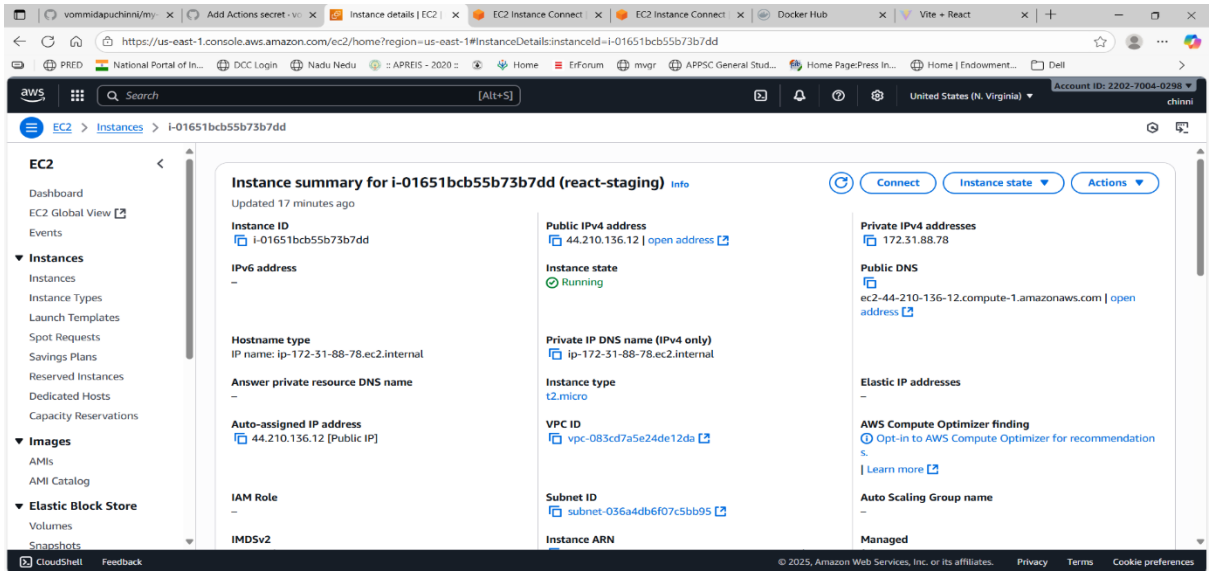
7.1 Staging

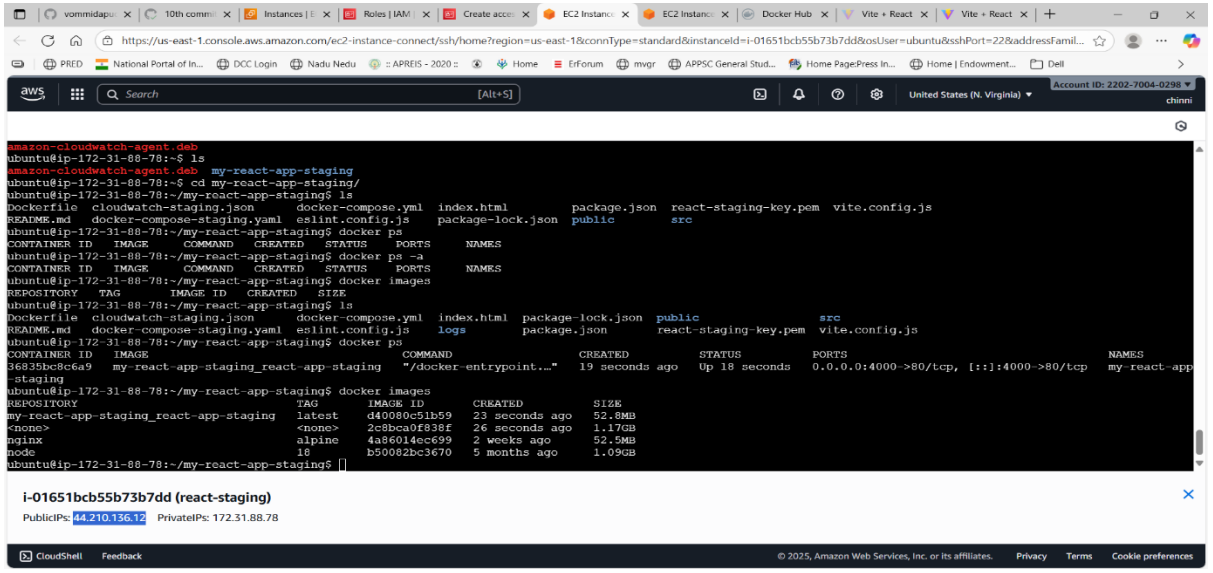
1. Push code to staging branch → triggers GitHub Actions workflow.
2. Workflow SSHs into staging EC2.

