**Application:**

Clone the below mentioned repository and deploy the application (Run application in port 3000).

Repo URL : <https://github.com/Vennilavan12/Brain-Tasks-App.gitCommand>

**Step-by-step conceptual explanations**

**1) Source control (GitHub)**

What it is

* A central place to store code, history, and collaboration (commits, branches, PRs).  
  Why it matters
* Everything in the pipeline is triggered from here. The pipeline watches the repo and reacts to changes.  
  Teaching points
* Explain commits, branches, and the commit→trigger model: push code → pipeline starts.

**2) Run locally (verify the app)**

What it is

* Execute the app on your laptop to confirm it actually works before packaging.  
  Why it matters
* Fixing bugs locally is faster and isolates problems from infra issues later.  
  Teaching points
* Show npm start and npm run build to illustrate development vs production build.

**3) Containerization (Docker)**

What it is

* Packaging the app and its runtime into a portable image. Dockerfile is the recipe for that image.  
  Why it matters
* Image = same behaviour anywhere you run it. Developers, CI servers, and production get identical runtime.  
  Key concepts (simple):
* **Image**: read-only template (like a snapshot).
* **Container**: live running instance of an image.
* **Dockerfile**: stepwise instructions to build the image.
* **Multi-stage build**: build artifacts in one stage, copy only final assets into a small runtime image (keeps images small).  
  Teaching points
* Explain "works on my machine" problem and how containers solve it.

**4) Container registry (Amazon ECR)**

What it is

* A private store for Docker images (like Docker Hub but private to your AWS account).  
  Why it matters
* Kubernetes and the CI system pull images from a central place; storing images in ECR integrates with AWS IAM for controlled access.  
  Key concepts:
* **Repository**: logical group of images.
* **Tag**: image label (use git SHA, not latest, for reproducibility).  
  Teaching points
* Demonstrate push/pull lifecycle: build → tag → push → pull.

**5) Kubernetes basics (EKS)**

What it is

* Kubernetes is an orchestration system that runs containers across many machines, and EKS is AWS’s managed Kubernetes service.  
  Why it matters
* Kubernetes manages running containers (scheduling, scaling, updates) so applications stay available under load or failure.  
  Core objects (plain terms):
* **Cluster**: the whole Kubernetes installation (control plane + worker nodes).
* **Node**: a machine (VM) that runs containers.
* **Pod**: the smallest run unit; one or more containers together.
* **Deployment**: describes how many replicas of a pod should run and how to upgrade them.
* **Service**: stable network endpoint to access pods; LoadBalancer type gives an external IP/DNS.
* **Ingress** (optional): manages HTTP routing, TLS termination, and virtual hosts.  
  Teaching points
* Show mapping: Docker container → Pod → Deployment + Service exposes it.

**6) Kubernetes manifests (YAML) — declarative approach**

What it is

* YAML files declare desired state (e.g., “I want 3 replicas running image X”).  
  Why it matters
* Kubernetes reconciles actual state to desired state automatically — more predictable than ad-hoc commands.  
  Important fields to explain:
* spec.replicas, spec.template, containers[].image, ports
* **Readiness probe**: says “only route traffic here when this check passes.”
* **Liveness probe**: says “restart this container if this check fails.”  
  Teaching points
* Emphasize declarative vs imperative: you commit YAML to version control and apply it — cluster converges.

**7) CI/CD: CodeBuild + CodePipeline**

What they are

* **CodeBuild**: runs build jobs (execute scripts, compile, build Docker images).
* **CodePipeline**: orchestrates stages (source, build, deploy) and wires services together.  
  Why it matters
* Automation: no manual rebuild/push/deploy. A commit → automatic build → push → deploy pipeline ensures consistency and speed.  
  How it typically flows (conceptually)

1. **Source stage**: pipeline detects a commit.
2. **Build stage**: CodeBuild pulls code, runs tests, builds Docker image, and pushes it to ECR.
3. **Deploy stage**: an action (often another CodeBuild step) updates Kubernetes (for example via kubectl set image or applying manifests).  
   Key concept: **buildspec.yml** is the script the build server reads — it says what to run in each phase.  
   Teaching points

* Emphasize atomic builds and storing artifact metadata (image tag, artifact manifest).

**8) Authentication & permissions (IAM)**

What it is

* AWS IAM controls who/what can do actions (push to ECR, create EKS resources, call AWS APIs).  
  Why it matters
* The pipeline and Kubernetes nodes need specific permissions — too little and deployment fails; too much and you create risk.  
  Important ideas:
* **Service roles** for CodeBuild: give minimal permissions (ECR push, CloudWatch logs, S3 artifacts).
* **Node role / aws-auth mapping**: give worker nodes and CI roles permission to pull images or call the Kubernetes API.  
  Teaching points
* Show principle of least privilege vs practical lab comfort (for a lab you may use broader permissions, but point out production differences).

**9) Networking & Load Balancers**

What it is

* When you expose your app, AWS creates a Load Balancer (ALB/NLB) and attaches it to the service so external users reach the app.  
  Why it matters
* Load balancer gives a stable public endpoint and manages traffic distribution, security groups, and health checks.  
  Teaching points
* Differentiate NodePort vs LoadBalancer vs Ingress in simple terms: LoadBalancer = public IP; Ingress = smart HTTP routing (domain + TLS).

**10) Monitoring & Logging (CloudWatch)**

What it is

* Collect logs and metrics from builds and the cluster so you can troubleshoot and measure performance.  
  Why it matters
* If something breaks, logs and metrics are how you find the cause quickly.  
  Key parts:
* **Build logs**: CodeBuild sends logs to CloudWatch.
* **Pod logs**: either kubectl logs for quick checks or a logging agent (Fluent Bit) to forward logs to CloudWatch.
* **Metrics**: CPU, memory, pod restarts (Container Insights).  
  Teaching points
* Explain differences: logs (what happened) vs metrics (how healthy is the system). Stress alerts for important signals.

**11) Deployment strategies and rollbacks**

What it is

* How you update running apps safely without downtime.  
  Common strategies (simple):
* **Rolling update**: replace pods gradually (default Deployment behavior).
* **Canary**: deploy to a small subset first, observe, then increase.
* **Blue/Green**: keep old version running while routing traffic to a new environment, then switch.  
  Why it matters
* Minimize user impact and make rollbacks quick.  
  Teaching points
* Use immutable tags (git SHA) for images so rollbacks are deterministic (kubectl rollout undo).

**12) Security & best practices (high level)**

Key recommendations to teach:

* Don’t bake secrets into images; use **AWS Secrets Manager** or Kubernetes Secrets.
* Use **least privilege** IAM.
* Tag images with commit SHA, not latest.
* Scan images for vulnerabilities (image scanning).
* Set resource requests/limits for pods to avoid noisy neighbors.
* Use readiness/liveness probes and non-root containers where possible.  
  Teaching points
* Mention network policies: you can restrict pod-to-pod traffic in a cluster.

**13) Cost awareness & cleanup**

What to watch for

* EKS control plane (charged), worker nodes (EC2 costs), Load Balancers (per hour + data), ECR storage.  
  Teaching points
* For labs, use small node types and delete cluster after demo to avoid surprises.

Command :

1 mkdir project

2 chmod 777 project/

3 cd project/

4 sudo apt install git-all

5 apt-get install git

6 sudo apt-get install git

7 sudo add-apt-repository ppa:git-core/ppa

8 sudo apt update;sudo apt install git

9 git clone https://github.com/Vennilavan12/Brain-Tasks-App.git

10 ls -lrth

11 cd Brain-Tasks-App/



**Docker:**

* Dockerize the application by creating Dockerfile
* Build an application and check output using docker image.

Command:

17 vi Dockerfile

18 chmod 777 \*

19 docker build -t brain-app:latest .

20 sudo apt-get update

21 sudo apt-get install ca-certificates curl

22 sudo install -m 0755 -d /etc/apt/keyrings

23 sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc

24 sudo chmod a+r /etc/apt/keyrings/docker.asc

25 echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] https://download.docker.com/linux/ubuntu \

$(. /etc/os-release && echo "${UBUNTU\_CODENAME:-$VERSION\_CODENAME}") stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

26 sudo apt-get update

27 sudo apt-get install docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-compose-plugin

28 sudo systemctl status docker

29 sudo systemctl start docker

30 sudo docker build -t brain-app:latest .