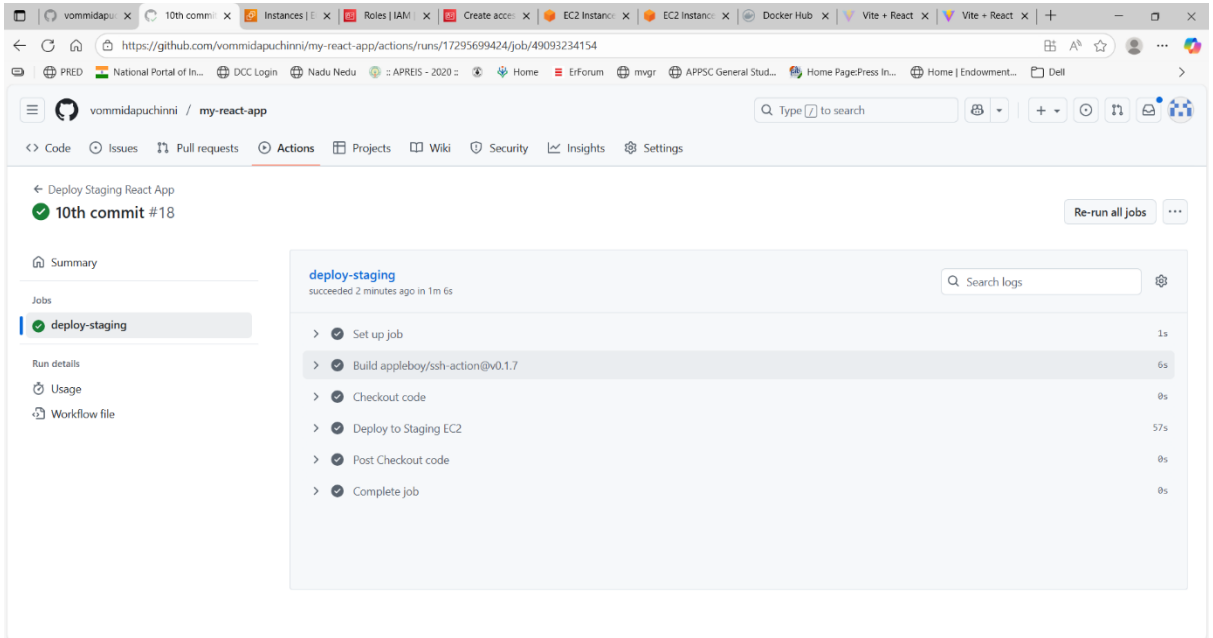


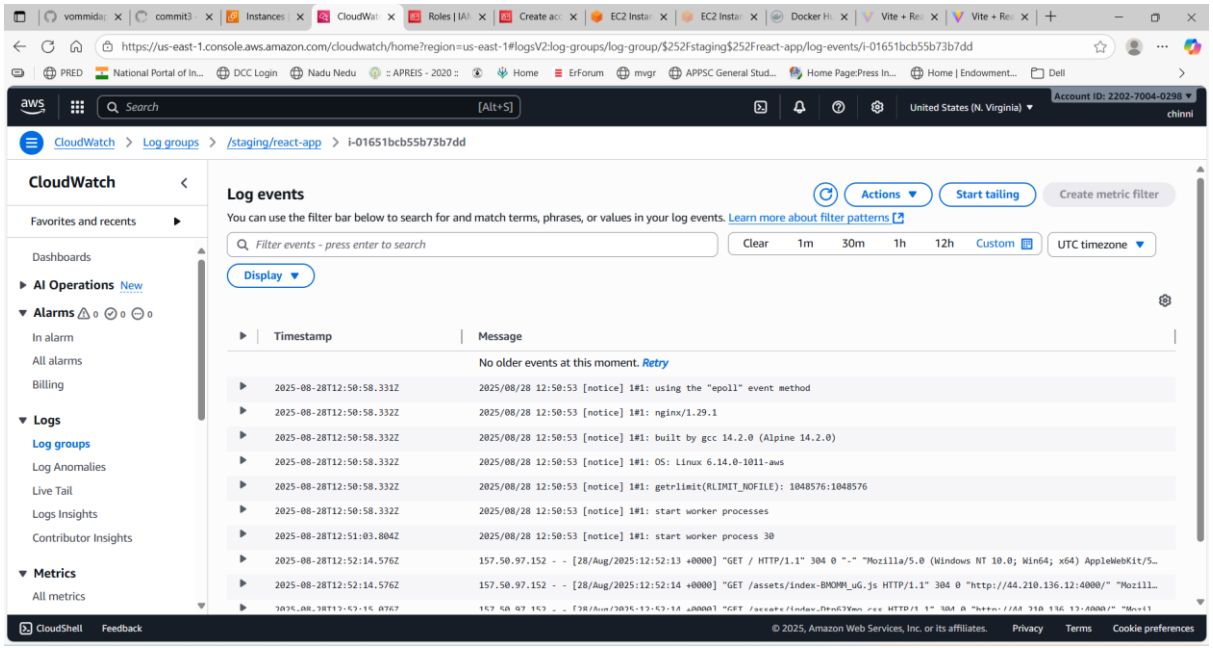


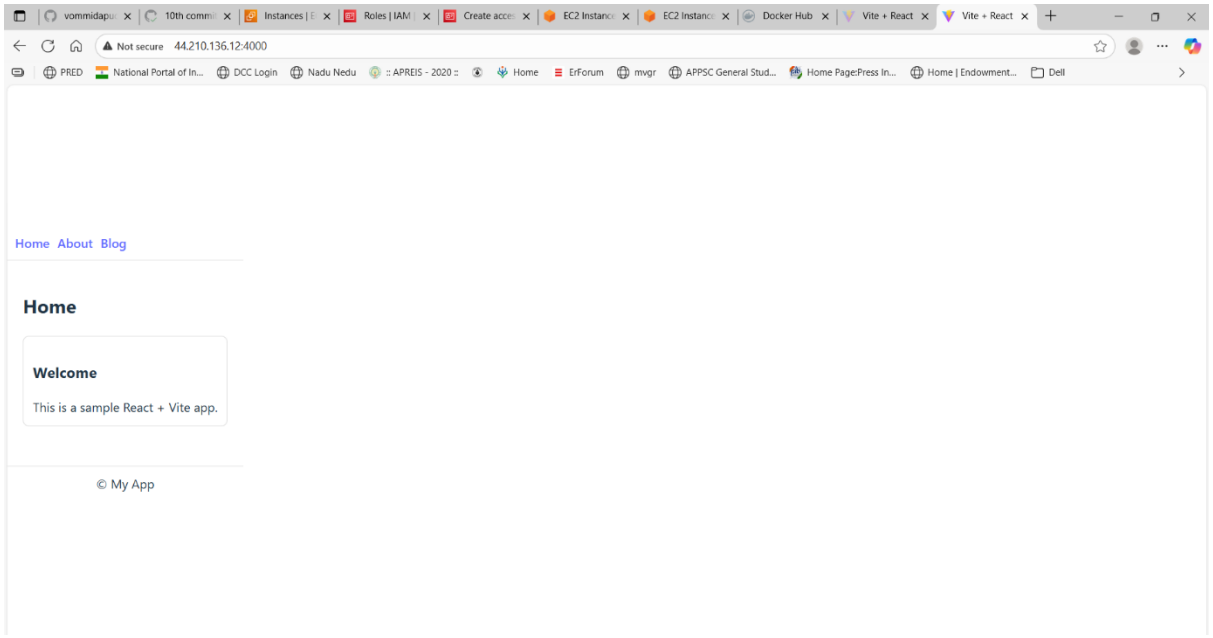
3. Pulls latest code, builds Docker image, runs container.
4. CloudWatch agent collects logs to /staging/react-app.


Instance summary for i-01651bcb55b73b7dd (react-staging) Info
Updated 17 minutes ago
Connect
Instance state
Actions
Instance ID
i-01651bcb55b73b7dd
Public IPv4 address
44.210.136.12 | open address
Private IPv4 addresses
172.31.88.78
IPv6 address
–
Instance state
Running
Public DNS
ec2-44-210-136-12.compute-1.amazonaws.com | open address
Hostname type
IP name: ip-172-31-88-78.ec2.internal
Private IP DNS name (IPv4 only)
ip-172-31-88-78.ec2.internal
Answer private resource DNS name
–
Instance type
t2.micro
Elastic IP addresses
–
Auto-assigned IP address
44.210.136.12 [Public IP]
VPC ID
vpc-083cd7a5e24de12da
AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations.
Learn more
IAM Role
–
Subnet ID
subnet-036a4db6f07c5bb95
Auto Scaling Group name
–
IMDSv2
Instance ARN
Managed


amazon-cloudwatch-agent.deb
ubuntu@ip-172-31-88-78:~$ ls
amazon-cloudwatch-agent.deb my-react-app-staging
ubuntu@ip-172-31-88-78:~$ cd my-react-app-staging/
ubuntu@ip-172-31-88-78:~/my-react-app-staging$ ls
Dockerfile cloudwatch-staging.json docker-compose.yml index.html package.json react-staging-key.pem vite.config.js
README.md docker-compose-staging.yaml eslint.config.js package-lock.json public src
ubuntu@ip-172-31-88-78:~/my-react-app-staging$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
ubuntu@ip-172-31-88-78:~/my-react-app-staging$ docker ps -a
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
ubuntu@ip-172-31-88-78:~/my-react-app-staging$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
ubuntu@ip-172-31-88-78:~/my-react-app-staging$ ls
Dockerfile cloudwatch-staging.json docker-compose.yml index.html package-lock.json public src
README.md docker-compose-staging.yaml eslint.config.js logs package.json react-staging-key.pem vite.config.js
ubuntu@ip-172-31-88-78:~/my-react-app-staging$ docker ps
CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES
36835bc8c6a9 my-react-app-staging_react-app-staging "/docker-entrypoint.…" 19 seconds ago Up 18 seconds 0.0.0.0:4000->80/tcp, [::]:4000->80/tcp my-react-app-staging
ubuntu@ip-172-31-88-78:~/my-react-app-staging$ docker images
REPOSITORY TAG IMAGE ID CREATED SIZE
my-react-app-staging_react-app-staging latest d40080c51b59 23 seconds ago 52.8MB
<none> <none> 2c8bca0f838f 26 seconds ago 1.17GB
nginx alpine 4a86014ec699 2 weeks ago 52.5MB
node 18 b50082bc3670 5 months ago 1.09GB
ubuntu@ip-172-31-88-78:~/my-react-app-staging$
i-01651bcb55b73b7dd (react-staging)
PublicIPs: 44.210.136.12 PrivateIPs: 172.31.88.78


vommidapuchinni / my-react-app
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Deploy Staging React App
10th commit #18
Re-run all jobs
Summary
Jobs
deploy-staging
Run details
Usage
Workflow file
deploy-staging
succeeded 2 minutes ago in 1m 6s
Search logs
Set up job 1s
Build appleboy/ssh-action@v0.1.7 6s
Checkout code 0s
Deploy to Staging EC2 57s
Post Checkout code 0s
Complete job 0s