31 sudo docker images

32 docker run -d -p 3000:80 --name brain-tasks-container brain-app:latest

33 sudo docker run -d -p 3000:80 --name brain-tasks-container brain-app:latest

34 sudo docker ps

35 history

ubuntu@ip-172-31-31-89:~/project/Brain-Tasks-App$ cat Dockerfile

# Use official Nginx image

FROM nginx:alpine

# Remove default Nginx website

RUN rm -rf /usr/share/nginx/html/\*

# Copy the build output to Nginx html directory

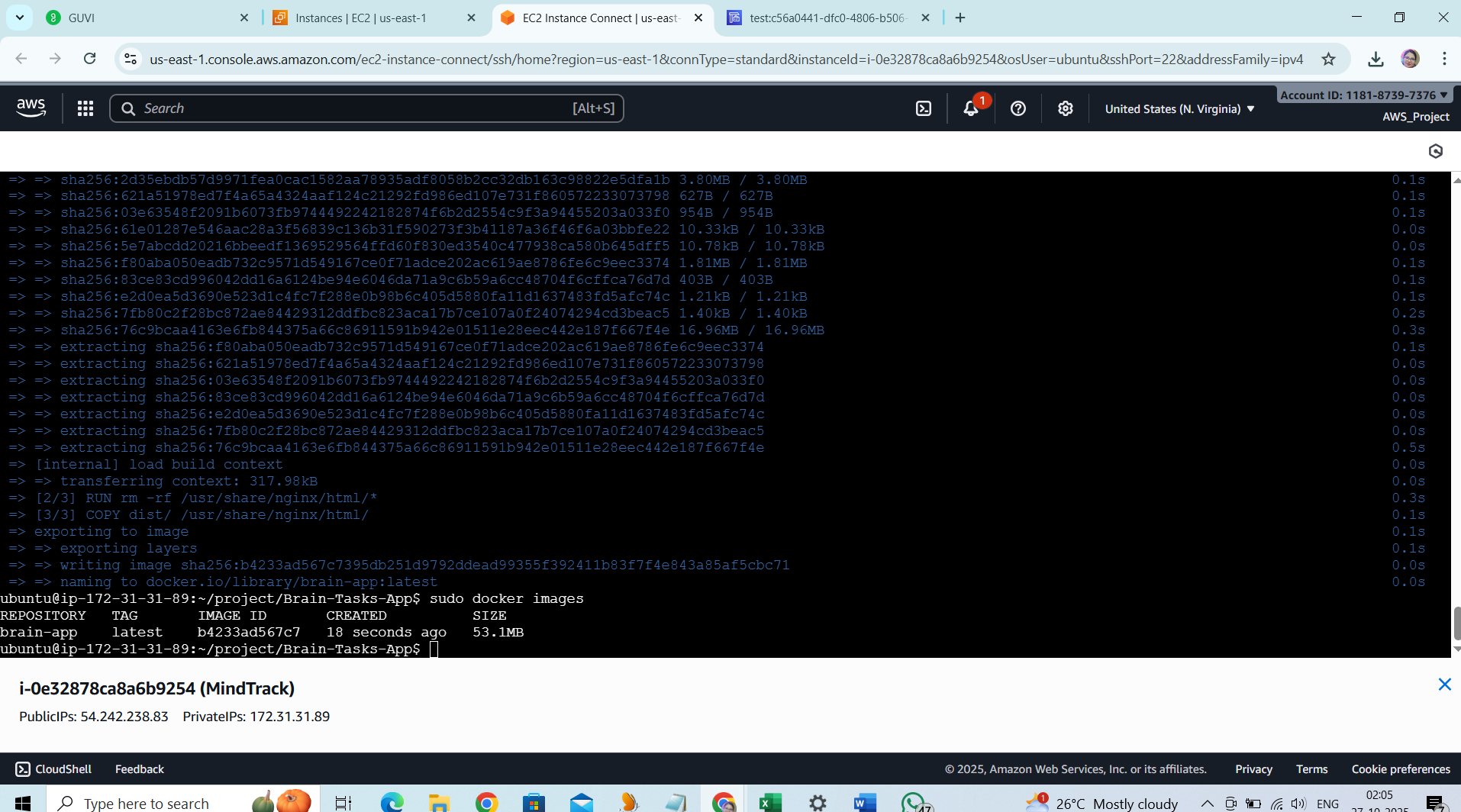
COPY dist/ /usr/share/nginx/html/

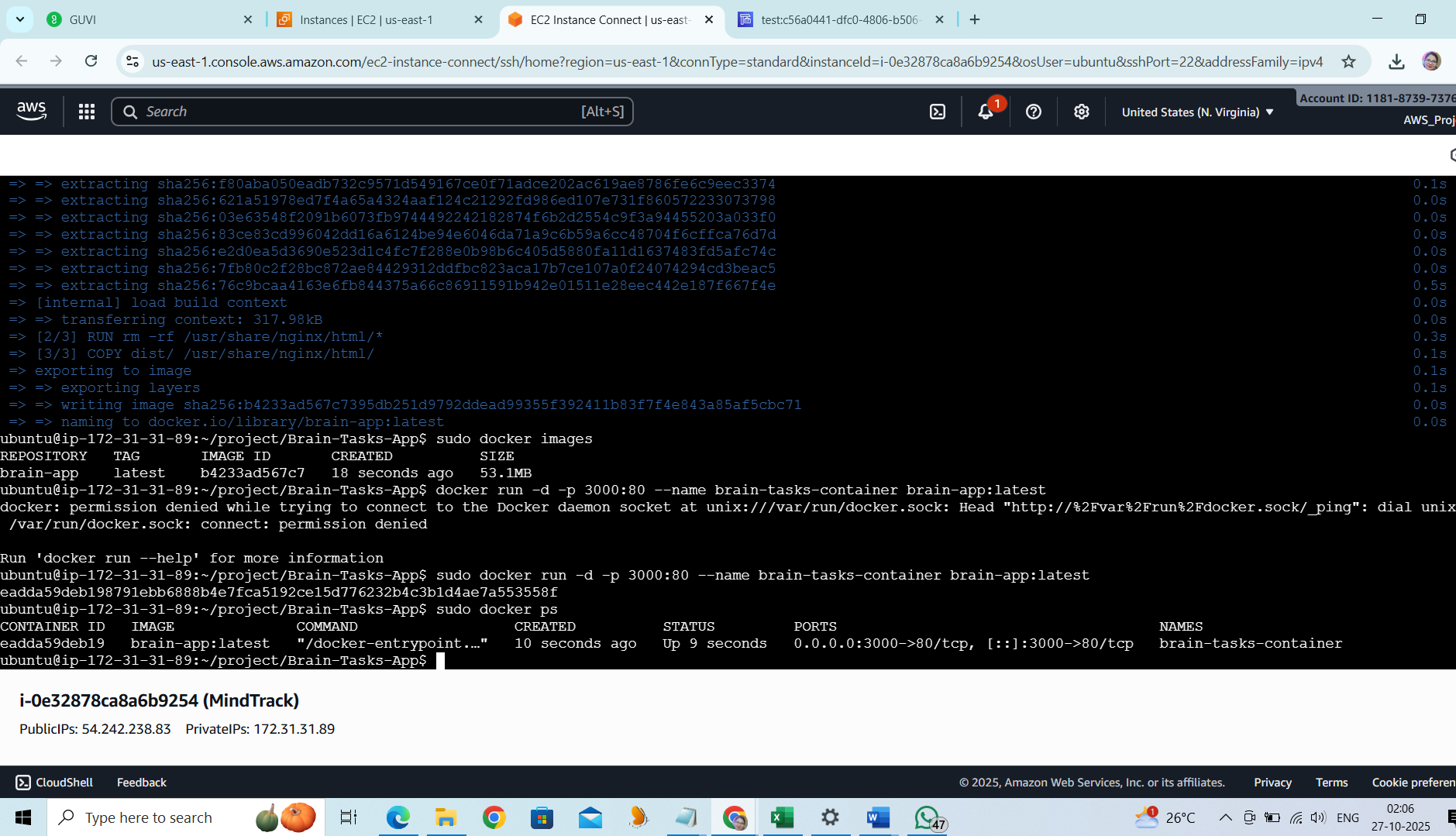
# Expose port 3000

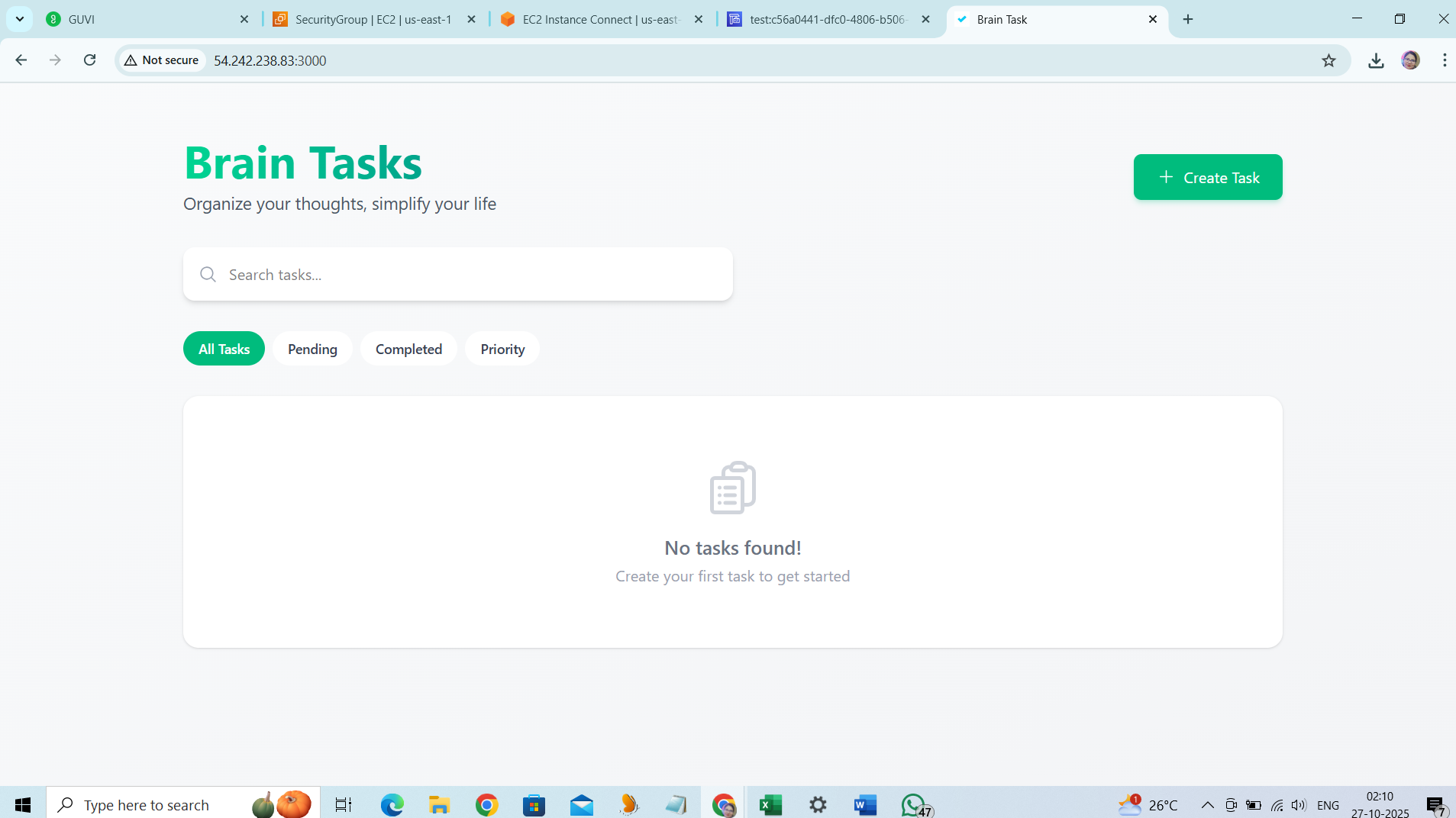
EXPOSE 80

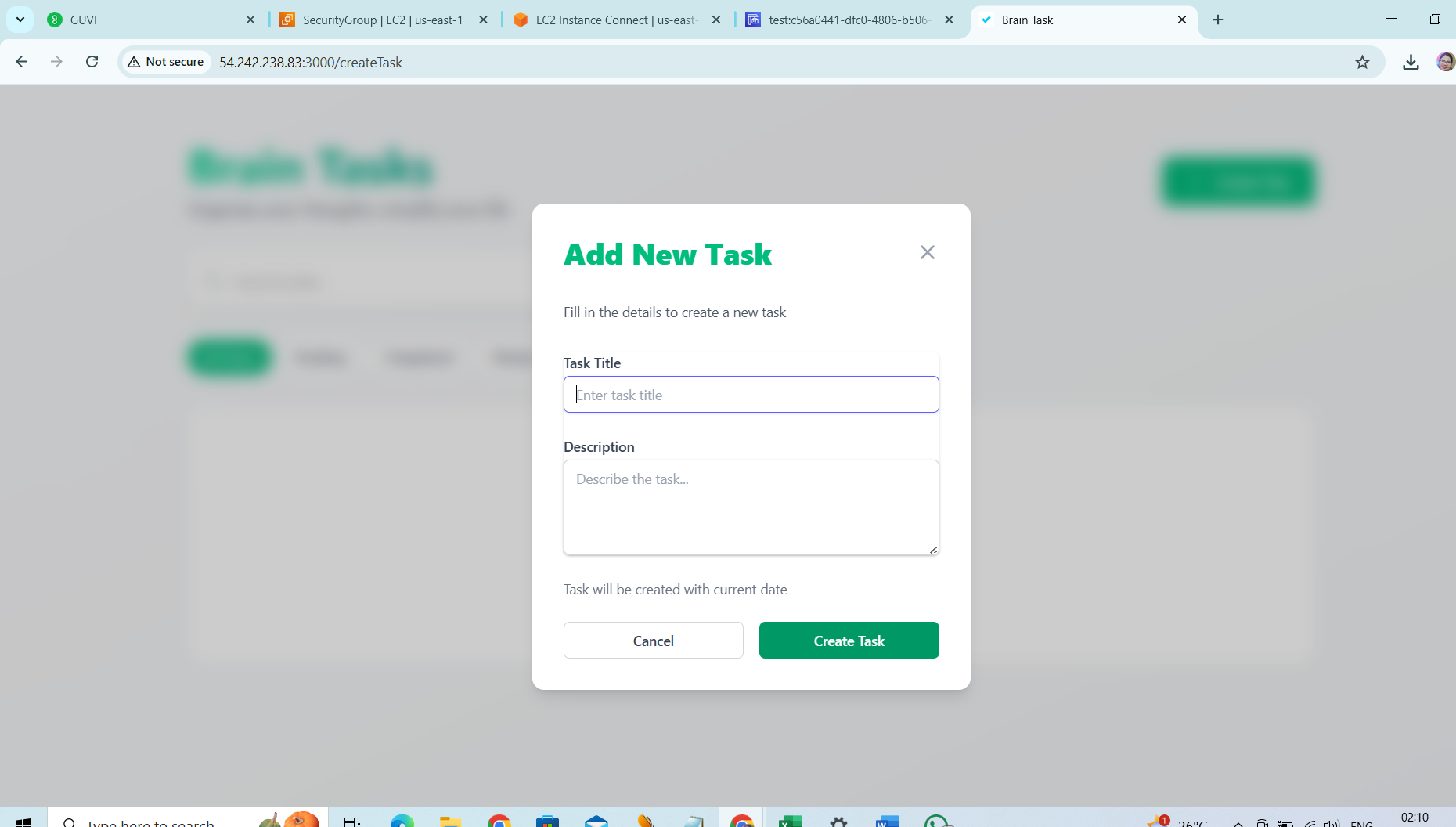
# Start Nginx

CMD ["nginx", "-g", "daemon off;"]



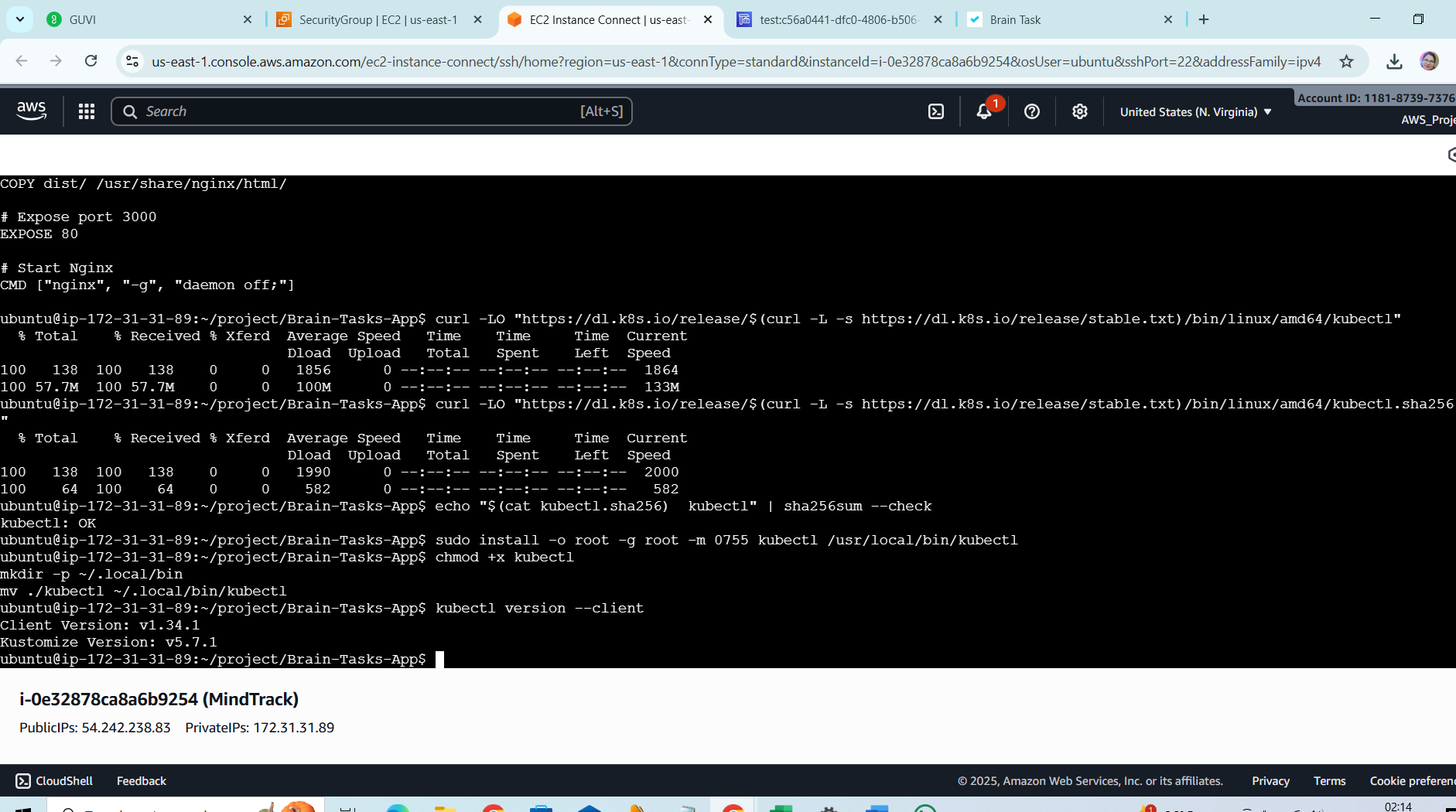


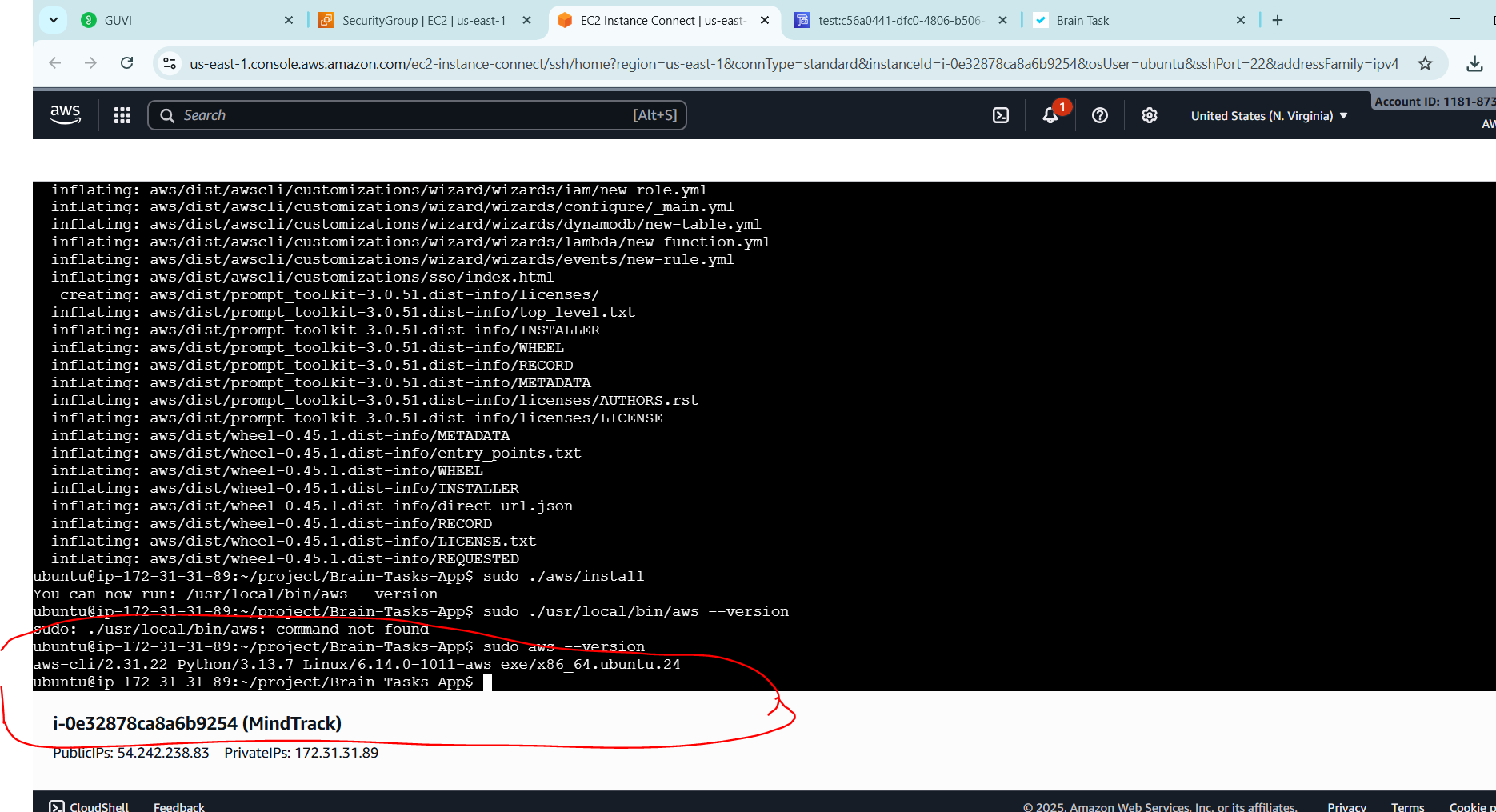


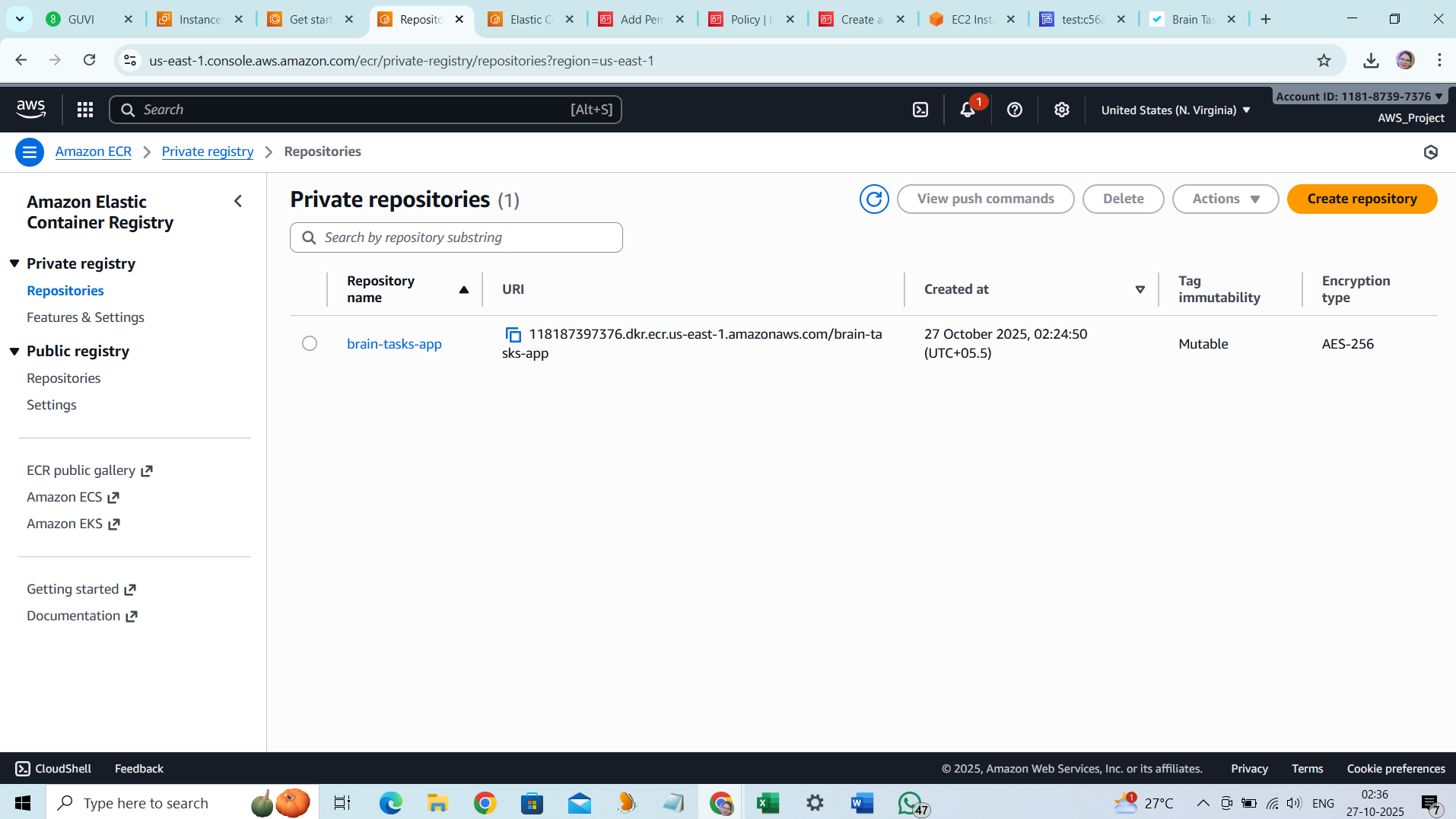


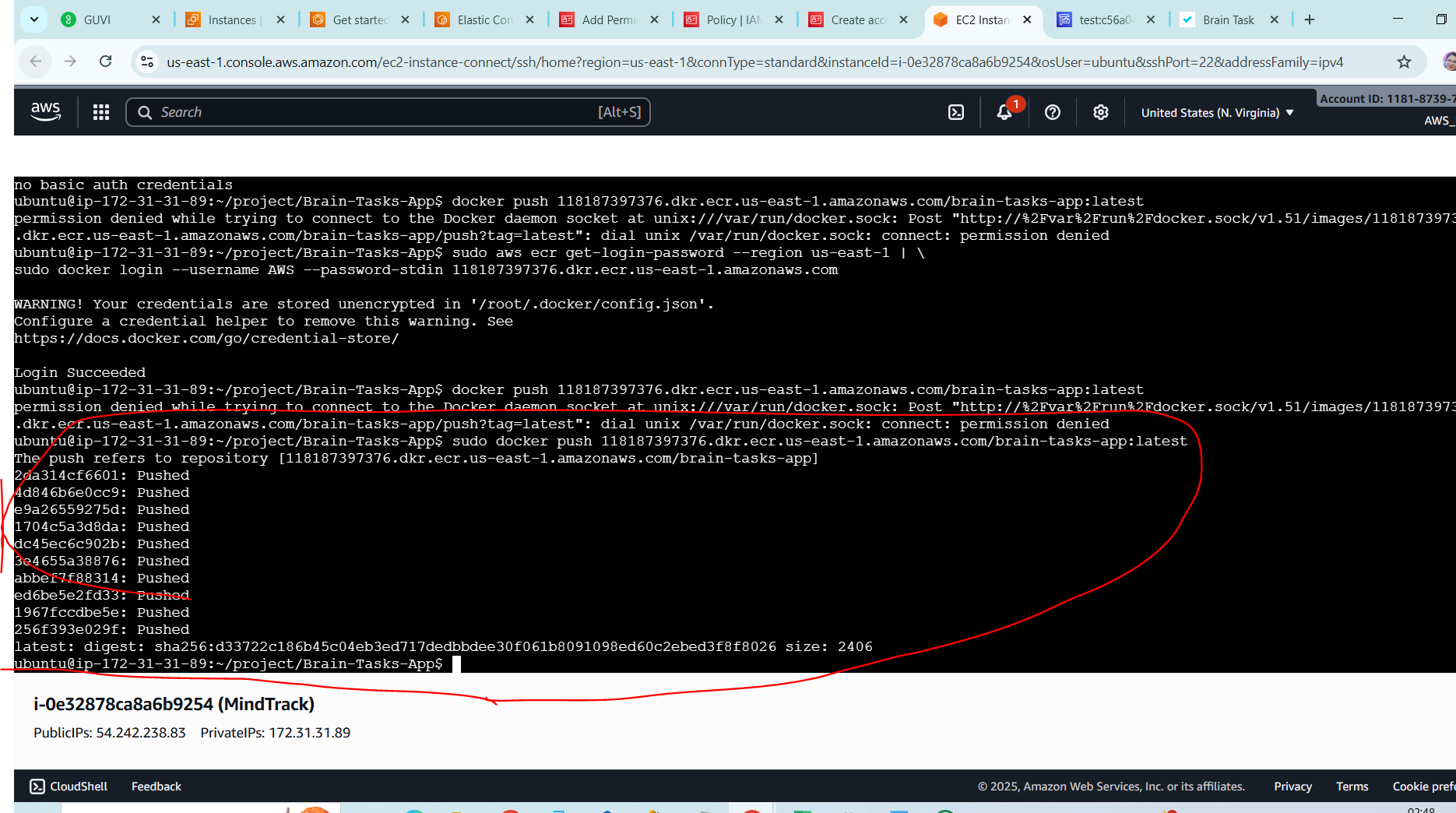
**ECR:**

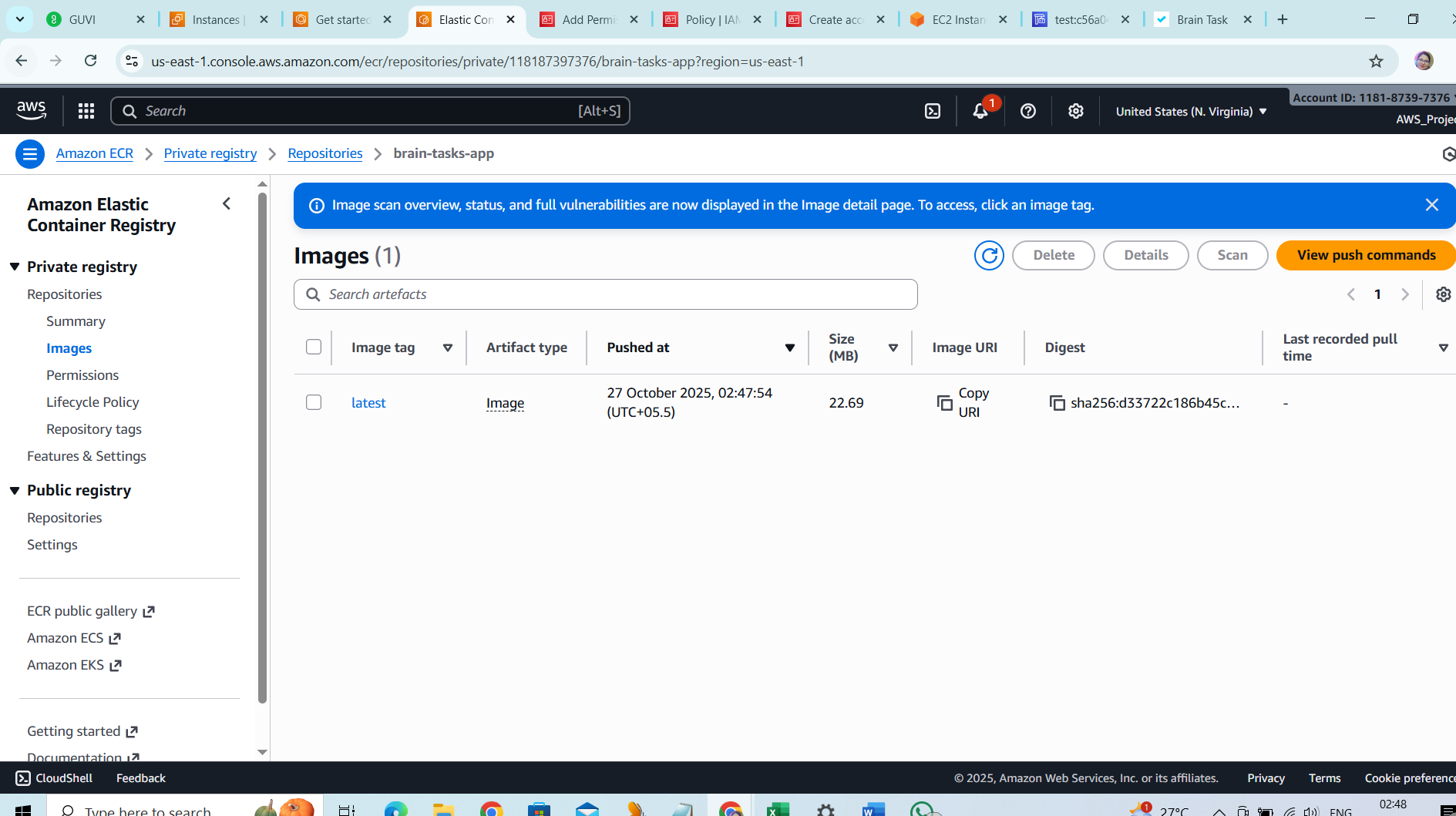
* Create an AWS ECR repository for store docker images.