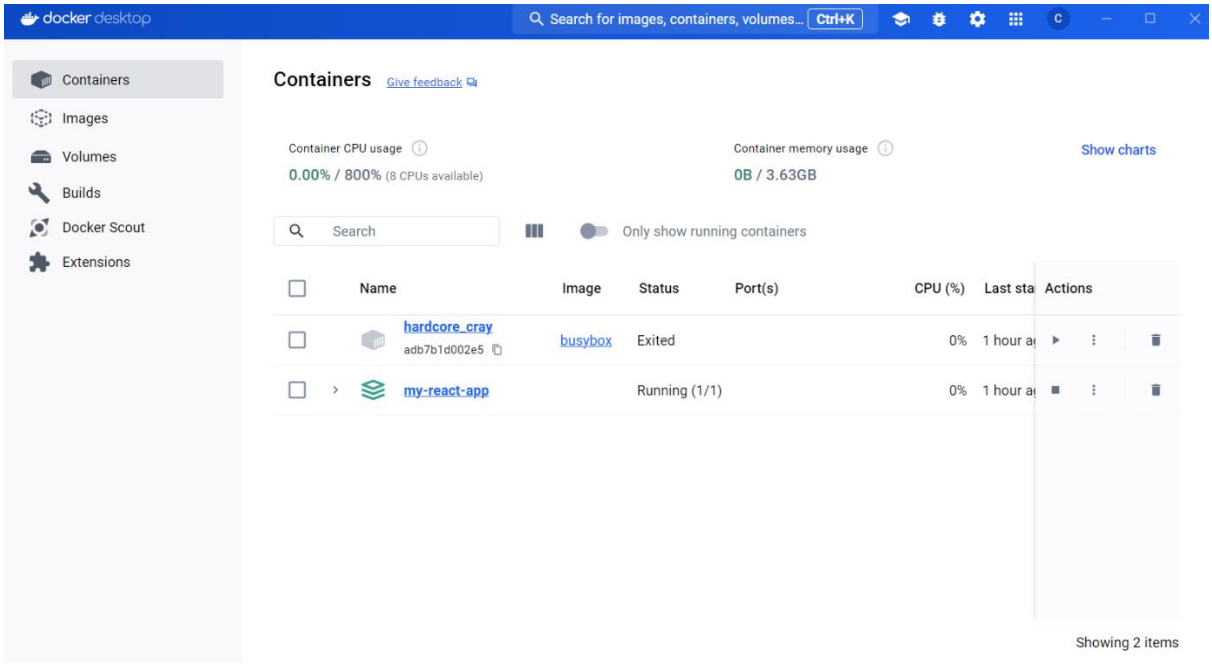


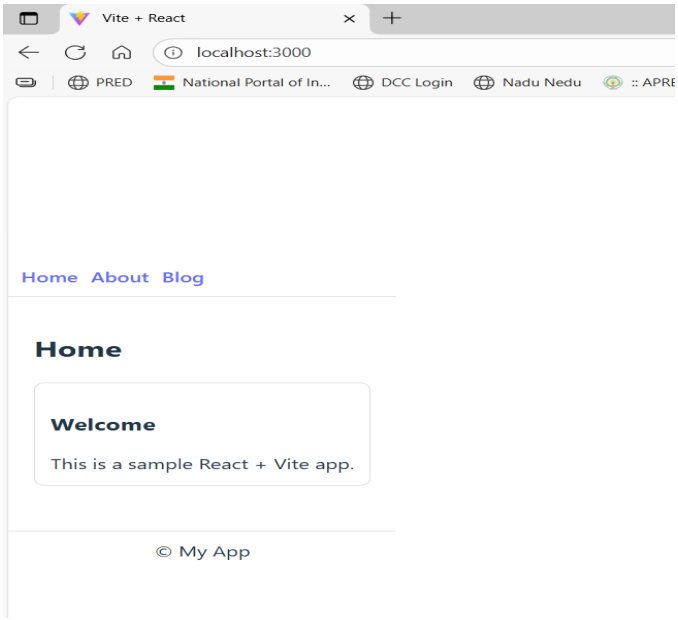


With docker files created locally with docker desktop

```
MINGW64:/c/Users/umama/my-react-app
umama@DESKTOP-HBLJM57 MINGW64 ~/my-react-app (main)
$ docker-compose up -d --build
[+] Building 0.0s (0/0)  docker:desktop-linux
[+] Building 57.0s (16/16) FINISHED                                                                    docker:desktop-linux
 => [react-app internal] load build definition from Dockerfile                                                     0.1s
 => => transferring dockerfile: 357B                                                                               0.0s
 => [react-app internal] load metadata for docker.io/library/nginx:alpine                                          6.2s
 => [react-app internal] load metadata for docker.io/library/node:18                                               6.2s
 => [react-app auth] library/nginx:pull token for registry-1.docker.io                                             0.0s
 => [react-app auth] library/node:pull token for registry-1.docker.io                                              0.0s
 => [react-app internal] load .dockerignore                                                                        0.1s
 => => transferring context: 2B                                                                                    0.0s
 => [react-app build 1/6] FROM docker.io/library/node:18@sha256:c6ae79e38498325db67193d391e6ec1d224d96c693a8a4d943498556716d3783  0.0s
 => CACHED [react-app stage-1 1/2] FROM docker.io/library/nginx:alpine@sha256:42a516af16b852e33b7682d5ef8acbd5d13fe08fecadc7ed98  0.0s
 => [react-app internal] load build context                                                                        1.8s
 => => transferring context: 388.48kB                                                                              1.8s
 => CACHED [react-app build 2/6] WORKDIR /app                                                                      0.0s
 => CACHED [react-app build 3/6] COPY package*.json ./                                                             0.0s
 => CACHED [react-app build 4/6] RUN npm install                                                                   0.0s
 => [react-app build 5/6] COPY . .                                                                                11.7s
 => [react-app build 6/6] RUN npm run build                                                                       16.2s
 => [react-app stage-1 2/2] COPY --from=build /app/dist /usr/share/nginx/html                                      0.7s
 => [react-app] exporting to image                                                                                 1.0s
 => => exporting layers                                                                                            0.3s
 => => writing image sha256:4bc09093c16cdc43790a79c4dcb5bf7e217fcb02c3bffd803c0018fe5f2c73ad                       0.0s
 => => naming to docker.io/library/my-react-app-react-app                                                          0.0s
[+] Running 2/2
 ✔ Network my-react-app_default        Created                                                                     4.5s
 ✔ Container my-react-app-react-app-1  Started                                                                    12.0s

umama@DESKTOP-HBLJM57 MINGW64 ~/my-react-app (main)
$ docker ps
CONTAINER ID   IMAGE                  COMMAND                  CREATED              STATUS          PORTS                    NAMES
32ad1e01aeab   my-react-app-react-app   "/docker-entrypoint.…"   About a minute ago   Up 57 seconds   0.0.0.0:3000->80/tcp   my-react-a
pp-react-app-1

umama@DESKTOP-HBLJM57 MINGW64 ~/my-react-app (main)
$ docker images
REPOSITORY                              TAG                                       IMAGE ID
```

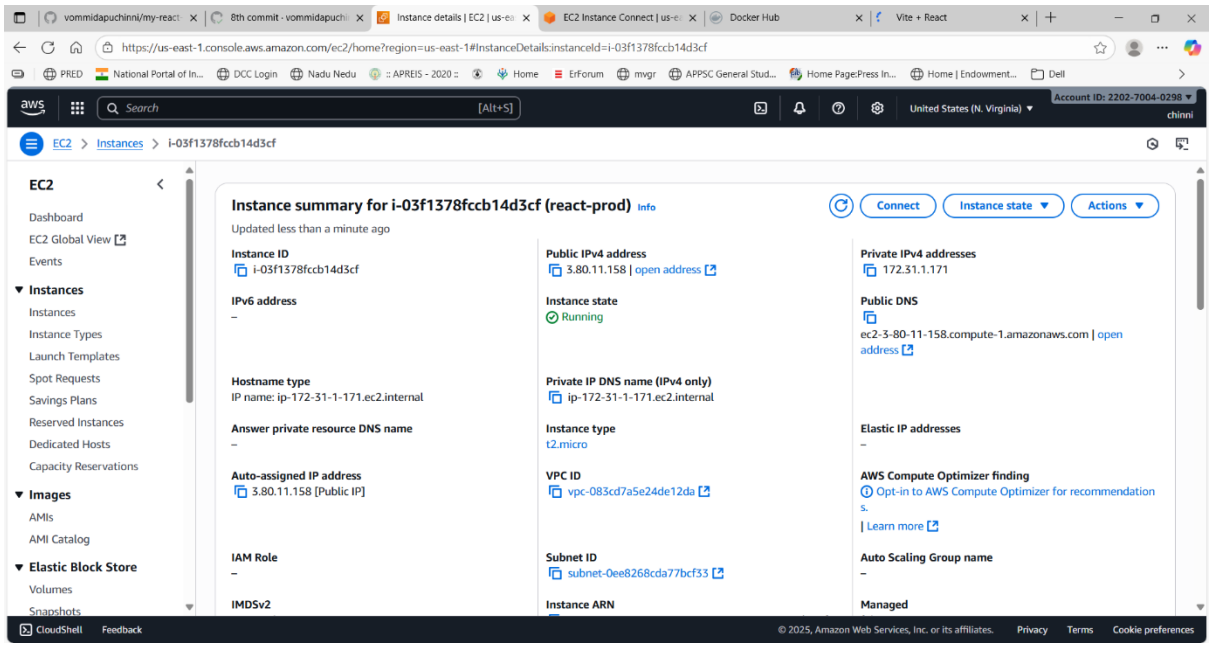
```
umama@DESKTOP-HBLJM57 MINGW64 ~/my-react-app (main)
$ docker version
Client:
 Version:           26.1.4
 API version:       1.45
 Go version:        go1.21.11
 Git commit:        5650f9b
 Built:             Wed Jun  5 11:29:54 2024
 OS/Arch:           windows/amd64
 Context:           desktop-linux

Server: Docker Desktop 4.31.1 (153621)
 Engine:
  Version:          26.1.4
  API version:      1.45 (minimum version 1.24)
  Go version:       go1.21.11
  Git commit:       de5c9cf
  Built:            Wed Jun  5 11:29:22 2024
  OS/Arch:          linux/amd64
  Experimental:     false
 containerd:
  Version:          1.6.33
  GitCommit:        d2d58213f83a351ca8f528a95fbd145f5654e957
 runc:
  Version:          1.1.12
  GitCommit:        v1.1.12-0-g51d5e94
 docker-init:
  Version:          0.19.0
  GitCommit:        de40ad0
```