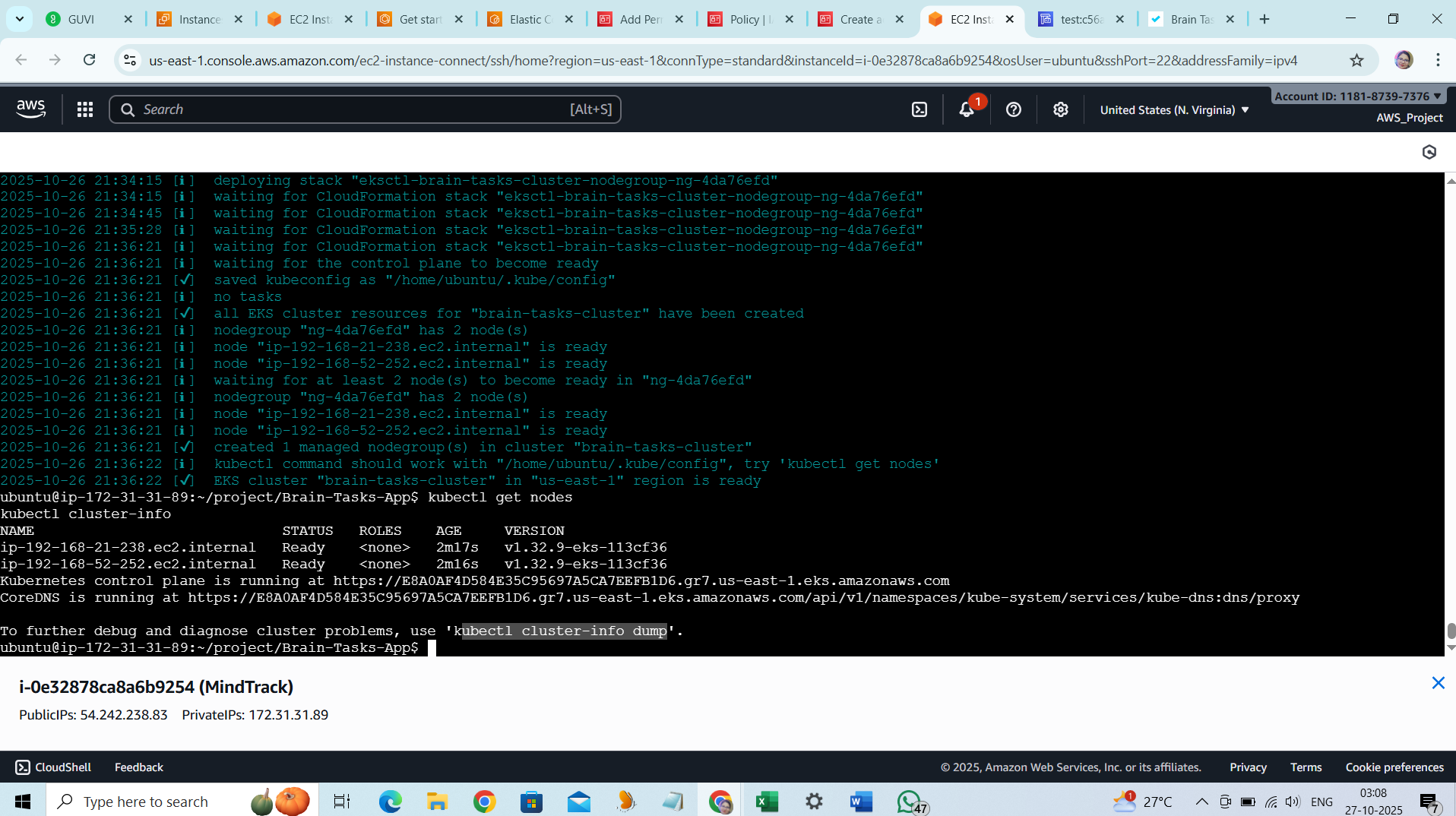


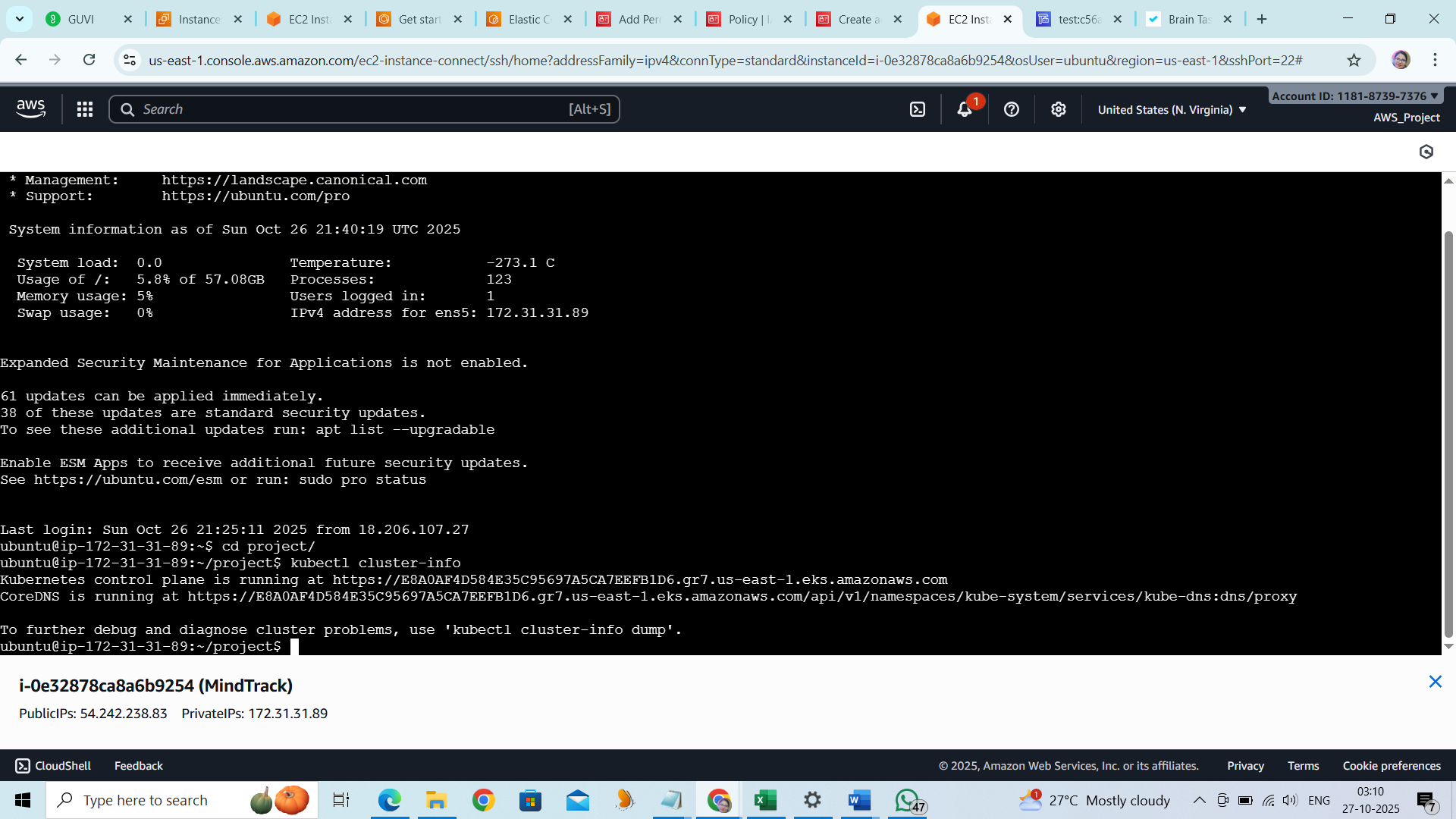


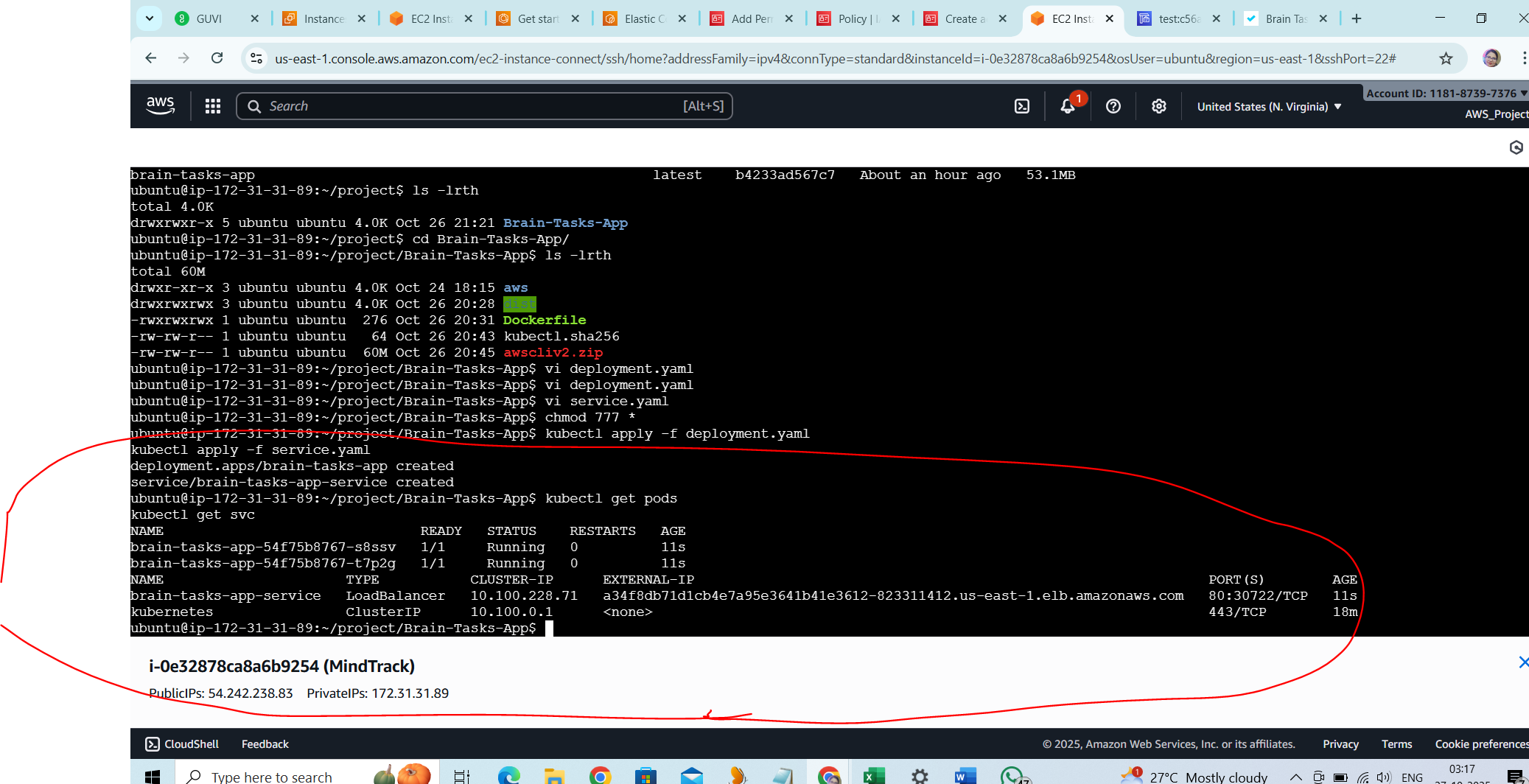




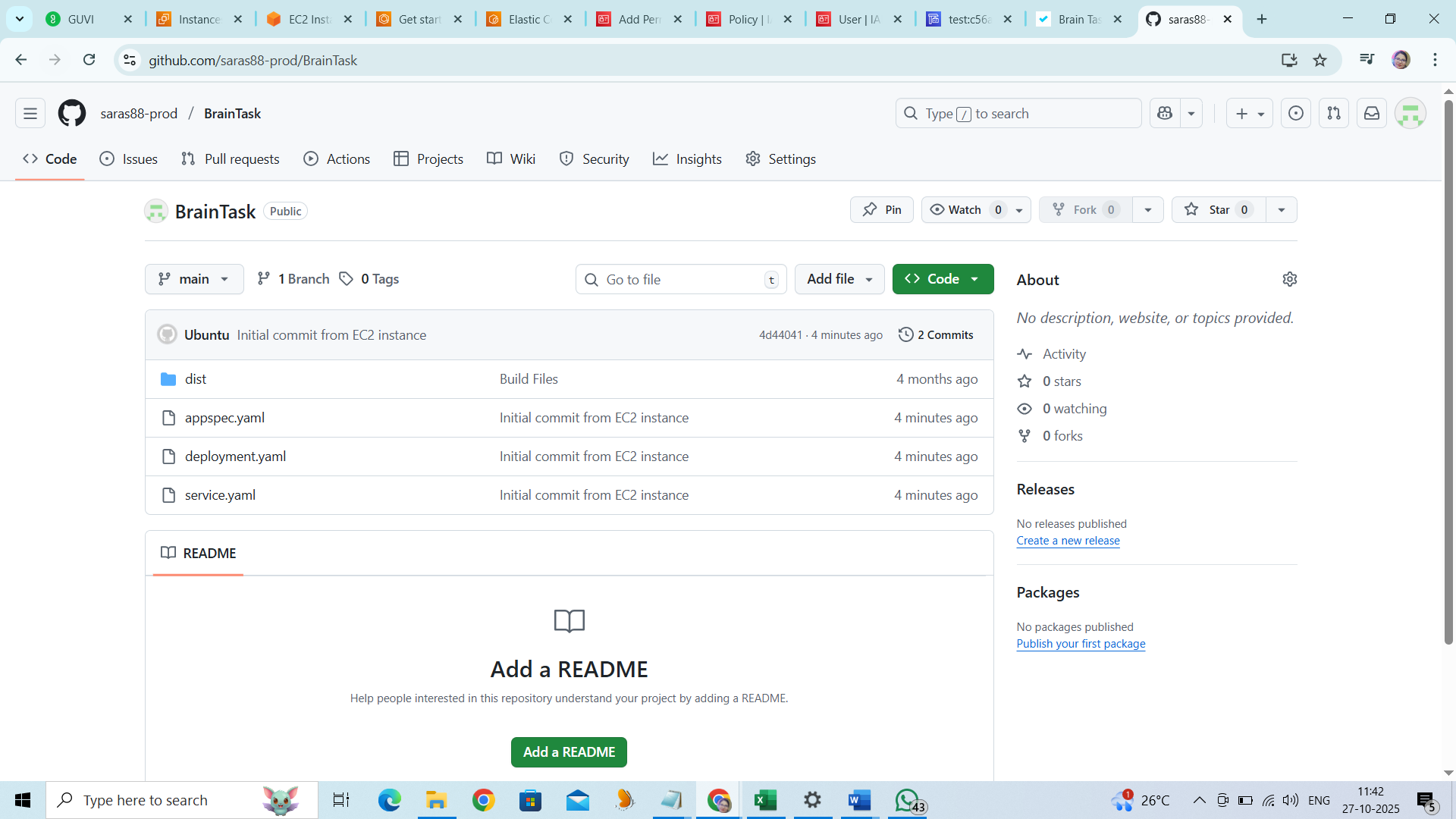


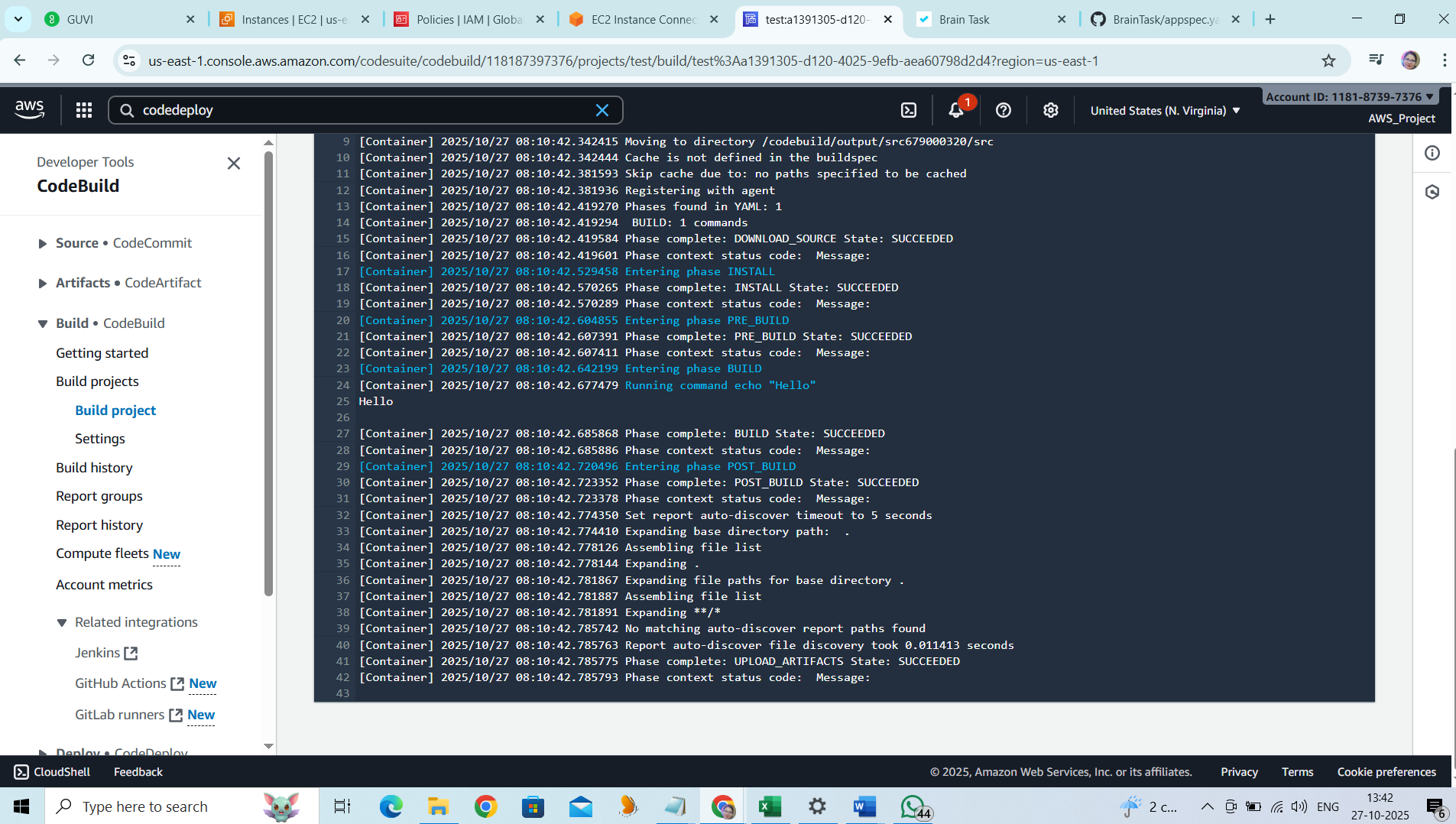




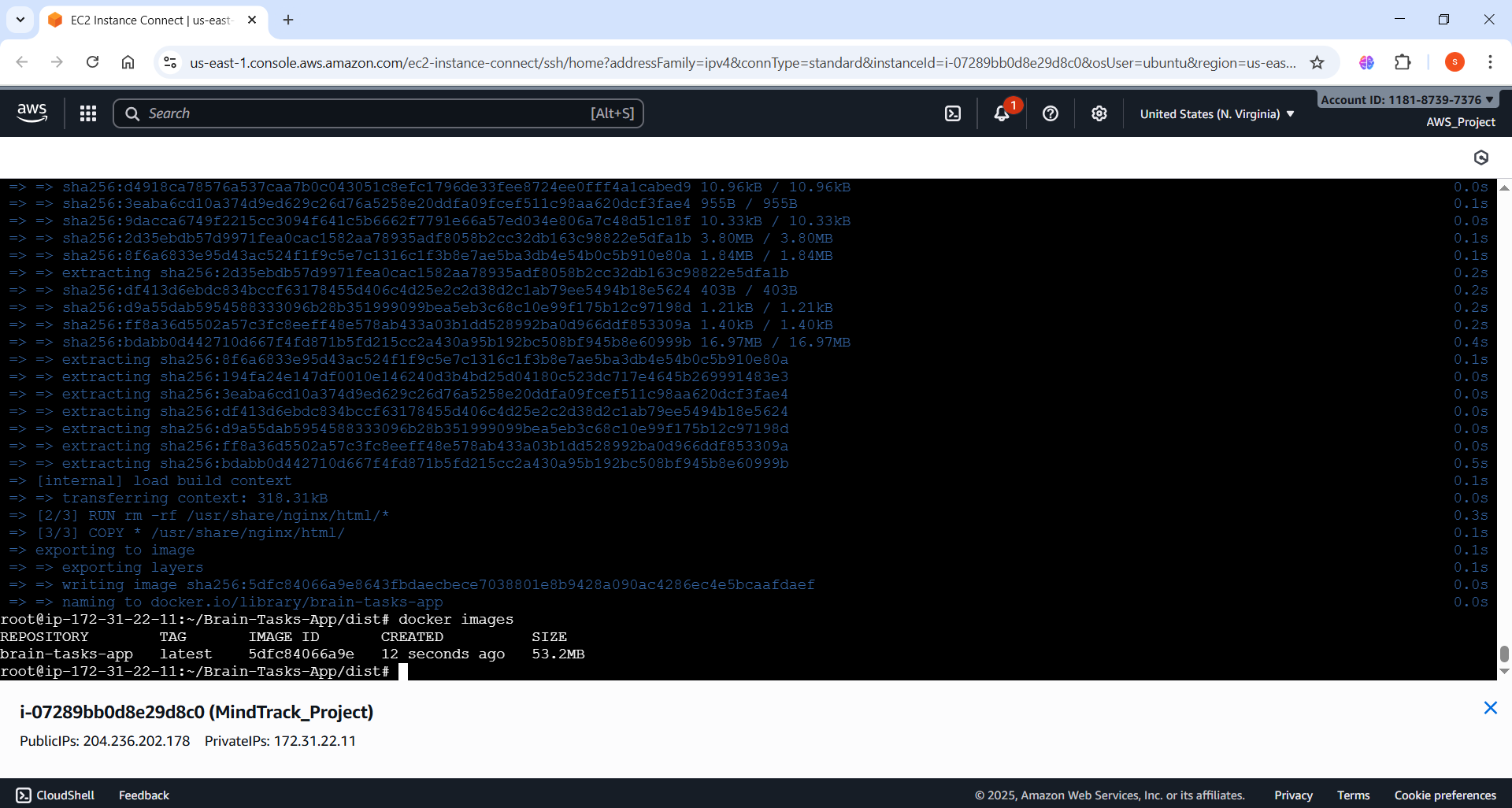


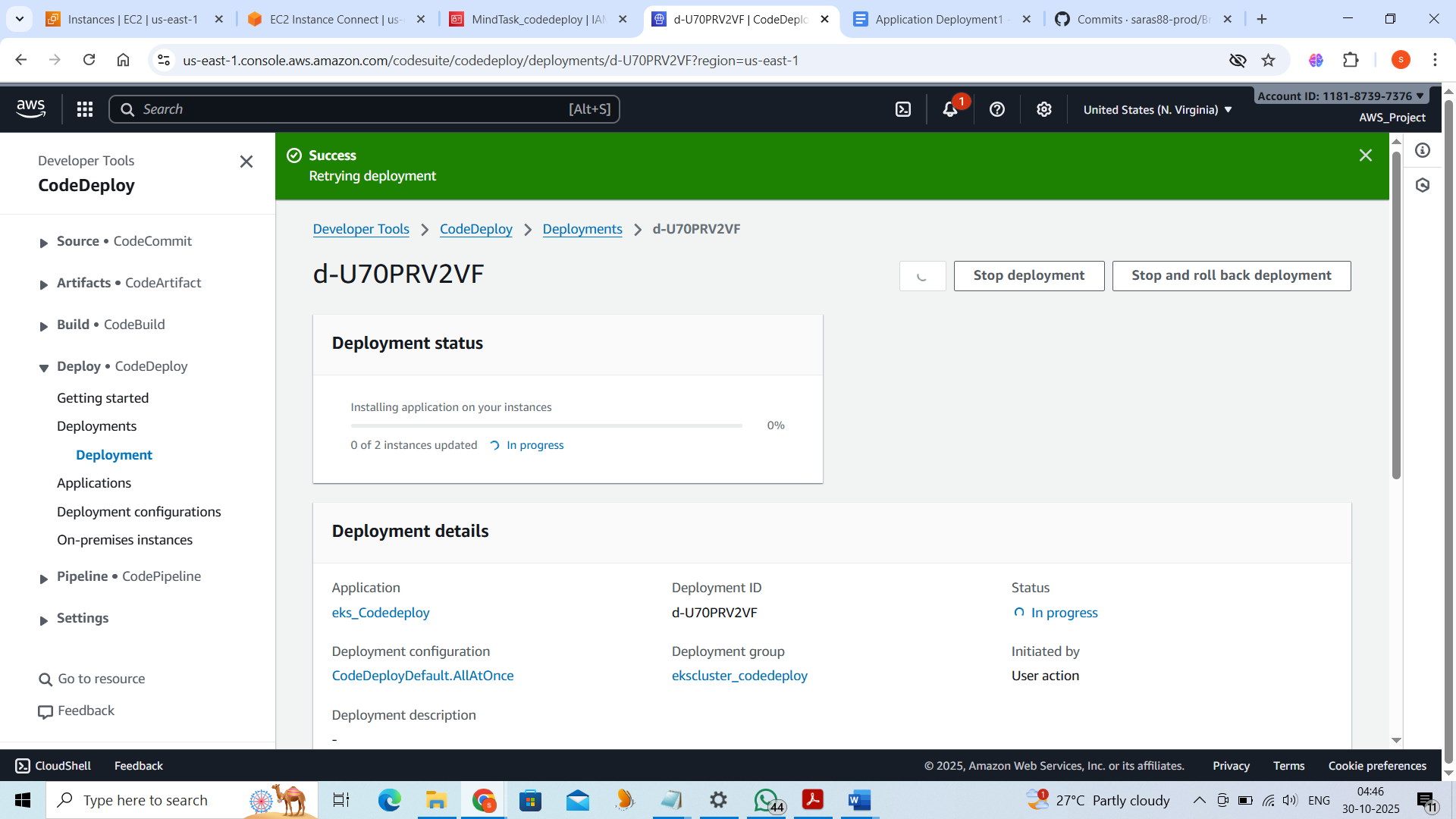






Docker image Latest:

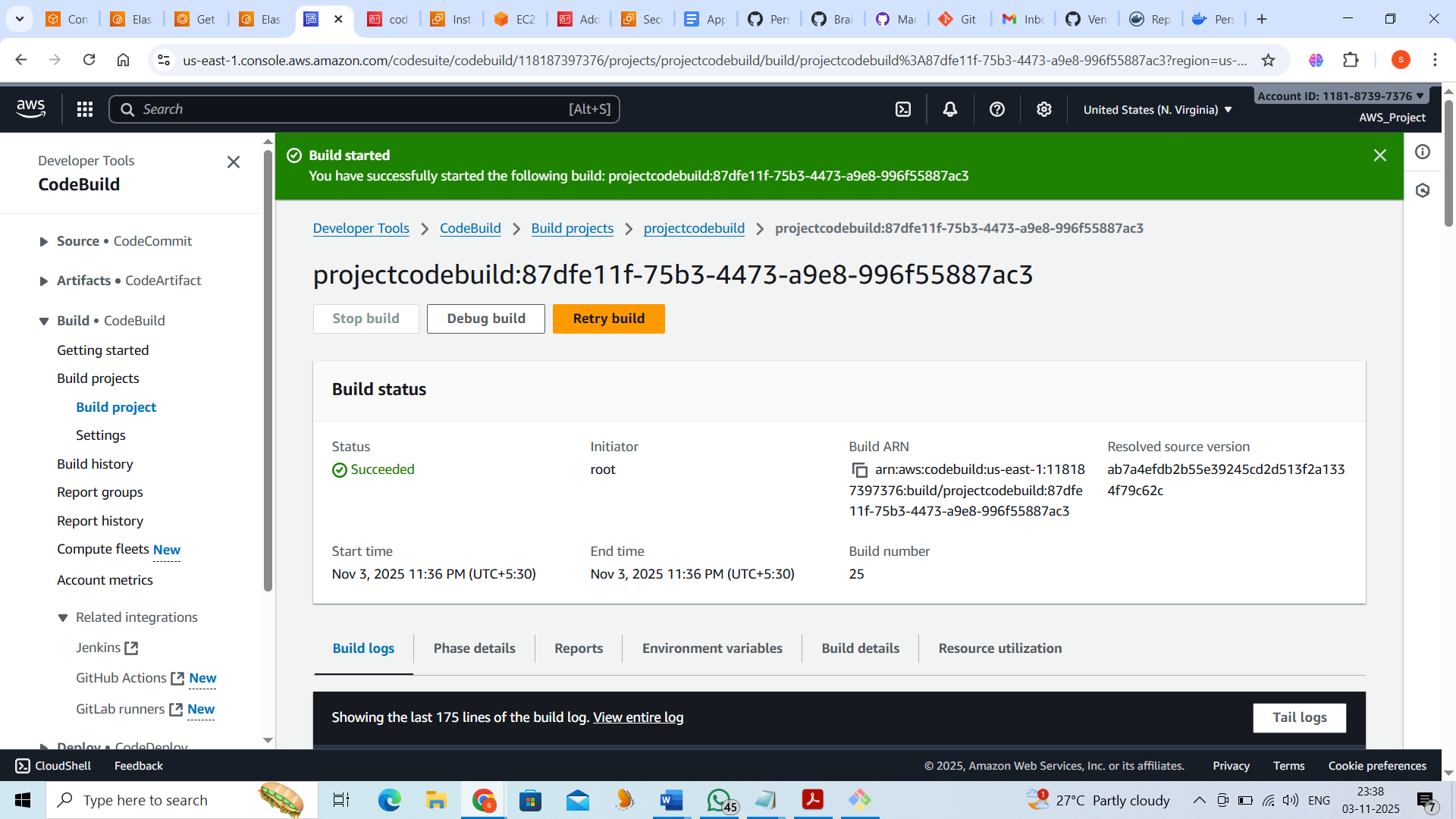


Deploy the kubectl pods/application through the code deploy:  
  


CodeBuild:

Logfile:

|  |
| --- |
| [Container] 2025/11/03 18:06:26.329945 Running on CodeBuild On-demand |
| [Container] 2025/11/03 18:06:26.329957 Waiting for agent ping |
| [Container] 2025/11/03 18:06:26.431105 Waiting for DOWNLOAD\_SOURCE |
| [Container] 2025/11/03 18:06:29.555842 Phase is DOWNLOAD\_SOURCE |
| [Container] 2025/11/03 18:06:29.557689 CODEBUILD\_SRC\_DIR=/codebuild/output/src2094498258/src/github.com/saras88-prod/BrainTask |
| [Container] 2025/11/03 18:06:29.558336 YAML location is /codebuild/readonly/buildspec.yml |
| [Container] 2025/11/03 18:06:29.560964 Setting HTTP client timeout to higher timeout for Github and GitHub Enterprise sources |
| [Container] 2025/11/03 18:06:29.561096 Processing environment variables |
| [Container] 2025/11/03 18:06:29.770573 No runtime version selected in buildspec. |
| [Container] 2025/11/03 18:06:29.790025 Moving to directory /codebuild/output/src2094498258/src/github.com/saras88-prod/BrainTask |
| [Container] 2025/11/03 18:06:29.790048 Cache is not defined in the buildspec |
| [Container] 2025/11/03 18:06:29.828773 Skip cache due to: no paths specified to be cached |
| [Container] 2025/11/03 18:06:29.829098 Registering with agent |
| [Container] 2025/11/03 18:06:29.864669 Phases found in YAML: 3 |
| [Container] 2025/11/03 18:06:29.864686 BUILD: 3 commands |
| [Container] 2025/11/03 18:06:29.864691 POST\_BUILD: 3 commands |
| [Container] 2025/11/03 18:06:29.864694 PRE\_BUILD: 9 commands |
| [Container] 2025/11/03 18:06:29.865053 Phase complete: DOWNLOAD\_SOURCE State: SUCCEEDED |
| [Container] 2025/11/03 18:06:29.865072 Phase context status code: Message: |
| [Container] 2025/11/03 18:06:29.980840 Entering phase INSTALL |
| [Container] 2025/11/03 18:06:30.017311 Phase complete: INSTALL State: SUCCEEDED |
| [Container] 2025/11/03 18:06:30.017410 Phase context status code: Message: |
| [Container] 2025/11/03 18:06:30.066062 Entering phase PRE\_BUILD |
| [Container] 2025/11/03 18:06:30.106145 Running command echo Logging in to Amazon ECR... |
| Logging in to Amazon ECR... |
|  |
| [Container] 2025/11/03 18:06:30.110053 Running command aws --version |
| aws-cli/2.31.1 Python/3.13.7 Linux/4.14.355-280.684.amzn2.x86\_64 exec-env/AWS\_ECS\_EC2 exe/x86\_64.ubuntu.22 |
|  |
| [Container] 2025/11/03 18:06:44.454006 Running command echo Logging in to Docker Hub... |
| Logging in to Docker Hub... |
|  |
| [Container] 2025/11/03 18:06:44.457794 Running command echo $DOCKERHUB\_PASSWORD | docker login -u $DOCKERHUB\_USERNAME --password-stdin |
| WARNING! Your password will be stored unencrypted in /root/.docker/config.json. |
| Configure a credential helper to remove this warning. See |
| https://docs.docker.com/engine/reference/commandline/login/#credential-stores |
|  |
| Login Succeeded |
|  |
| [Container] 2025/11/03 18:06:44.512072 Running command aws ecr get-login-password --region $AWS\_DEFAULT\_REGION | docker login --username AWS --password-stdin 118187397376.dkr.ecr.$AWS\_DEFAULT\_REGION.amazonaws.com |
| WARNING! Your password will be stored unencrypted in /root/.docker/config.json. |
| Configure a credential helper to remove this warning. See |
| https://docs.docker.com/engine/reference/commandline/login/#credential-stores |
|  |
| Login Succeeded |
|  |
| [Container] 2025/11/03 18:06:45.501746 Running command echo Build started on `date` |
| Build started on Mon Nov 3 18:06:45 UTC 2025 |
|  |
| [Container] 2025/11/03 18:06:45.506197 Running command echo Cloning GitHub repository... |
| Cloning GitHub repository... |
|  |
| [Container] 2025/11/03 18:06:45.509590 Running command git clone https://github.com/saras88-prod/BrainTask.git app |
| Cloning into 'app'... |
|  |
| [Container] 2025/11/03 18:06:45.663962 Running command cd app |
|  |
| [Container] 2025/11/03 18:06:45.667431 Phase complete: PRE\_BUILD State: SUCCEEDED |
| [Container] 2025/11/03 18:06:45.667444 Phase context status code: Message: |
| [Container] 2025/11/03 18:06:45.709070 Entering phase BUILD |
| [Container] 2025/11/03 18:06:45.710323 Running command echo Building the Docker image... |
| Building the Docker image... |
|  |
| [Container] 2025/11/03 18:06:45.714163 Running command docker build -t $ECR\_REPO\_NAME:$IMAGE\_TAG . |
| #0 building with "default" instance using docker driver |
|  |
| #1 [internal] load build definition from dockerfile |
| #1 transferring dockerfile: 408B done |
| #1 DONE 0.0s |
|  |
| #2 [auth] library/nginx:pull token for registry-1.docker.io |
| #2 DONE 0.0s |
|  |
| #3 [internal] load metadata for docker.io/library/nginx:alpine |
| #3 DONE 0.4s |
|  |
| #4 [internal] load .dockerignore |
| #4 transferring context: 2B done |
| #4 DONE 0.0s |
|  |
| #5 [internal] load build context |
| #5 transferring context: 317.98kB done |
| #5 DONE 0.0s |
|  |
| #6 [1/2] FROM docker.io/library/nginx:alpine@sha256:b3c656d55d7ad751196f21b7fd2e8d4da9cb430e32f646adcf92441b72f82b14 |
| #6 resolve docker.io/library/nginx:alpine@sha256:b3c656d55d7ad751196f21b7fd2e8d4da9cb430e32f646adcf92441b72f82b14 0.0s done |
| #6 sha256:b3c656d55d7ad751196f21b7fd2e8d4da9cb430e32f646adcf92441b72f82b14 10.33kB / 10.33kB done |
| #6 sha256:667473807103639a0aca5b49534a216d2b64f0fb868aaa801f023da0cdd781c7 2.50kB / 2.50kB done |
| #6 sha256:d4918ca78576a537caa7b0c043051c8efc1796de33fee8724ee0fff4a1cabed9 10.96kB / 10.96kB done |
| #6 sha256:2d35ebdb57d9971fea0cac1582aa78935adf8058b2cc32db163c98822e5dfa1b 3.80MB / 3.80MB 0.1s done |
| #6 sha256:8f6a6833e95d43ac524f1f9c5e7c1316c1f3b8e7ae5ba3db4e54b0c5b910e80a 1.05MB / 1.84MB 0.1s |
| #6 sha256:194fa24e147df0010e146240d3b4bd25d04180c523dc717e4645b269991483e3 628B / 628B 0.1s done |
| #6 sha256:3eaba6cd10a374d9ed629c26d76a5258e20ddfa09fcef511c98aa620dcf3fae4 0B / 955B 0.1s |
| #6 sha256:df413d6ebdc834bccf63178455d406c4d25e2c2d38d2c1ab79ee5494b18e5624 0B / 403B 0.1s |
| #6 sha256:8f6a6833e95d43ac524f1f9c5e7c1316c1f3b8e7ae5ba3db4e54b0c5b910e80a 1.84MB / 1.84MB 0.1s done |
| #6 sha256:3eaba6cd10a374d9ed629c26d76a5258e20ddfa09fcef511c98aa620dcf3fae4 955B / 955B 0.1s done |
| #6 sha256:df413d6ebdc834bccf63178455d406c4d25e2c2d38d2c1ab79ee5494b18e5624 403B / 403B 0.2s done |
| #6 extracting sha256:2d35ebdb57d9971fea0cac1582aa78935adf8058b2cc32db163c98822e5dfa1b 0.1s |
| #6 sha256:d9a55dab5954588333096b28b351999099bea5eb3c68c10e99f175b12c97198d 1.21kB / 1.21kB 0.2s done |
| #6 sha256:ff8a36d5502a57c3fc8eeff48e578ab433a03b1dd528992ba0d966ddf853309a 1.40kB / 1.40kB 0.2s done |
| #6 sha256:bdabb0d442710d667f4fd871b5fd215cc2a430a95b192bc508bf945b8e60999b 0B / 16.97MB 0.2s |
| #6 extracting sha256:2d35ebdb57d9971fea0cac1582aa78935adf8058b2cc32db163c98822e5dfa1b 0.2s done |
| #6 sha256:bdabb0d442710d667f4fd871b5fd215cc2a430a95b192bc508bf945b8e60999b 16.97MB / 16.97MB 0.4s done |
| #6 extracting sha256:8f6a6833e95d43ac524f1f9c5e7c1316c1f3b8e7ae5ba3db4e54b0c5b910e80a |
| #6 extracting sha256:8f6a6833e95d43ac524f1f9c5e7c1316c1f3b8e7ae5ba3db4e54b0c5b910e80a 0.1s done |
| #6 extracting sha256:194fa24e147df0010e146240d3b4bd25d04180c523dc717e4645b269991483e3 |
| #6 extracting sha256:194fa24e147df0010e146240d3b4bd25d04180c523dc717e4645b269991483e3 done |
| #6 extracting sha256:3eaba6cd10a374d9ed629c26d76a5258e20ddfa09fcef511c98aa620dcf3fae4 done |
| #6 extracting sha256:df413d6ebdc834bccf63178455d406c4d25e2c2d38d2c1ab79ee5494b18e5624 done |
| #6 extracting sha256:d9a55dab5954588333096b28b351999099bea5eb3c68c10e99f175b12c97198d |
| #6 extracting sha256:d9a55dab5954588333096b28b351999099bea5eb3c68c10e99f175b12c97198d done |
| #6 extracting sha256:ff8a36d5502a57c3fc8eeff48e578ab433a03b1dd528992ba0d966ddf853309a done |
| #6 extracting sha256:bdabb0d442710d667f4fd871b5fd215cc2a430a95b192bc508bf945b8e60999b |
| #6 extracting sha256:bdabb0d442710d667f4fd871b5fd215cc2a430a95b192bc508bf945b8e60999b 0.4s done |
| #6 DONE 1.6s |
|  |
| #7 [2/2] COPY dist/ /usr/share/nginx/html |
| #7 DONE 0.4s |
|  |
| #8 exporting to image |
| #8 exporting layers 0.0s done |
| #8 writing image sha256:02d9e94d903c131f1eb525e203d1386503ccc2e2f64ec38577b8040dc41c5b18 done |
| #8 naming to docker.io/library/brain-tasks-app:latest done |
| #8 DONE 0.0s |
|  |
| [Container] 2025/11/03 18:06:48.417831 Running command docker tag $ECR\_REPO\_NAME:$IMAGE\_TAG 118187397376.dkr.ecr.$AWS\_DEFAULT\_REGION.amazonaws.com/$ECR\_REPO\_NAME:$IMAGE\_TAG |
|  |
| [Container] 2025/11/03 18:06:48.439135 Phase complete: BUILD State: SUCCEEDED |
| [Container] 2025/11/03 18:06:48.439148 Phase context status code: Message: |
| [Container] 2025/11/03 18:06:48.477451 Entering phase POST\_BUILD |
| [Container] 2025/11/03 18:06:48.478690 Running command echo Pushing the Docker image to ECR... |
| Pushing the Docker image to ECR... |
|  |
| [Container] 2025/11/03 18:06:48.484666 Running command docker push 118187397376.dkr.ecr.$AWS\_DEFAULT\_REGION.amazonaws.com/$ECR\_REPO\_NAME:$IMAGE\_TAG |
| The push refers to repository [118187397376.dkr.ecr.us-east-1.amazonaws.com/brain-tasks-app] |
| 2c120b3e1ecc: Preparing |
| 25906c27b84d: Preparing |
| 99ea4bde418d: Preparing |
| 3297b9628ff3: Preparing |
| b74d92be8225: Preparing |
| 2c79d5d895bb: Preparing |
| 2660a7d4b906: Preparing |
| 50b58ca2a3f5: Preparing |
| 256f393e029f: Preparing |
| 2660a7d4b906: Waiting |
| 50b58ca2a3f5: Waiting |
| 256f393e029f: Waiting |
| 2c79d5d895bb: Waiting |
| b74d92be8225: Pushed |
| 3297b9628ff3: Pushed |
| 99ea4bde418d: Pushed |
| 2c120b3e1ecc: Pushed |
| 2c79d5d895bb: Pushed |
| 2660a7d4b906: Pushed |
| 50b58ca2a3f5: Pushed |
| 256f393e029f: Pushed |
| 25906c27b84d: Pushed |
| latest: digest: sha256:f0dd23b33e32248e187102bd4ea678be884d94332c25ff1c900fed412855d6e8 size: 2199 |
|  |
| [Container] 2025/11/03 18:06:52.458677 Running command echo Build completed on `date` |
| Build completed on Mon Nov 3 18:06:52 UTC 2025 |
|  |
| [Container] 2025/11/03 18:06:52.467628 Phase complete: POST\_BUILD State: SUCCEEDED |
| [Container] 2025/11/03 18:06:52.467646 Phase context status code: Message: |
| [Container] 2025/11/03 18:06:52.512830 Set report auto-discover timeout to 5 seconds |
| [Container] 2025/11/03 18:06:52.512873 Expanding base directory path: . |
| [Container] 2025/11/03 18:06:52.514461 Assembling file list |
| [Container] 2025/11/03 18:06:52.514473 Expanding . |
| [Container] 2025/11/03 18:06:52.516106 Expanding file paths for base directory . |
| [Container] 2025/11/03 18:06:52.516116 Assembling file list |
| [Container] 2025/11/03 18:06:52.516120 Expanding \*\*/\* |
| [Container] 2025/11/03 18:06:52.518446 No matching auto-discover report paths found |
| [Container] 2025/11/03 18:06:52.518463 Report auto-discover file discovery took 0.005633 seconds |
| [Container] 2025/11/03 18:06:52.518473 Phase complete: UPLOAD\_ARTIFACTS State: SUCCEEDED |
| [Container] 2025/11/03 18:06:52.518582 Phase context status code: Message: |
|  |



Buildspec.yaml:

version: 0.2

env:

variables:

# Name of your ECR repository

ECR\_REPO\_NAME: brain-tasks-app

# AWS region

AWS\_DEFAULT\_REGION: us-east-1

# Docker image tag

IMAGE\_TAG: latest

DOCKERHUB\_PASSWORD: dckr\_pat\_s45MuxORVSJONytj0wvAnREY9ys

DOCKERHUB\_USERNAME: saraswathi6

phases:

pre\_build:

commands:

- echo Logging in to Amazon ECR...

- aws --version

- echo Logging in to Docker Hub...

- echo $DOCKERHUB\_PASSWORD | docker login -u $DOCKERHUB\_USERNAME --password-stdin

- aws ecr get-login-password --region $AWS\_DEFAULT\_REGION | docker login --username AWS --password-stdin 118187397376.dkr.ecr.$AWS\_DEFAULT\_REGION.amazonaws.com

- echo Build started on `date`

- echo Cloning GitHub repository...

- git clone https://github.com/saras88-prod/BrainTask.git app

- cd app

build:

commands:

- echo Building the Docker image...

- docker build -t $ECR\_REPO\_NAME:$IMAGE\_TAG .

- docker tag $ECR\_REPO\_NAME:$IMAGE\_TAG 118187397376.dkr.ecr.$AWS\_DEFAULT\_REGION.amazonaws.com/$ECR\_REPO\_NAME:$IMAGE\_TAG

post\_build:

commands:

- echo Pushing the Docker image to ECR...