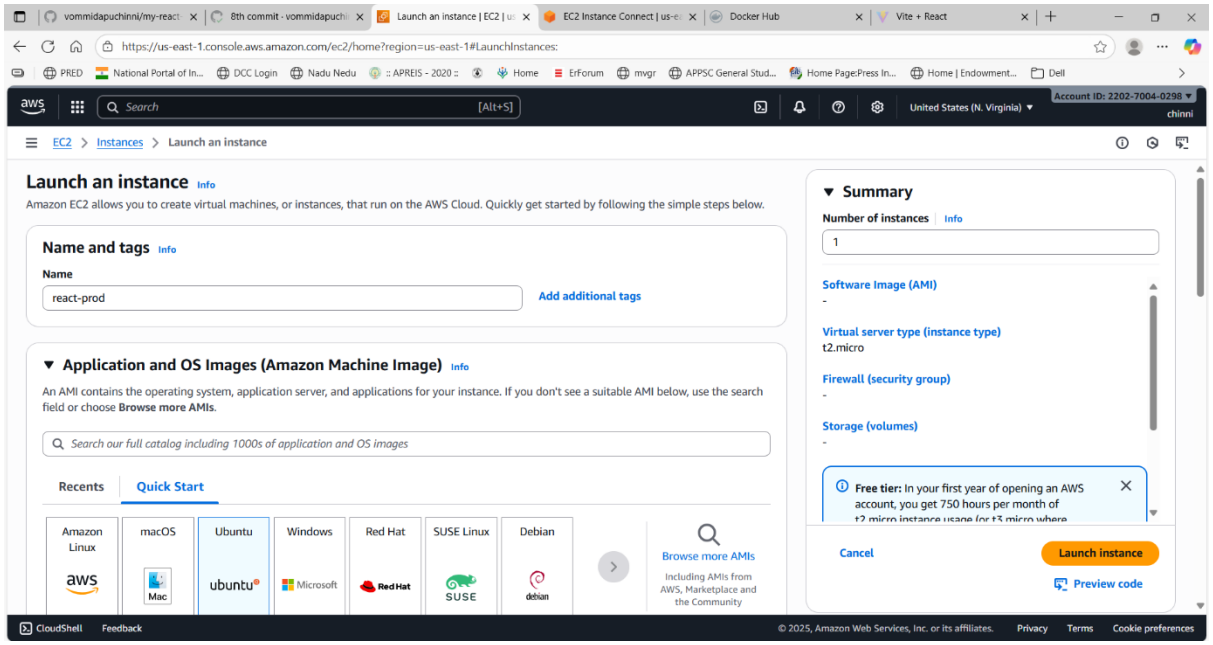


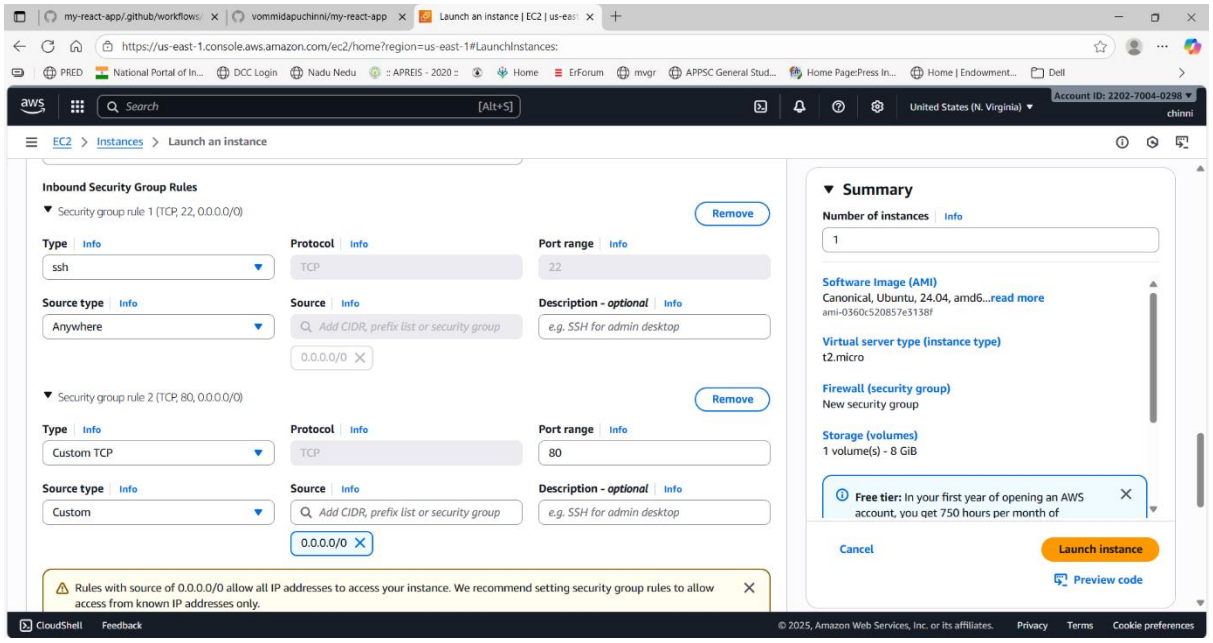
7.2 Production

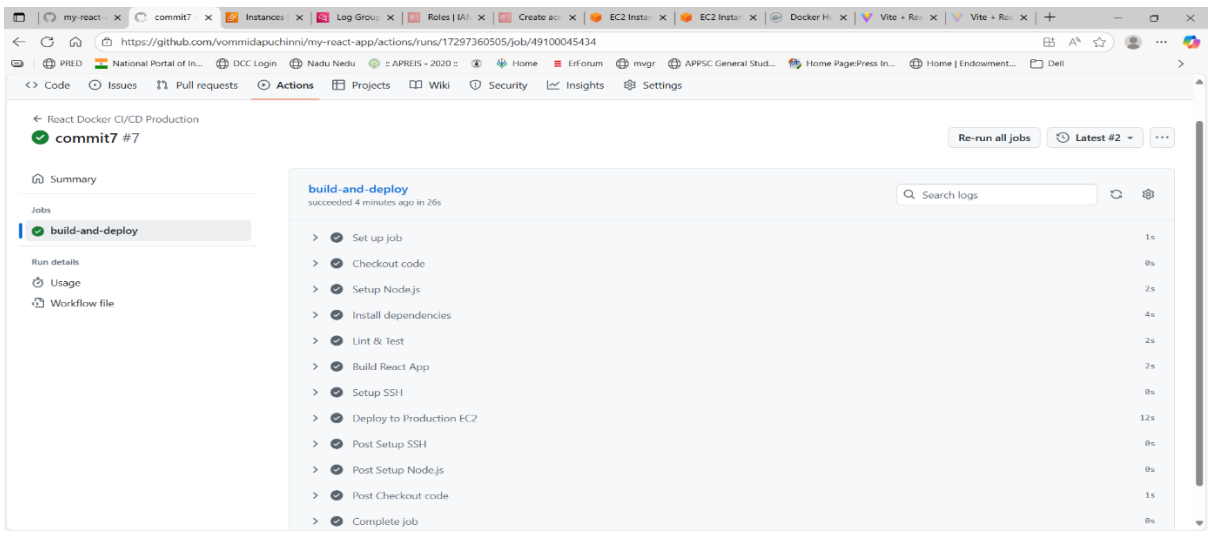
1. Merge PR to main branch → triggers GitHub Actions workflow.
2. Workflow SSHs into production EC2.

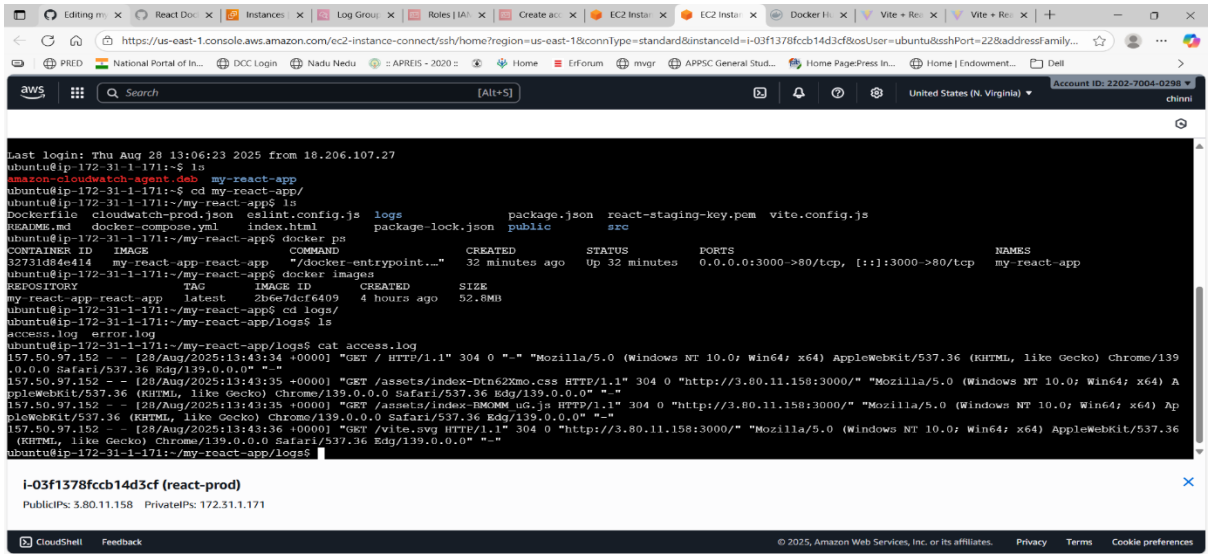


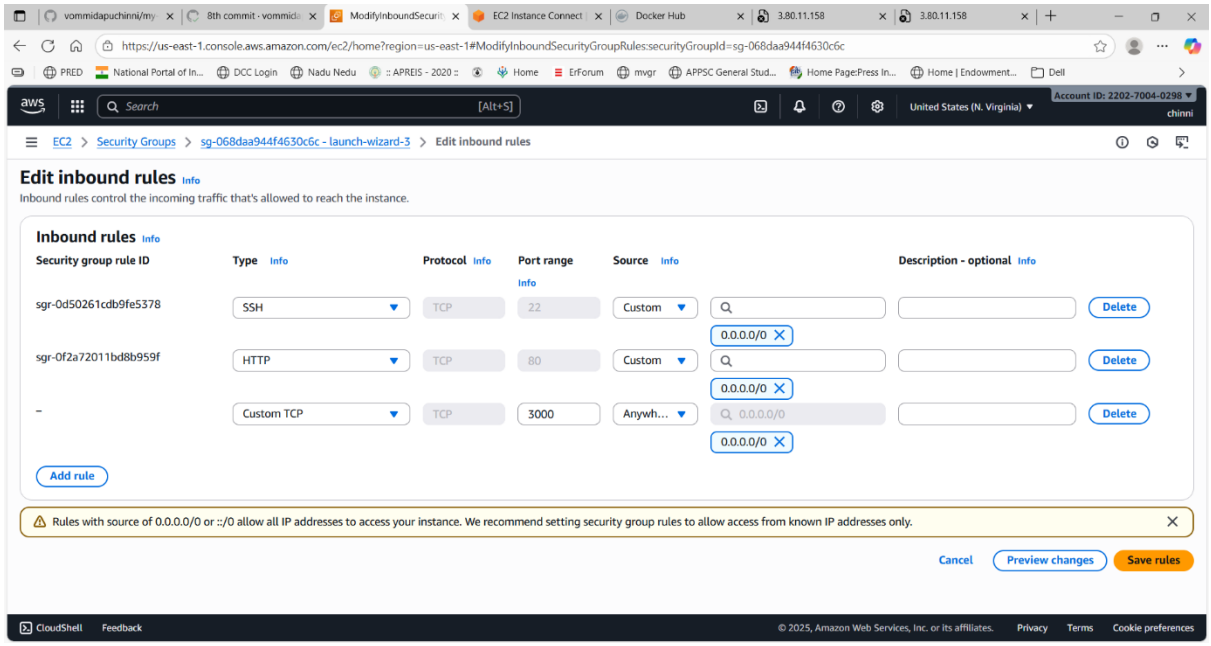
3. Pulls latest code, builds Docker image, runs container.
4. CloudWatch agent collects logs to /production/react-app.

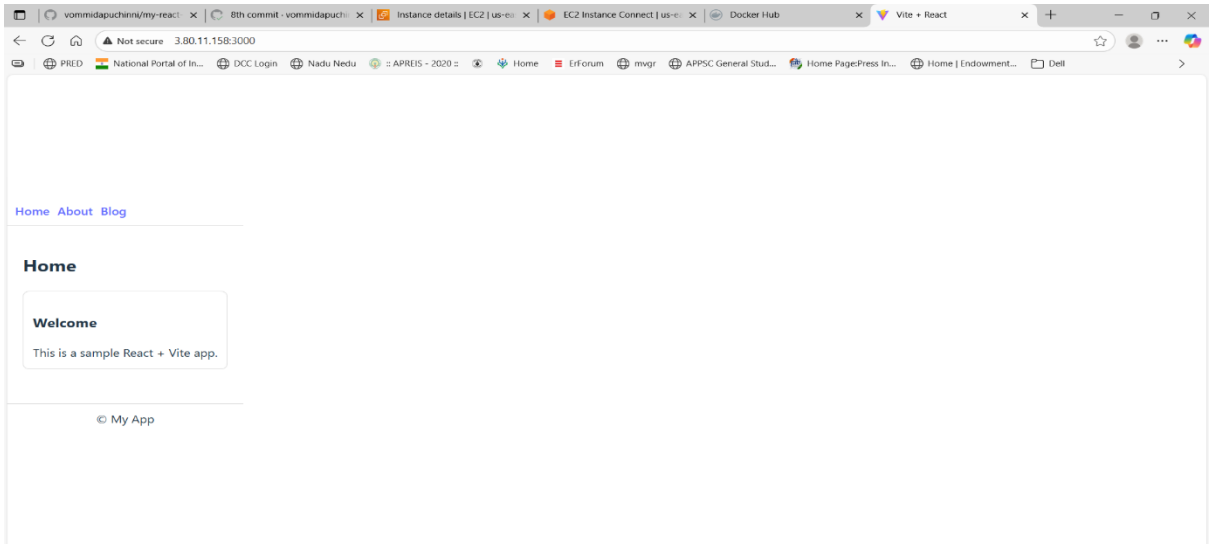

Launch an instance | EC2 | us-
https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#LaunchInstances:
Search
[Alt+S]
United States (N. Virginia)
Account ID: 2202-7004-0298
chinni
EC2 > Instances > Launch an instance
Launch an instance Info
Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.
Name and tags Info
Name
react-prod
Add additional tags
Application and OS Images (Amazon Machine Image) Info
An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose Browse more AMIs.
Search our full catalog including 1000s of application and OS images
Recents
Quick Start
Amazon Linux
macOS
Ubuntu
Windows
Red Hat
SUSE Linux
Debian
Browse more AMIs
Including AMIs from AWS, Marketplace and the Community
Summary
Number of instances Info
1
Software Image (AMI)
-
Virtual server type (instance type)
t2.micro
Firewall (security group)
-
Storage (volumes)
-
Free tier: In your first year of opening an AWS account, you get 750 hours per month of
Cancel
Launch instance
Preview code
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


Launch an instance | EC2 | us-east
https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#LaunchInstances:
Search
[Alt+S]
United States (N. Virginia)
Account ID: 2202-7004-0298
chinni
EC2 > Instances > Launch an instance
Inbound Security Group Rules
Security group rule 1 (TCP, 22, 0.0.0.0/0)
Remove
Type Info
ssh
Protocol Info
TCP
Port range Info
22
Source type Info
Anywhere
Source Info
Add CIDR, prefix list or security group
0.0.0.0/0
Description - optional Info
e.g. SSH for admin desktop
Security group rule 2 (TCP, 80, 0.0.0.0/0)
Remove
Type Info
Custom TCP
Protocol Info
TCP
Port range Info
80
Source type Info
Custom
Source Info
Add CIDR, prefix list or security group
0.0.0.0/0
Description - optional Info
e.g. SSH for admin desktop
Rules with source of 0.0.0.0/0 allow all IP addresses to access your instance. We recommend setting security group rules to allow access from known IP addresses only.
Summary
Number of instances Info
1
Software Image (AMI)
Canonical, Ubuntu, 24.04, amd6...read more
ami-0360c520857e3138f
Virtual server type (instance type)
t2.micro
Firewall (security group)
New security group
Storage (volumes)
1 volume(s) - 8 GiB
Free tier: In your first year of opening an AWS account, you get 750 hours per month of
Cancel
Launch instance
Preview code
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

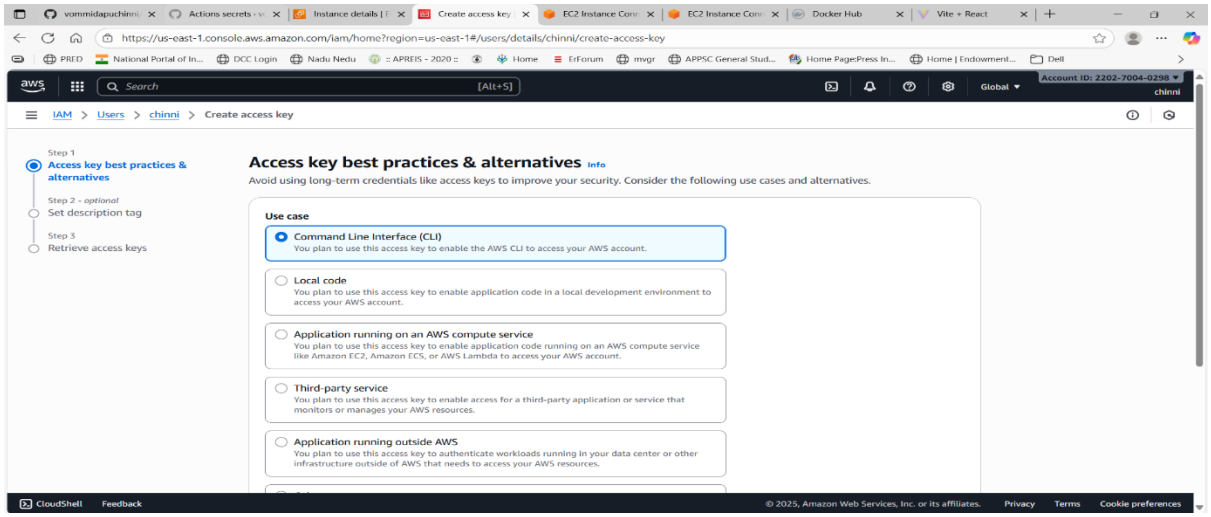

https://github.com/vommidapuchinni/my-react-app/actions/runs/17297360505/job/49100045434
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
React Docker CI/CD Production
commit7 #7
Re-run all jobs
Latest #2
Summary
Jobs
build-and-deploy
Run details
Usage
Workflow file
build-and-deploy
succeeded 4 minutes ago in 26s
Search logs
Set up job
Checkout code
Setup Node.js
Install dependencies
Lint & Test
Build React App
Setup SSH
Deploy to Production EC2
Post Setup SSH
Post Setup Node.js
Post Checkout code
Complete job

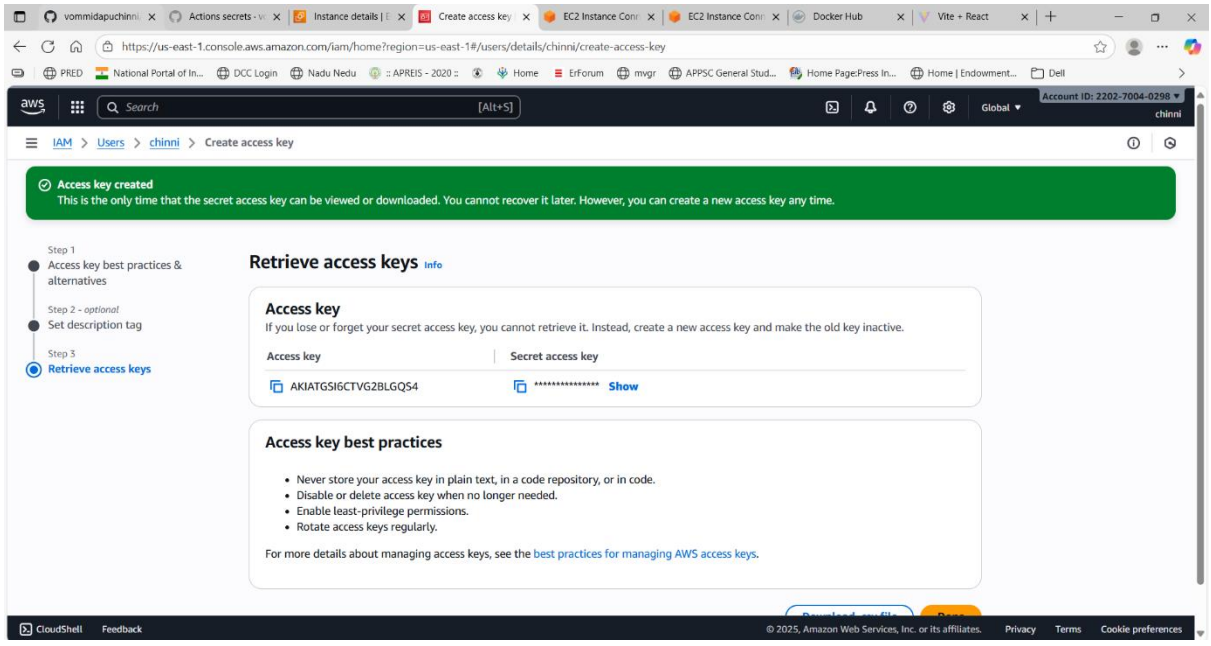


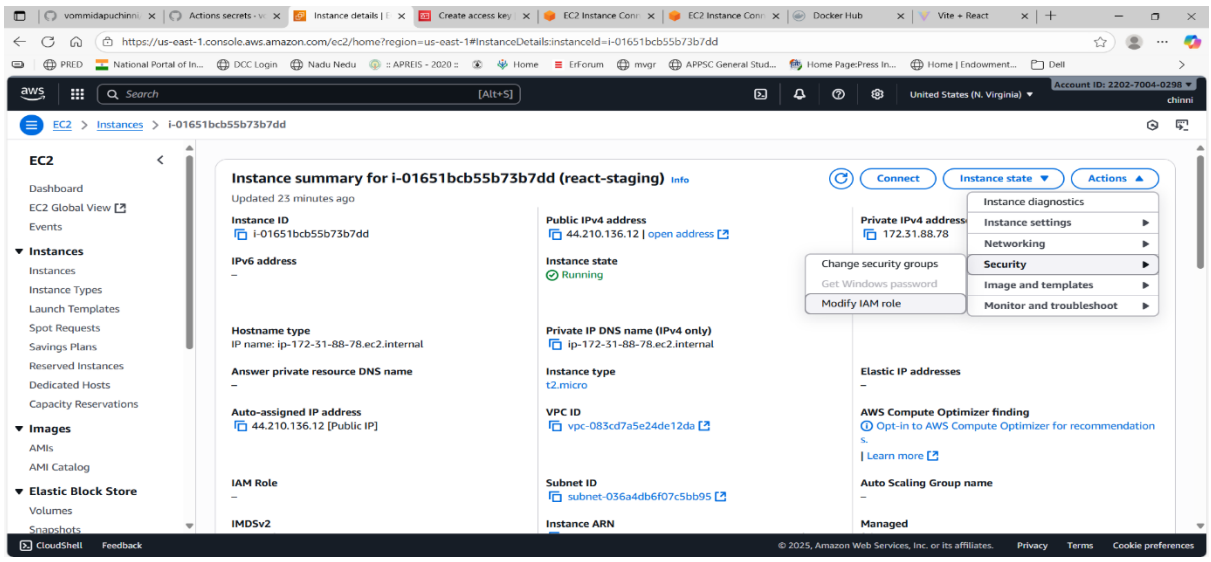


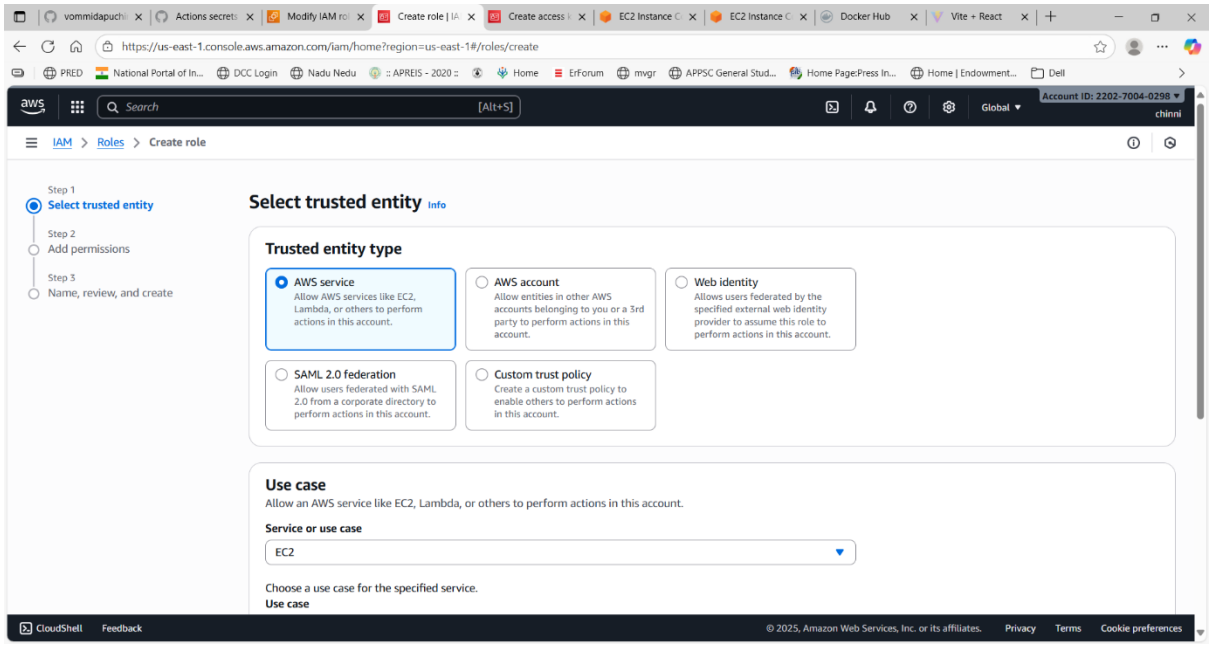


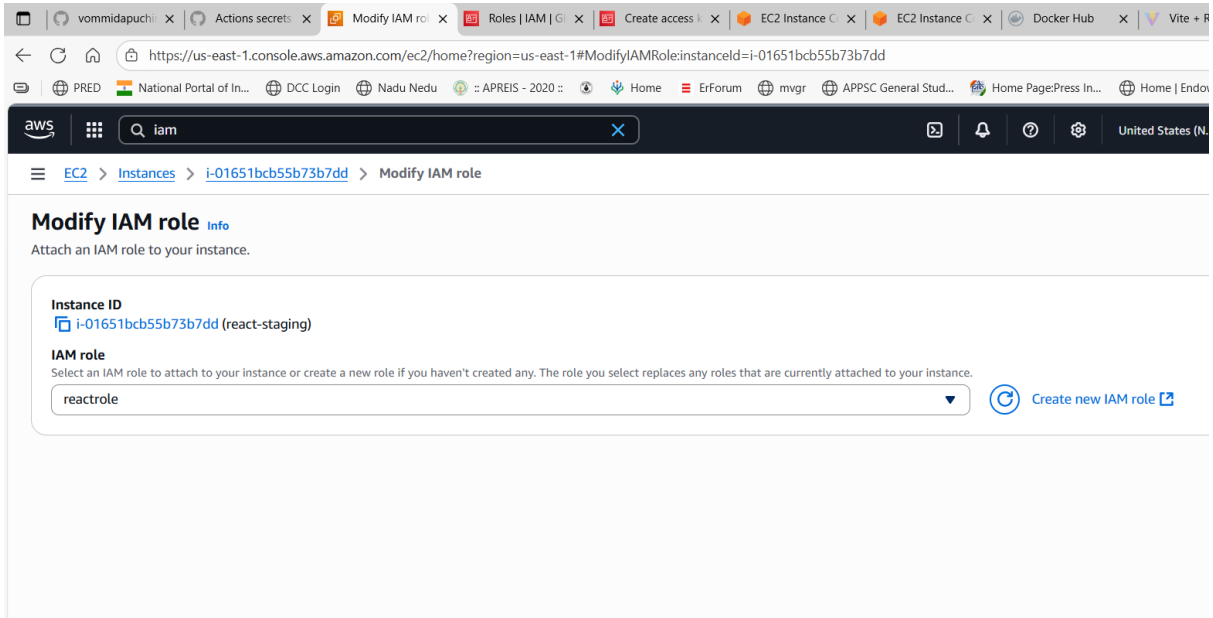
For cloud watching logs we have to give accesse for it by iam roles and keys.

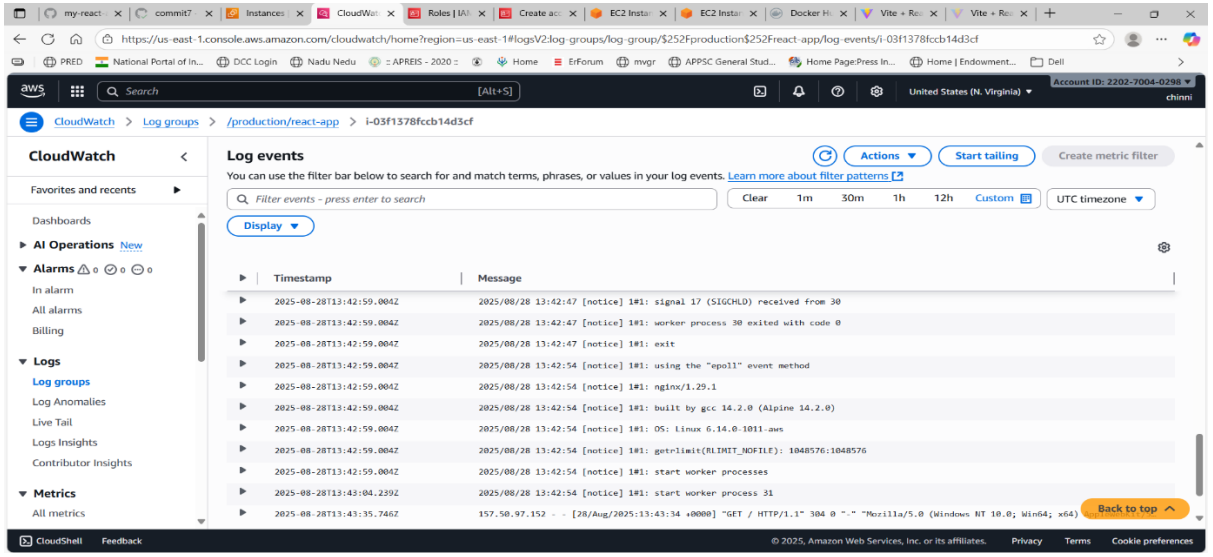

https://us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/users/details/chinni/create-access-key
IAM > Users > chinni > Create access key
Step 1
Access key best practices & alternatives
Step 2 - optional
Set description tag
Step 3
Retrieve access keys
Access key best practices & alternatives Info
Avoid using long-term credentials like access keys to improve your security. Consider the following use cases and alternatives.
Use case
Command Line Interface (CLI)
You plan to use this access key to enable the AWS CLI to access your AWS account.
Local code
You plan to use this access key to enable application code in a local development environment to access your AWS account.
Application running on an AWS compute service
You plan to use this access key to enable application code running on an AWS compute service like Amazon EC2, Amazon ECS, or AWS Lambda to access your AWS account.
Third-party service
You plan to use this access key to enable access for a third-party application or service that monitors or manages your AWS resources.
Application running outside AWS
You plan to use this access key to authenticate workloads running in your data center or other infrastructure outside of AWS that needs to access your AWS resources.
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

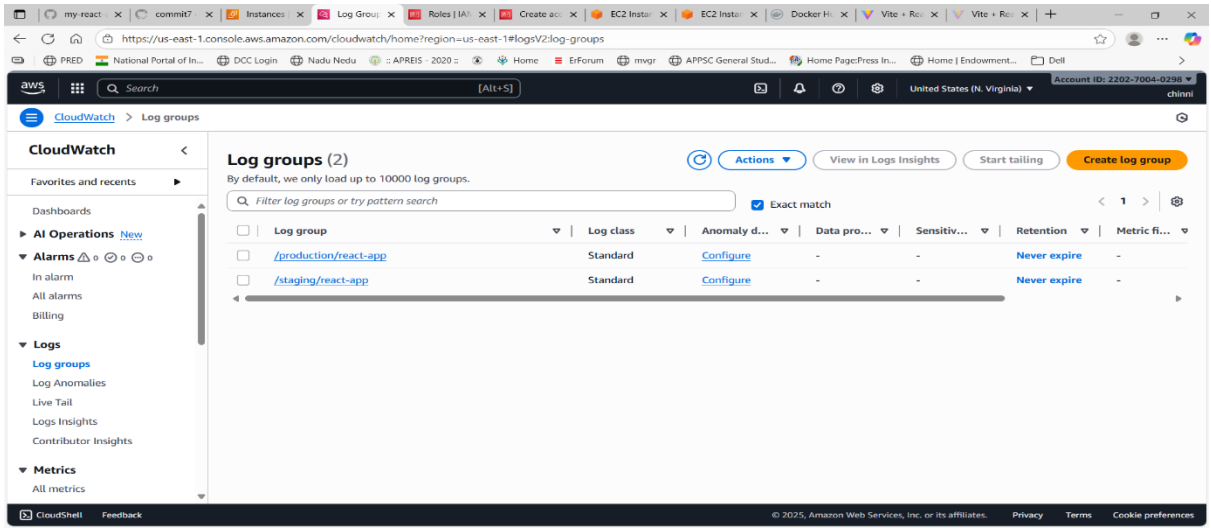

https://us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/users/details/chinni/create-access-key
IAM > Users > chinni > Create access key
Access key created
This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.
Step 1
Access key best practices & alternatives
Step 2 - optional
Set description tag
Step 3
Retrieve access keys
Retrieve access keys Info
Access key
If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.
Access key
Secret access key
AKIATGSI6CTVG2BLGQS4
********** Show
Access key best practices
Never store your access key in plain text, in a code repository, or in code.
Disable or delete access key when no longer needed.
Enable least-privilege permissions.
Rotate access keys regularly.
For more details about managing access keys, see the best practices for managing AWS access keys.
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

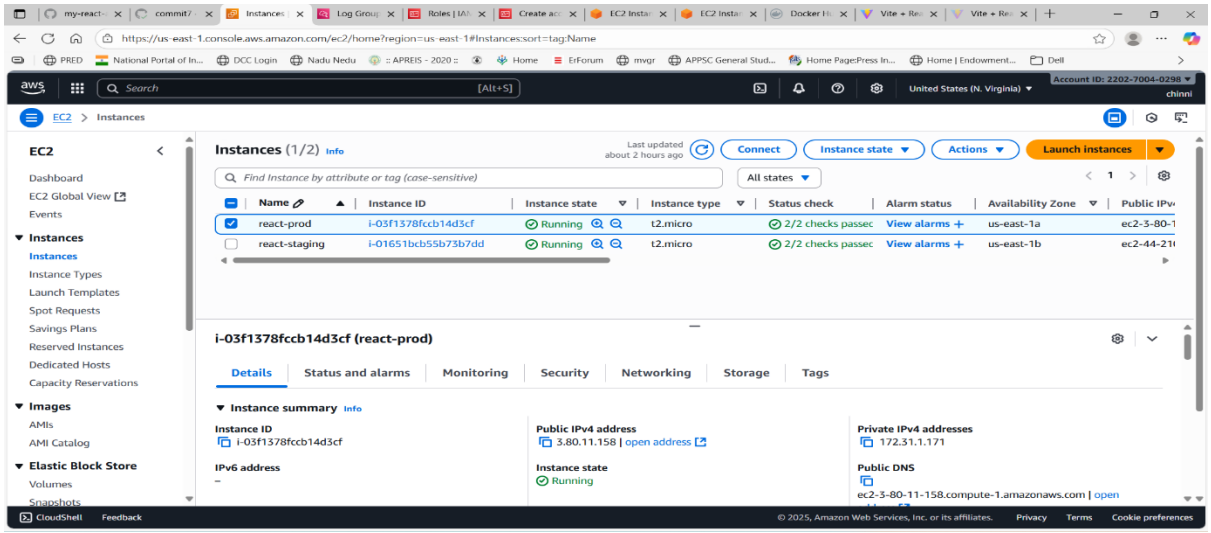

https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#InstanceDetails:instanceId=i-01651bcb55b73b7dd
EC2 > Instances > i-01651bcb55b73b7dd
EC2
Dashboard
EC2 Global View
Events
Instances
Instances
Instance Types
Launch Templates
Spot Requests
Savings Plans
Reserved Instances
Dedicated Hosts
Capacity Reservations
Images
AMIs
AMI Catalog
Elastic Block Store
Volumes
Snapshots
Instance summary for i-01651bcb55b73b7dd (react-staging) Info
Updated 23 minutes ago
Connect
Instance state
Actions
Instance diagnostics
Instance settings
Networking
Security
Image and templates
Monitor and troubleshoot
Change security groups
Get Windows password
Modify IAM role
Instance ID
i-01651bcb55b73b7dd
IPv6 address
–
Hostname type
IP name: ip-172-31-88-78.ec2.internal
Answer private resource DNS name
–
Auto-assigned IP address
44.210.136.12 [Public IP]
Public IPv4 address
44.210.136.12 | open address
Instance state
Running
Private IP DNS name (IPv4 only)
ip-172-31-88-78.ec2.internal
Instance type
t2.micro
VPC ID
vpc-083cd7a5e24de12da
Subnet ID
subnet-036a4db6f07c5bb95
Instance ARN
Private IPv4 address
172.31.88.78
Elastic IP addresses
–
AWS Compute Optimizer finding
Opt-in to AWS Compute Optimizer for recommendations.
Learn more
Auto Scaling Group name
–
IAM Role
–
IMDSv2
Managed
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences


https://us-east-1.console.aws.amazon.com/iam/home?region=us-east-1#/roles/create
IAM > Roles > Create role
Step 1
Select trusted entity
Step 2
Add permissions
Step 3
Name, review, and create
Select trusted entity Info
Trusted entity type
AWS service
Allow AWS services like EC2, Lambda, or others to perform actions in this account.
AWS account
Allow entities in other AWS accounts belonging to you or a 3rd party to perform actions in this account.
Web identity
Allows users federated by the specified external web identity provider to assume this role to perform actions in this account.
SAML 2.0 federation
Allow users federated with SAML 2.0 from a corporate directory to perform actions in this account.
Custom trust policy
Create a custom trust policy to enable others to perform actions in this account.
Use case
Allow an AWS service like EC2, Lambda, or others to perform actions in this account.
Service or use case
EC2
Choose a use case for the specified service.
Use case
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

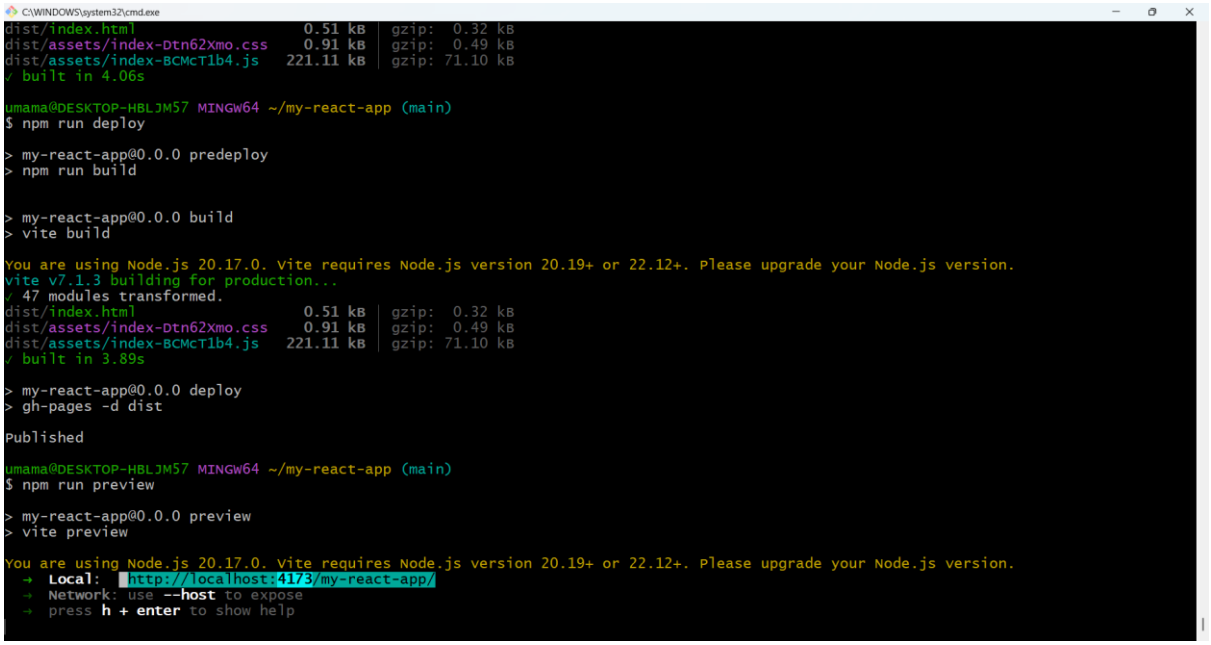

https://us-east-1.console.aws.amazon.com/ec2/home?region=us-east-1#ModifyIAMRole:instanceId=i-01651bcb55b73b7dd
iam
EC2 > Instances > i-01651bcb55b73b7dd > Modify IAM role
Modify IAM role Info
Attach an IAM role to your instance.
Instance ID
i-01651bcb55b73b7dd (react-staging)
IAM role
Select an IAM role to attach to your instance or create a new role if you haven't created any. The role you select replaces any roles that are currently attached to your instance.
reactrole
Create new IAM role

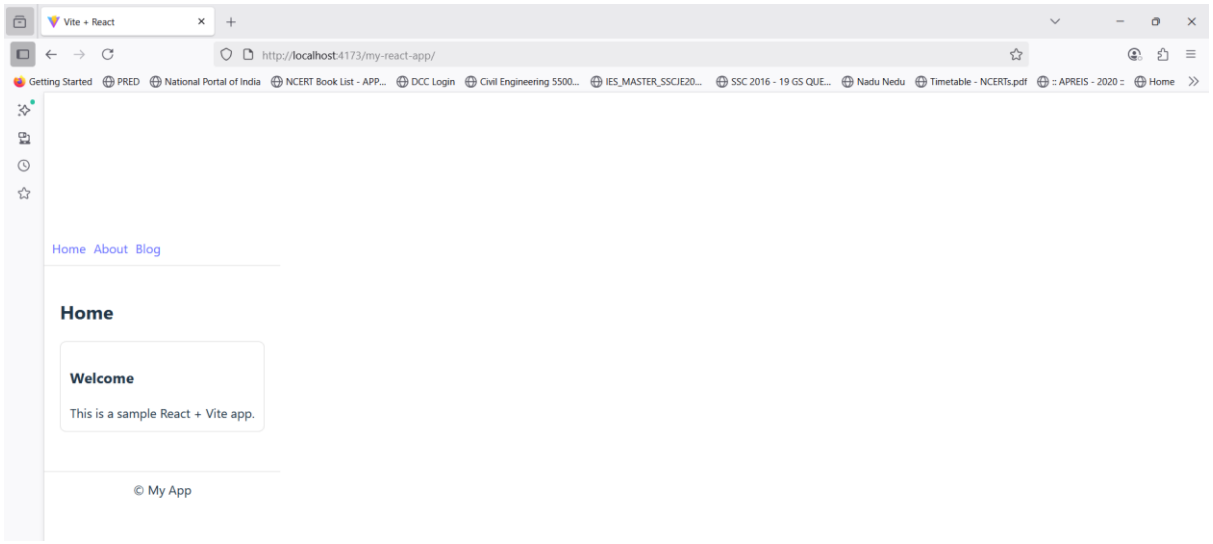

https://us-east-1.console.aws.amazon.com/cloudwatch/home?region=us-east-1#logsV2:log-groups/log-group/$252Fproduction$252Freact-app/log-events/i-03f1378fccb14d3cf
CloudWatch > Log groups > /production/react-app > i-03f1378fccb14d3cf
CloudWatch
Favorites and recents
Dashboards
AI Operations New
Alarms
In alarm
All alarms
Billing
Logs
Log groups
Log Anomalies
Live Tail
Logs Insights
Contributor Insights
Metrics
All metrics
Log events
Actions
Start tailing
Create metric filter
You can use the filter bar below to search for and match terms, phrases, or values in your log events. Learn more about filter patterns
Filter events - press enter to search
Clear
1m
30m
1h
12h
Custom
UTC timezone
Display
Timestamp
Message
2025-08-28T13:42:59.004Z 2025/08/28 13:42:47 [notice] 1#1: signal 17 (SIGCHLD) received from 30
2025-08-28T13:42:59.004Z 2025/08/28 13:42:47 [notice] 1#1: worker process 30 exited with code 0
2025-08-28T13:42:59.004Z 2025/08/28 13:42:47 [notice] 1#1: exit
2025-08-28T13:42:59.004Z 2025/08/28 13:42:54 [notice] 1#1: using the "epoll" event method
2025-08-28T13:42:59.004Z 2025/08/28 13:42:54 [notice] 1#1: nginx/1.29.1
2025-08-28T13:42:59.004Z 2025/08/28 13:42:54 [notice] 1#1: built by gcc 14.2.0 (Alpine 14.2.0)
2025-08-28T13:42:59.004Z 2025/08/28 13:42:54 [notice] 1#1: OS: Linux 6.14.0-1011-aws
2025-08-28T13:42:59.004Z 2025/08/28 13:42:54 [notice] 1#1: getrlimit(RLIMIT_NOFILE): 1048576:1048576
2025-08-28T13:42:59.004Z 2025/08/28 13:42:54 [notice] 1#1: start worker processes
2025-08-28T13:43:04.239Z 2025/08/28 13:42:54 [notice] 1#1: start worker process 31
2025-08-28T13:43:35.746Z 157.50.97.152 - - [28/Aug/2025:13:43:34 +0000] "GET / HTTP/1.1" 304 0 "-" "Mozilla/5.0 (Windows NT 10.0; Win64; x64)
Back to top
CloudShell
Feedback
© 2025, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

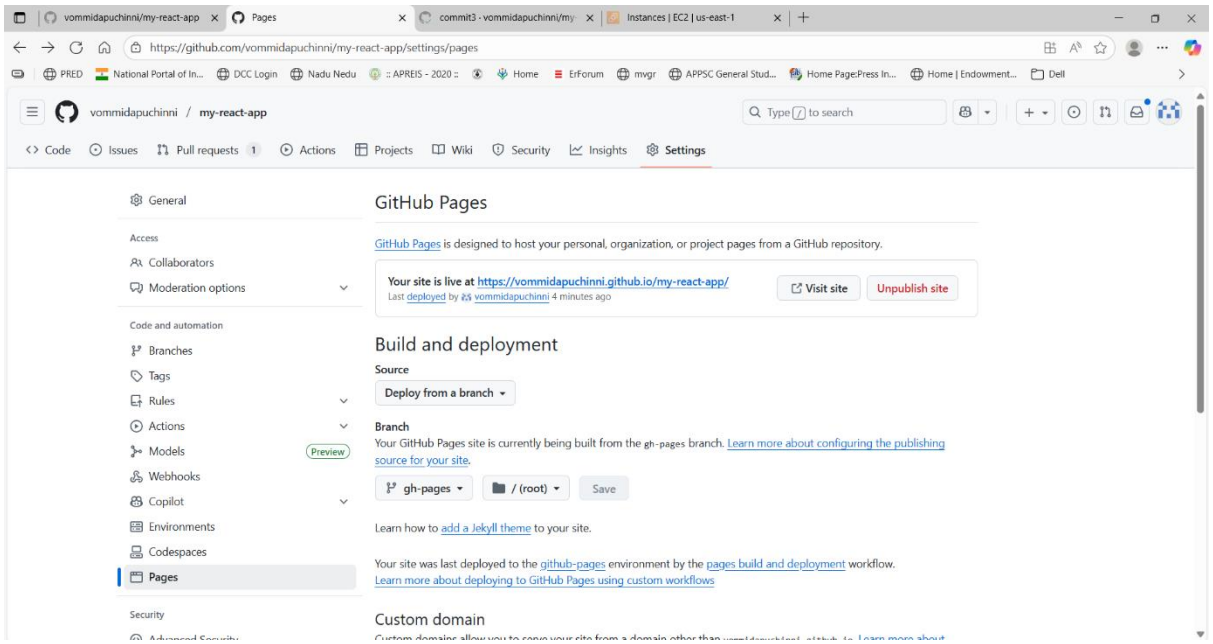


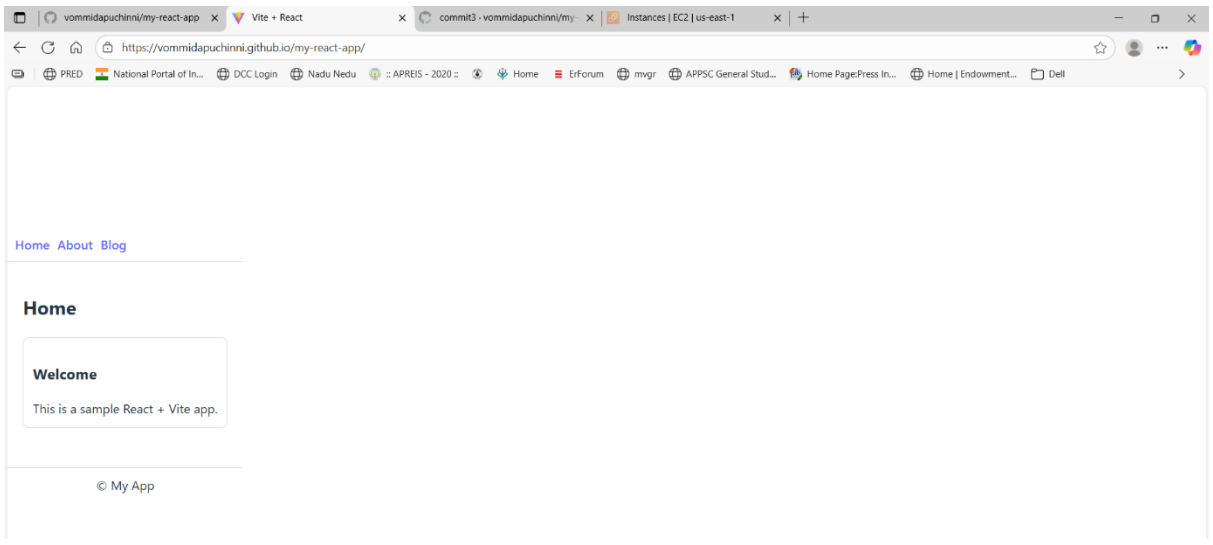


To keep my project live I am using github pages, to that I edited package.json and main.jsx files.




Vite + React
http://localhost:4173/my-react-app/
Home About Blog
Home
Welcome
This is a sample React + Vite app.
© My App


https://github.com/vommidapuchinni/my-react-app/settings/pages
vommidapuchinni / my-react-app
Code Issues Pull requests Actions Projects Wiki Security Insights Settings
General
Access
Collaborators
Moderation options
Code and automation
Branches
Tags
Rules
Actions
Models
Preview
Webhooks
Copilot
Environments
Codespaces
Pages
Security
GitHub Pages
GitHub Pages is designed to host your personal, organization, or project pages from a GitHub repository.
Your site is live at https://vommidapuchinni.github.io/my-react-app/
Last deployed by vommidapuchinni 4 minutes ago
Visit site
Unpublish site
Build and deployment
Source
Deploy from a branch
Branch
Your GitHub Pages site is currently being built from the gh-pages branch. Learn more about configuring the publishing source for your site.
gh-pages
/ (root)
Save
Learn how to add a Jekyll theme to your site.
Your site was last deployed to the github-pages environment by the pages build and deployment workflow.
Learn more about deploying to GitHub Pages using custom workflows
Custom domain


https://vommidapuchinni.github.io/my-react-app/
Home About Blog
Home
Welcome
This is a sample React + Vite app.
© My App

Troubleshooting

- **Container restarts** → check Dockerfile build stage.
- **Logs not in CloudWatch** → verify agent config, JSON path, and volume mapping.
- **Workflow fails** → confirm SSH keys and IAM role permissions.
- **App not accessible** → check port mapping and EC2 security group.

Key Concepts

- **Docker multi-stage build**: Node.js build → Nginx serve.
- **Docker Compose**: defines service, ports, and log volume.
- **CloudWatch Agent**: collects logs from host directory to log groups.
- **CI/CD with GitHub Actions**: auto-deploys on branch push using SSH.
- **Monitoring**: logs viewable in CloudWatch for staging (/staging/react-app) and production (/production/react-app).

Conclusion

Successfully deployed a React app with **staging and production environments** on AWS EC2 using **Docker** and **GitHub Actions CI/CD**. Logs are collected in **CloudWatch** for monitoring.