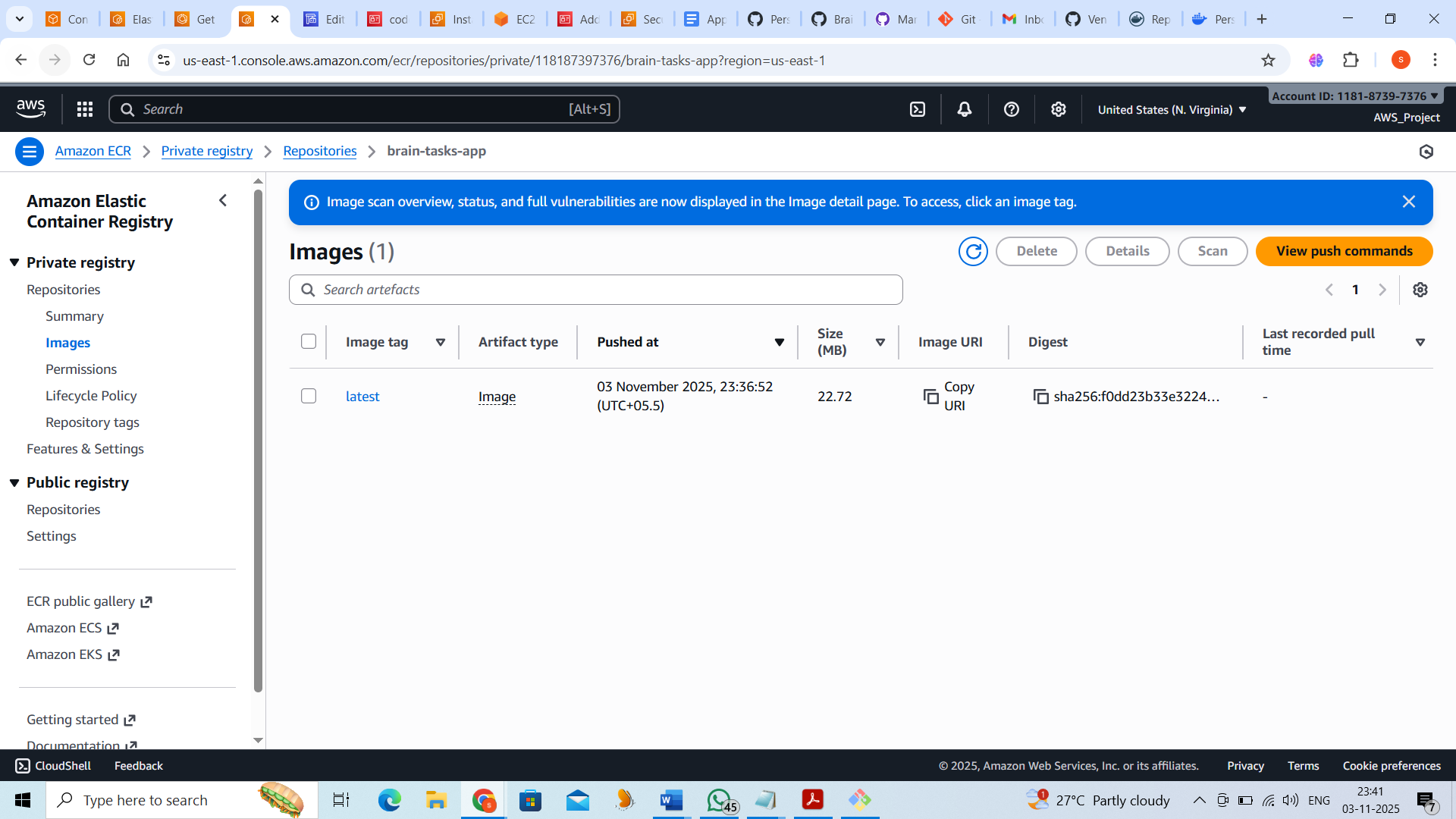
- docker push 118187397376.dkr.ecr.$AWS\_DEFAULT\_REGION.amazonaws.com/$ECR\_REPO\_NAME:$IMAGE\_TAG

- echo Build completed on `date`

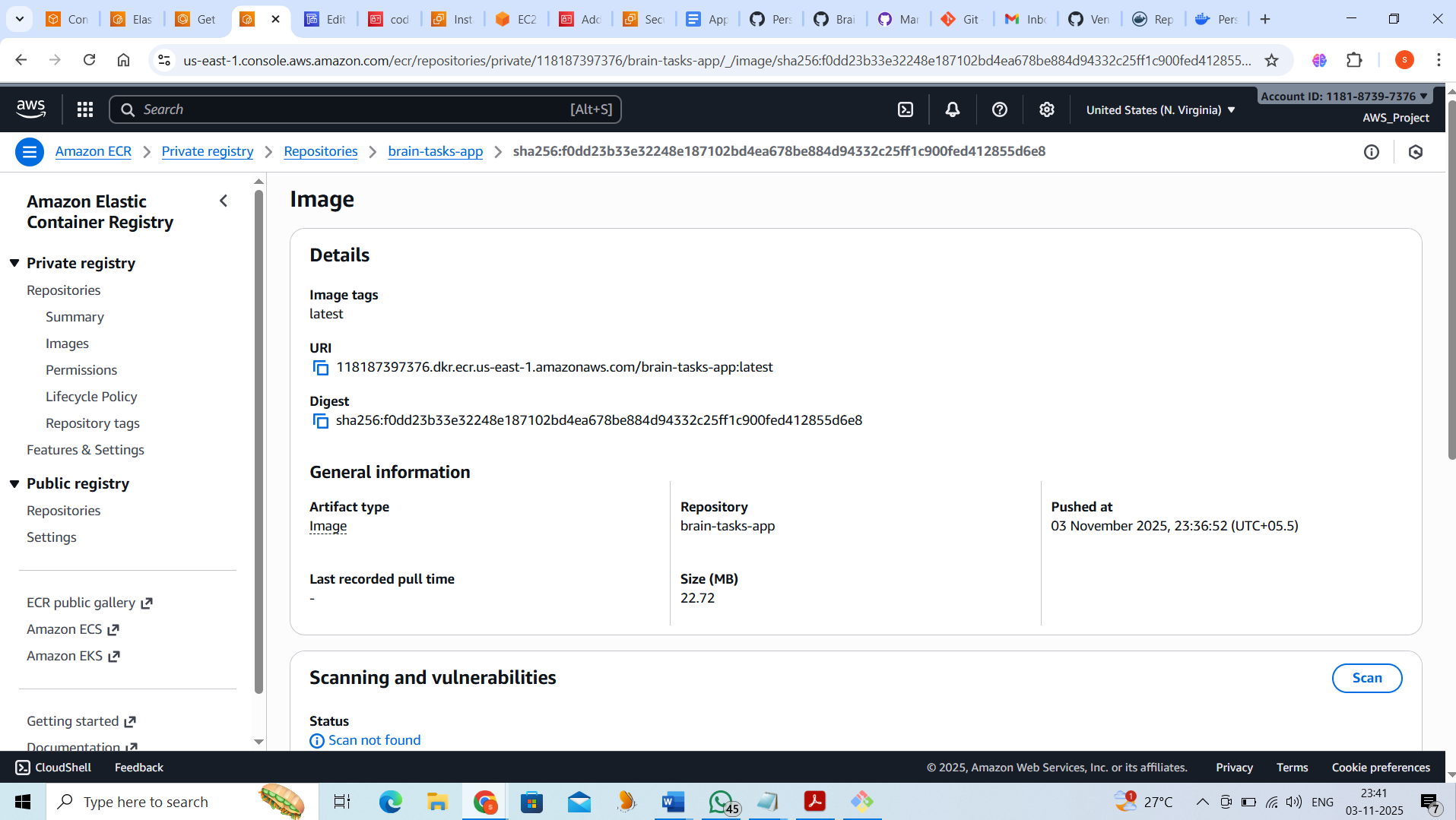
artifacts:

files: []

ECR :

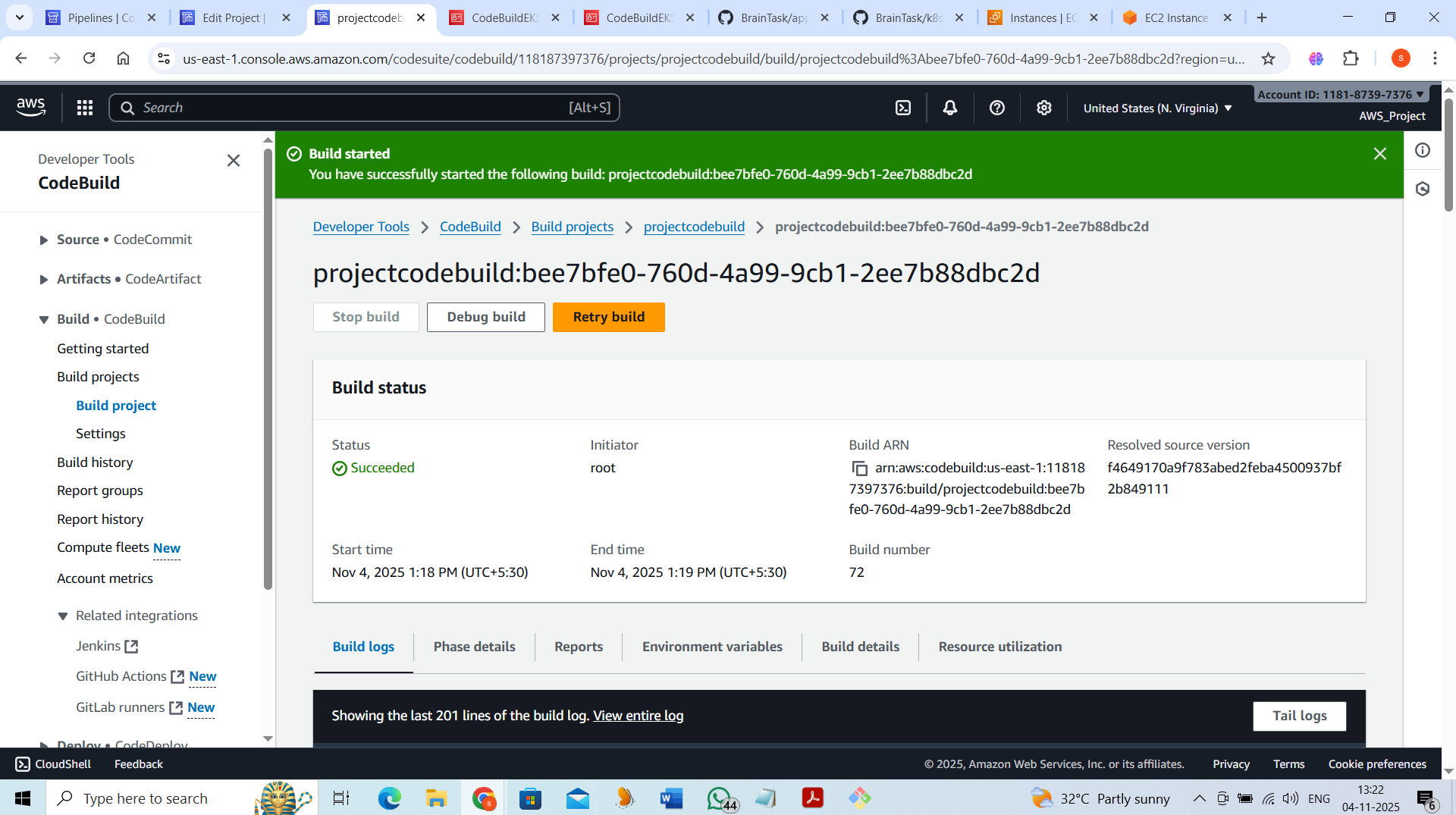


118187397376.dkr.ecr.us-east-1.amazonaws.com/brain-tasks-app:latest



Pod Deployment:

|  |
| --- |
| [Container] 2025/11/04 07:48:52.262943 Running on CodeBuild On-demand |
| [Container] 2025/11/04 07:48:52.262954 Waiting for agent ping |
| [Container] 2025/11/04 07:48:52.365656 Waiting for DOWNLOAD\_SOURCE |
| [Container] 2025/11/04 07:48:55.189053 Phase is DOWNLOAD\_SOURCE |
| [Container] 2025/11/04 07:48:55.191297 CODEBUILD\_SRC\_DIR=/codebuild/output/src3088391916/src/github.com/saras88-prod/BrainTask |
| [Container] 2025/11/04 07:48:55.191943 YAML location is /codebuild/readonly/buildspec.yml |
| [Container] 2025/11/04 07:48:55.195204 Setting HTTP client timeout to higher timeout for Github and GitHub Enterprise sources |
| [Container] 2025/11/04 07:48:55.195341 Processing environment variables |
| [Container] 2025/11/04 07:48:55.372561 No runtime version selected in buildspec. |
| [Container] 2025/11/04 07:48:55.392464 Moving to directory /codebuild/output/src3088391916/src/github.com/saras88-prod/BrainTask |
| [Container] 2025/11/04 07:48:55.392583 Cache is not defined in the buildspec |
| [Container] 2025/11/04 07:48:55.428960 Skip cache due to: no paths specified to be cached |
| [Container] 2025/11/04 07:48:55.429266 Registering with agent |
| [Container] 2025/11/04 07:48:55.462852 Phases found in YAML: 3 |
| [Container] 2025/11/04 07:48:55.462872 BUILD: 3 commands |
| [Container] 2025/11/04 07:48:55.462878 POST\_BUILD: 12 commands |
| [Container] 2025/11/04 07:48:55.462883 PRE\_BUILD: 9 commands |
| [Container] 2025/11/04 07:48:55.463210 Phase complete: DOWNLOAD\_SOURCE State: SUCCEEDED |
| [Container] 2025/11/04 07:48:55.463225 Phase context status code: Message: |
| [Container] 2025/11/04 07:48:55.570283 Entering phase INSTALL |
| [Container] 2025/11/04 07:48:55.605859 Phase complete: INSTALL State: SUCCEEDED |
| [Container] 2025/11/04 07:48:55.605887 Phase context status code: Message: |
| [Container] 2025/11/04 07:48:55.636229 Entering phase PRE\_BUILD |
| [Container] 2025/11/04 07:48:55.668773 Running command echo Logging in to Amazon ECR... |
| Logging in to Amazon ECR... |
|  |
| [Container] 2025/11/04 07:48:55.672805 Running command aws --version |
| aws-cli/2.31.1 Python/3.13.7 Linux/4.14.355-280.684.amzn2.x86\_64 exec-env/AWS\_ECS\_EC2 exe/x86\_64.ubuntu.22 |
|  |
| [Container] 2025/11/04 07:49:07.415874 Running command echo Logging in to Docker Hub... |
| Logging in to Docker Hub... |
|  |
| [Container] 2025/11/04 07:49:07.419760 Running command echo $DOCKERHUB\_PASSWORD | docker login -u $DOCKERHUB\_USERNAME --password-stdin |
| WARNING! Your password will be stored unencrypted in /root/.docker/config.json. |
| Configure a credential helper to remove this warning. See |
| https://docs.docker.com/engine/reference/commandline/login/#credential-stores |
|  |
| Login Succeeded |
|  |
| [Container] 2025/11/04 07:49:07.479331 Running command aws ecr get-login-password --region $AWS\_DEFAULT\_REGION | docker login --username AWS --password-stdin 118187397376.dkr.ecr.$AWS\_DEFAULT\_REGION.amazonaws.com |
| WARNING! Your password will be stored unencrypted in /root/.docker/config.json. |
| Configure a credential helper to remove this warning. See |
| https://docs.docker.com/engine/reference/commandline/login/#credential-stores |
|  |
| Login Succeeded |
|  |
| [Container] 2025/11/04 07:49:08.605023 Running command echo Build started on `date` |
| Build started on Tue Nov 4 07:49:08 UTC 2025 |
|  |
| [Container] 2025/11/04 07:49:08.614220 Running command echo Cloning GitHub repository... |
| Cloning GitHub repository... |
|  |
| [Container] 2025/11/04 07:49:08.618763 Running command git clone https://github.com/saras88-prod/BrainTask.git app |
| Cloning into 'app'... |
|  |
| [Container] 2025/11/04 07:49:08.787022 Running command cd app |
|  |
| [Container] 2025/11/04 07:49:08.790988 Phase complete: PRE\_BUILD State: SUCCEEDED |
| [Container] 2025/11/04 07:49:08.791006 Phase context status code: Message: |
| [Container] 2025/11/04 07:49:08.834093 Entering phase BUILD |
| [Container] 2025/11/04 07:49:08.835525 Running command echo Building the Docker image... |
| Building the Docker image... |
|  |
| [Container] 2025/11/04 07:49:08.839179 Running command docker build -t $ECR\_REPO\_NAME:$IMAGE\_TAG . |
| #0 building with "default" instance using docker driver |
|  |
| #1 [internal] load build definition from dockerfile |
| #1 transferring dockerfile: 408B done |
| #1 DONE 0.0s |
|  |
| #2 [auth] library/nginx:pull token for registry-1.docker.io |
| #2 DONE 0.0s |
|  |
| #3 [internal] load metadata for docker.io/library/nginx:alpine |
| #3 DONE 0.4s |
|  |
| #4 [internal] load .dockerignore |
| #4 transferring context: 2B done |
| #4 DONE 0.0s |
|  |
| #5 [internal] load build context |
| #5 transferring context: 317.98kB done |
| #5 DONE 0.0s |
|  |
| #6 [1/2] FROM docker.io/library/nginx:alpine@sha256:b3c656d55d7ad751196f21b7fd2e8d4da9cb430e32f646adcf92441b72f82b14 |
| #6 resolve docker.io/library/nginx:alpine@sha256:b3c656d55d7ad751196f21b7fd2e8d4da9cb430e32f646adcf92441b72f82b14 0.0s done |
| #6 sha256:3eaba6cd10a374d9ed629c26d76a5258e20ddfa09fcef511c98aa620dcf3fae4 0B / 955B 0.1s |
| #6 sha256:667473807103639a0aca5b49534a216d2b64f0fb868aaa801f023da0cdd781c7 2.50kB / 2.50kB done |
| #6 sha256:8f6a6833e95d43ac524f1f9c5e7c1316c1f3b8e7ae5ba3db4e54b0c5b910e80a 1.84MB / 1.84MB 0.1s done |
| #6 sha256:194fa24e147df0010e146240d3b4bd25d04180c523dc717e4645b269991483e3 628B / 628B 0.1s |
| #6 sha256:df413d6ebdc834bccf63178455d406c4d25e2c2d38d2c1ab79ee5494b18e5624 0B / 403B 0.1s |
| #6 sha256:b3c656d55d7ad751196f21b7fd2e8d4da9cb430e32f646adcf92441b72f82b14 10.33kB / 10.33kB done |
| #6 sha256:d4918ca78576a537caa7b0c043051c8efc1796de33fee8724ee0fff4a1cabed9 10.96kB / 10.96kB done |
| #6 sha256:2d35ebdb57d9971fea0cac1582aa78935adf8058b2cc32db163c98822e5dfa1b 3.80MB / 3.80MB 0.1s done |
| #6 sha256:3eaba6cd10a374d9ed629c26d76a5258e20ddfa09fcef511c98aa620dcf3fae4 955B / 955B 0.1s done |
| #6 sha256:194fa24e147df0010e146240d3b4bd25d04180c523dc717e4645b269991483e3 628B / 628B 0.1s done |
| #6 sha256:df413d6ebdc834bccf63178455d406c4d25e2c2d38d2c1ab79ee5494b18e5624 403B / 403B 0.1s done |
| #6 extracting sha256:2d35ebdb57d9971fea0cac1582aa78935adf8058b2cc32db163c98822e5dfa1b 0.1s |
| #6 sha256:ff8a36d5502a57c3fc8eeff48e578ab433a03b1dd528992ba0d966ddf853309a 1.40kB / 1.40kB 0.2s done |
| #6 sha256:d9a55dab5954588333096b28b351999099bea5eb3c68c10e99f175b12c97198d 1.21kB / 1.21kB 0.1s done |
| #6 sha256:bdabb0d442710d667f4fd871b5fd215cc2a430a95b192bc508bf945b8e60999b 1.05MB / 16.97MB 0.2s |
| #6 extracting sha256:2d35ebdb57d9971fea0cac1582aa78935adf8058b2cc32db163c98822e5dfa1b 0.2s done |
| #6 sha256:bdabb0d442710d667f4fd871b5fd215cc2a430a95b192bc508bf945b8e60999b 16.97MB / 16.97MB 0.4s done |
| #6 extracting sha256:8f6a6833e95d43ac524f1f9c5e7c1316c1f3b8e7ae5ba3db4e54b0c5b910e80a |
| #6 extracting sha256:8f6a6833e95d43ac524f1f9c5e7c1316c1f3b8e7ae5ba3db4e54b0c5b910e80a 0.1s done |
| #6 extracting sha256:194fa24e147df0010e146240d3b4bd25d04180c523dc717e4645b269991483e3 |
| #6 extracting sha256:194fa24e147df0010e146240d3b4bd25d04180c523dc717e4645b269991483e3 done |
| #6 extracting sha256:3eaba6cd10a374d9ed629c26d76a5258e20ddfa09fcef511c98aa620dcf3fae4 done |
| #6 extracting sha256:df413d6ebdc834bccf63178455d406c4d25e2c2d38d2c1ab79ee5494b18e5624 done |
| #6 extracting sha256:d9a55dab5954588333096b28b351999099bea5eb3c68c10e99f175b12c97198d |
| #6 extracting sha256:d9a55dab5954588333096b28b351999099bea5eb3c68c10e99f175b12c97198d done |
| #6 extracting sha256:ff8a36d5502a57c3fc8eeff48e578ab433a03b1dd528992ba0d966ddf853309a done |
| #6 extracting sha256:bdabb0d442710d667f4fd871b5fd215cc2a430a95b192bc508bf945b8e60999b |
| #6 extracting sha256:bdabb0d442710d667f4fd871b5fd215cc2a430a95b192bc508bf945b8e60999b 0.4s done |
| #6 DONE 1.7s |
|  |
| #7 [2/2] COPY dist/ /usr/share/nginx/html |
| #7 DONE 0.4s |
|  |
| #8 exporting to image |
| #8 exporting layers 0.0s done |
| #8 writing image sha256:c9d10026a33e9b017c9979053108c803f1262a856bcfb5e309d9259279809320 done |
| #8 naming to docker.io/library/brain-tasks-app:latest done |
| #8 DONE 0.0s |
|  |
| [Container] 2025/11/04 07:49:11.581093 Running command docker tag $ECR\_REPO\_NAME:$IMAGE\_TAG 118187397376.dkr.ecr.$AWS\_DEFAULT\_REGION.amazonaws.com/$ECR\_REPO\_NAME:$IMAGE\_TAG |
|  |
| [Container] 2025/11/04 07:49:11.613981 Phase complete: BUILD State: SUCCEEDED |
| [Container] 2025/11/04 07:49:11.614003 Phase context status code: Message: |
| [Container] 2025/11/04 07:49:11.652074 Entering phase POST\_BUILD |
| [Container] 2025/11/04 07:49:11.654116 Running command echo Pushing the Docker image to ECR... |
| Pushing the Docker image to ECR... |
|  |
| [Container] 2025/11/04 07:49:11.669260 Running command docker push 118187397376.dkr.ecr.$AWS\_DEFAULT\_REGION.amazonaws.com/$ECR\_REPO\_NAME:$IMAGE\_TAG |
| The push refers to repository [118187397376.dkr.ecr.us-east-1.amazonaws.com/brain-tasks-app] |
| 1c4a6313bfc7: Preparing |
| 25906c27b84d: Preparing |
| 99ea4bde418d: Preparing |
| 3297b9628ff3: Preparing |
| b74d92be8225: Preparing |
| 2c79d5d895bb: Preparing |
| 2660a7d4b906: Preparing |
| 50b58ca2a3f5: Preparing |
| 256f393e029f: Preparing |
| 2660a7d4b906: Waiting |
| 50b58ca2a3f5: Waiting |
| 256f393e029f: Waiting |
| 2c79d5d895bb: Waiting |
| b74d92be8225: Layer already exists |
| 25906c27b84d: Layer already exists |
| 99ea4bde418d: Layer already exists |
| 3297b9628ff3: Layer already exists |
| 2c79d5d895bb: Layer already exists |
| 256f393e029f: Layer already exists |
| 50b58ca2a3f5: Layer already exists |
| 2660a7d4b906: Layer already exists |
| 1c4a6313bfc7: Pushed |
| latest: digest: sha256:729dd9b4b263944f33f6f6caf622d9163ba521cf40ba7b802f1c7043d9ae8816 size: 2199 |
|  |
| [Container] 2025/11/04 07:49:13.254567 Running command echo Build completed on `date` |
| Build completed on Tue Nov 4 07:49:13 UTC 2025 |
|  |
| [Container] 2025/11/04 07:49:13.261196 Running command echo Updating kubeconfig for EKS cluster... |
| Updating kubeconfig for EKS cluster... |
|  |
| [Container] 2025/11/04 07:49:13.266751 Running command aws eks update-kubeconfig --region us-east-1 --name brain-tasks-clusters |
| Added new context arn:aws:eks:us-east-1:118187397376:cluster/brain-tasks-clusters to /root/.kube/config |
|  |
| [Container] 2025/11/04 07:49:14.287873 Running command echo Applying Kubernetes deployment... |
| Applying Kubernetes deployment... |
|  |
| [Container] 2025/11/04 07:49:14.294085 Running command kubectl apply -f k8s/deployment.yaml -n default |
| deployment.apps/brain-tasks-deployment unchanged |
|  |
| [Container] 2025/11/04 07:49:18.824291 Running command echo Updating Kubernetes deployment with new image... |
| Updating Kubernetes deployment with new image... |
|  |
| [Container] 2025/11/04 07:49:18.830420 Running command kubectl set image deployment/brain-tasks-deployment brain-tasks-container=118187397376.dkr.ecr.$AWS\_DEFAULT\_REGION.amazonaws.com/$ECR\_REPO\_NAME:$IMAGE\_TAG -n default |
|  |
| [Container] 2025/11/04 07:49:19.704202 Running command echo Applying service manifest |
| Applying service manifest |
|  |
| [Container] 2025/11/04 07:49:19.710306 Running command kubectl apply -f k8s/service.yaml -n default |
| service/brain-tasks-service created |
|  |
| [Container] 2025/11/04 07:49:20.613730 Running command echo Deployment to EKS completed successfully! |
| Deployment to EKS completed successfully! |
|  |
| [Container] 2025/11/04 07:49:20.620122 Phase complete: POST\_BUILD State: SUCCEEDED |
| [Container] 2025/11/04 07:49:20.620142 Phase context status code: Message: |
| [Container] 2025/11/04 07:49:20.661634 Set report auto-discover timeout to 5 seconds |
| [Container] 2025/11/04 07:49:20.661761 Expanding base directory path: . |
| [Container] 2025/11/04 07:49:20.663329 Assembling file list |
| [Container] 2025/11/04 07:49:20.663342 Expanding . |
| [Container] 2025/11/04 07:49:20.664950 Expanding file paths for base directory . |
| [Container] 2025/11/04 07:49:20.664962 Assembling file list |
| [Container] 2025/11/04 07:49:20.664965 Expanding \*\*/\* |
| [Container] 2025/11/04 07:49:20.667608 No matching auto-discover report paths found |
| [Container] 2025/11/04 07:49:20.667836 Report auto-discover file discovery took 0.006202 seconds |
| [Container] 2025/11/04 07:49:20.667852 Phase complete: UPLOAD\_ARTIFACTS State: SUCCEEDED |
| [Container] 2025/11/04 07:49:20.667858 Phase context status code: Message: |
|  |



version: 0.2

env:

variables:

# Name of your ECR repository

ECR\_REPO\_NAME: brain-tasks-app

# AWS region

AWS\_DEFAULT\_REGION: us-east-1

# Docker image tag

IMAGE\_TAG: latest

DOCKERHUB\_PASSWORD: dckr\_pat\_s45MuxORVSJONytj0wvAnREY9ys

DOCKERHUB\_USERNAME: saraswathi6

phases:

pre\_build:

commands:

- echo Logging in to Amazon ECR...

- aws --version

- echo Logging in to Docker Hub...

- echo $DOCKERHUB\_PASSWORD | docker login -u $DOCKERHUB\_USERNAME --password-stdin

- aws ecr get-login-password --region $AWS\_DEFAULT\_REGION | docker login --username AWS --password-stdin 118187397376.dkr.ecr.$AWS\_DEFAULT\_REGION.amazonaws.com

- echo Build started on `date`

- echo Cloning GitHub repository...

- git clone https://github.com/saras88-prod/BrainTask.git app

- cd app

build:

commands:

- echo Building the Docker image...

- docker build -t $ECR\_REPO\_NAME:$IMAGE\_TAG .

- docker tag $ECR\_REPO\_NAME:$IMAGE\_TAG 118187397376.dkr.ecr.$AWS\_DEFAULT\_REGION.amazonaws.com/$ECR\_REPO\_NAME:$IMAGE\_TAG

post\_build:

commands:

- echo Pushing the Docker image to ECR...

- docker push 118187397376.dkr.ecr.$AWS\_DEFAULT\_REGION.amazonaws.com/$ECR\_REPO\_NAME:$IMAGE\_TAG

- echo Build completed on `date`

- echo Updating kubeconfig for EKS cluster...

- aws eks update-kubeconfig --region us-east-1 --name brain-tasks-clusters

- echo Applying Kubernetes deployment...

- kubectl apply -f k8s/deployment.yaml -n default

- echo Updating Kubernetes deployment with new image...

- kubectl set image deployment/brain-tasks-deployment brain-tasks-container=118187397376.dkr.ecr.$AWS\_DEFAULT\_REGION.amazonaws.com/$ECR\_REPO\_NAME:$IMAGE\_TAG -n default

- echo Applying service manifest

- kubectl apply -f k8s/service.yaml -n default

- echo Deployment to EKS completed successfully!

artifacts:

files: []

appspec.yaml:

version: 0.0

Resources:

- TargetService:

Type: AWS::EKS::Service

Properties:

ClusterName: brain-tasks-clusters

Namespace: default

# No specific name needed here; handled in deploy.sh

Hooks:

BeforeInstall:

- location: scripts/deploy.sh

timeout: 300

runas: root

deployment.yaml:

apiVersion: apps/v1

kind: Deployment

metadata:

name: brain-tasks-deployment

namespace: default

spec:

replicas: 2

selector:

matchLabels:

app: brain-tasks

template:

metadata:

labels:

app: brain-tasks

spec:

containers:

- name: brain-tasks-container

image: 118187397376.dkr.ecr.us-east-1.amazonaws.com/brain-tasks-app:latest

ports:

- containerPort: 3000

Service.yaml:

apiVersion: v1

kind: Service

metadata:

name: brain-tasks-service

namespace: default

labels:

app: brain-tasks

spec:

selector:

app: brain-tasks

ports:

- protocol: TCP

port: 80

targetPort: 3000

type: LoadBalancer

deploy.sh:

#!/bin/bash

set -e

# Variables

NAMESPACE="default"

CLUSTER\_NAME="brain-tasks-clusters"

AWS\_REGION="us-east-1"

ECR\_REPO="118187397376.dkr.ecr.$AWS\_REGION.amazonaws.com/brain-tasks-app"

IMAGE\_TAG="latest"

echo "Logging in to EKS cluster..."

aws eks update-kubeconfig --region $AWS\_REGION --name $CLUSTER\_NAME

echo "Updating deployment image in Kubernetes..."

kubectl set image deployment/brain-tasks-deployment \

brain-tasks-container=$ECR\_REPO:$IMAGE\_TAG \

-n $NAMESPACE

echo "Applying Kubernetes manifests..."

kubectl apply -f k8s/deployment.yaml -n $NAMESPACE

kubectl apply -f k8s/service.yaml -n $NAMESPACE

echo "Deployment completed successfully!"

Cloudwatch monitor:

