Project Report: End-to-End CI/CD Pipeline on AWS

1. Project Objective

The primary objective of this project was to build a fully automated Continuous Integration and Continuous Deployment (CI/CD) pipeline on AWS. The system is designed to take a web application from a GitHub repository to a live, containerized deployment on an EC2 server, with the entire process triggered automatically by a git push.

2. Implementation & Features

This project was broken down into six core tasks, each demonstrating a key DevOps principle.

Task 1 & 2: Infrastructure & Configuration (Terraform & Ansible)

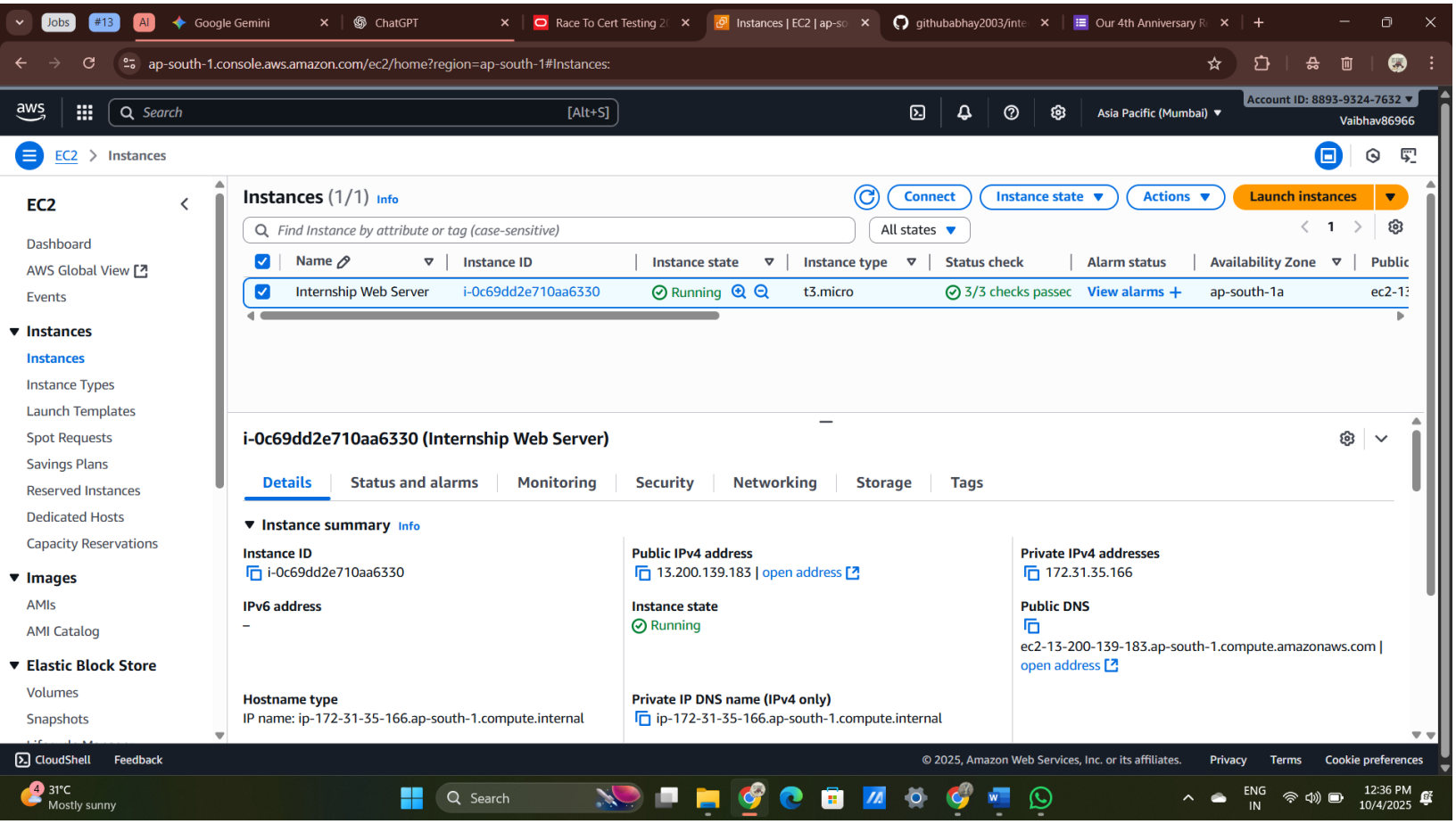
Infrastructure was provisioned using **Terraform** for an Infrastructure as Code (IaC) approach. This ensures the environment is reproducible and version-controlled. The server was then configured using **Ansible**, which provides an idempotent method for installing software.

Key Features:

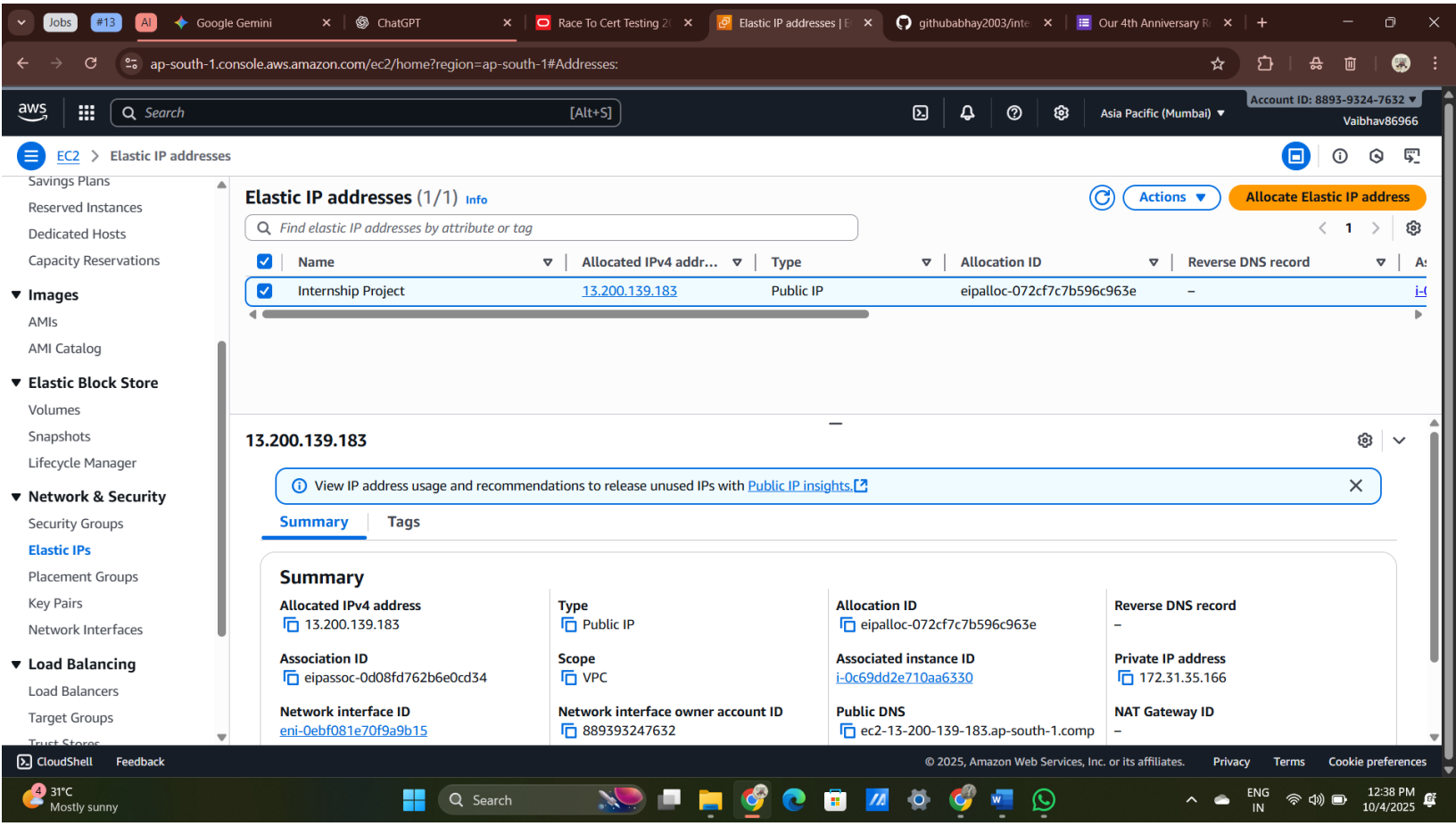
- An EC2 instance (t3.micro) was launched in the ap-south-1 region.
- A permanent, static IP address was assigned using an **AWS Elastic IP**.
- An AWS Security Group was configured to allow inbound traffic on **port 80 (HTTP)** and **port 22 (SSH)**.
- An Ansible playbook automated the complete installation of Docker Engine and Docker Compose on the server.

Visual Evidence:

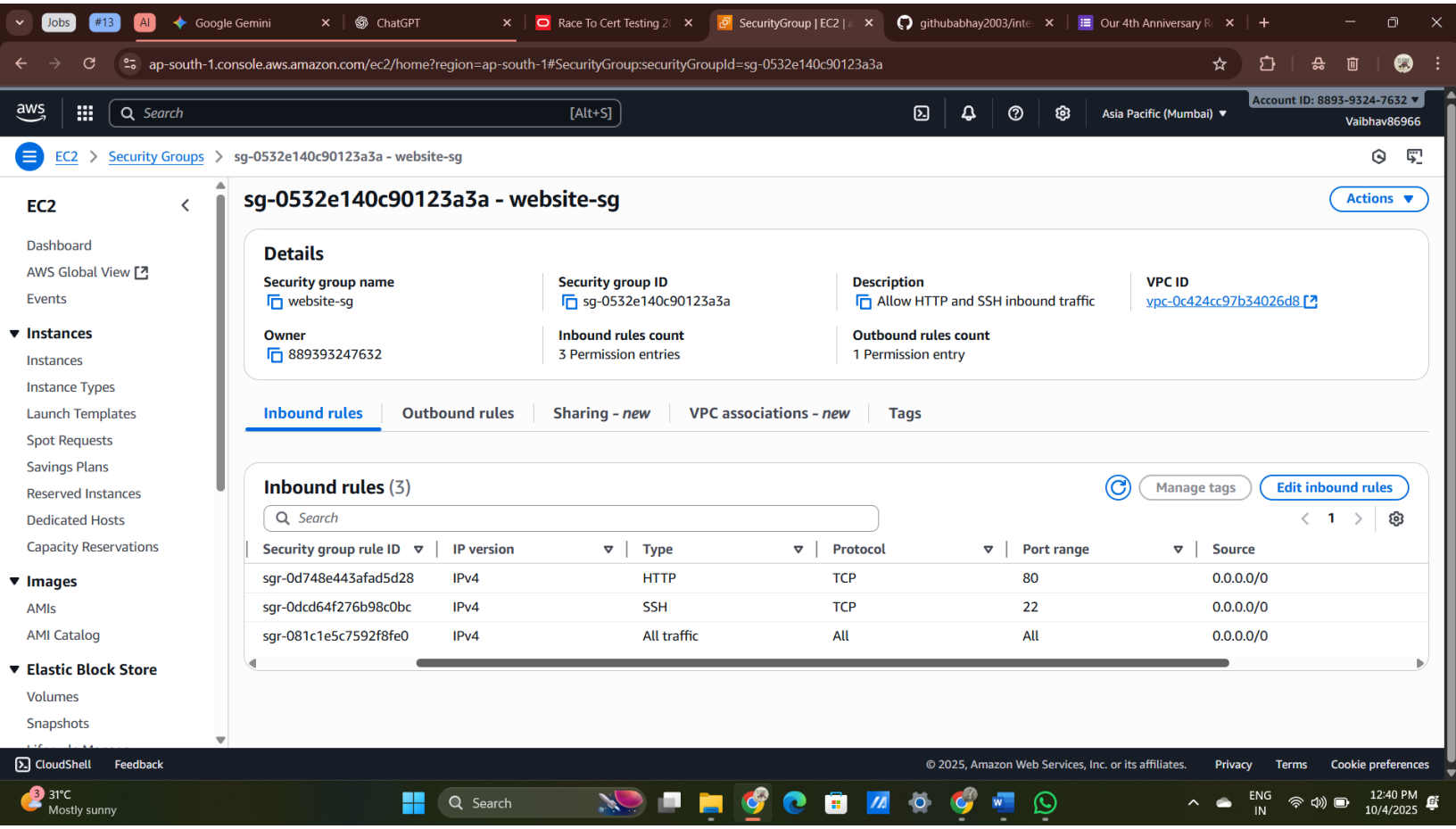
EC2 instance running in the AWS Console.



Elastic IP associated with the EC2 instance.



Security Group rules allowing web and SSH traffic.



Task 3 & 6: Application & Pipeline (Docker & Jenkins)

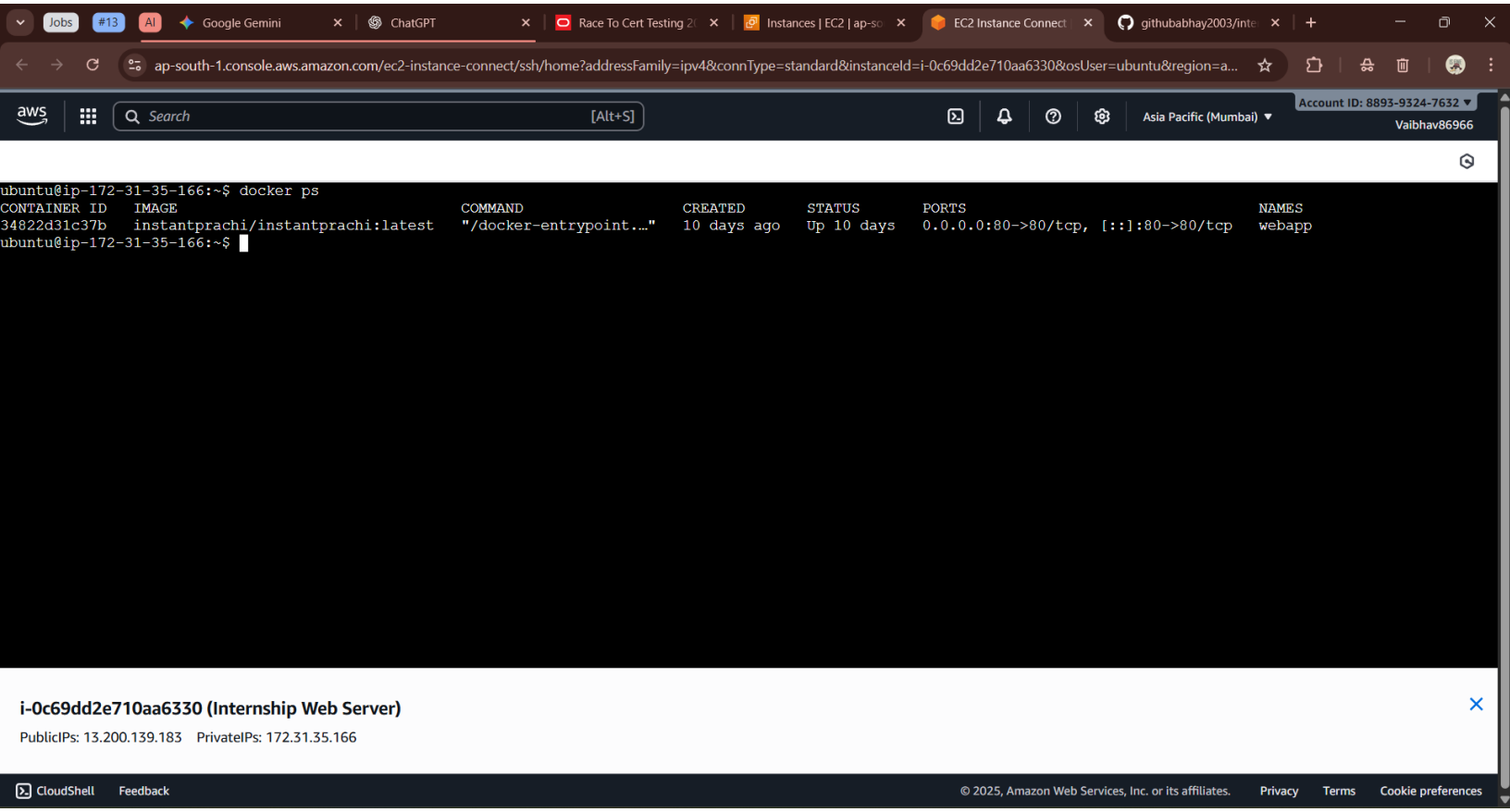
The application was containerized using **Docker** for consistency and portability. The entire CI/CD workflow was automated using a declarative **Jenkinsfile** (Pipeline as Code).

Key Features:

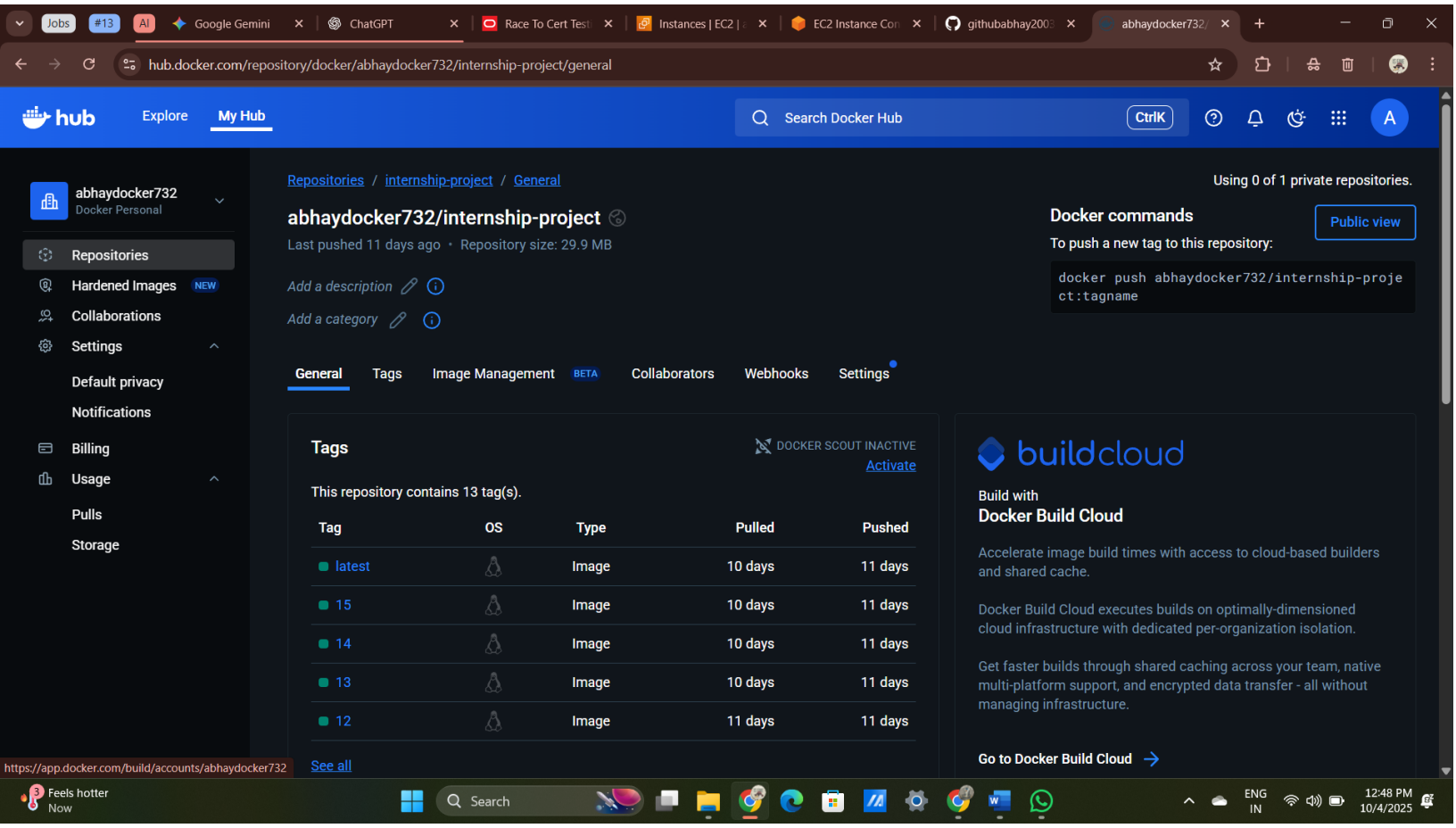
- A Dockerfile packages the Nginx web application into a portable image.
- A docker-compose.yml file defines how the application runs as a service.
- A Jenkinsfile defines the multi-stage pipeline: Checkout -> Build -> Push -> Deploy.
- Secrets (Docker Hub and SSH credentials) are managed securely using the Jenkins Credentials Manager.

Visual Evidence:

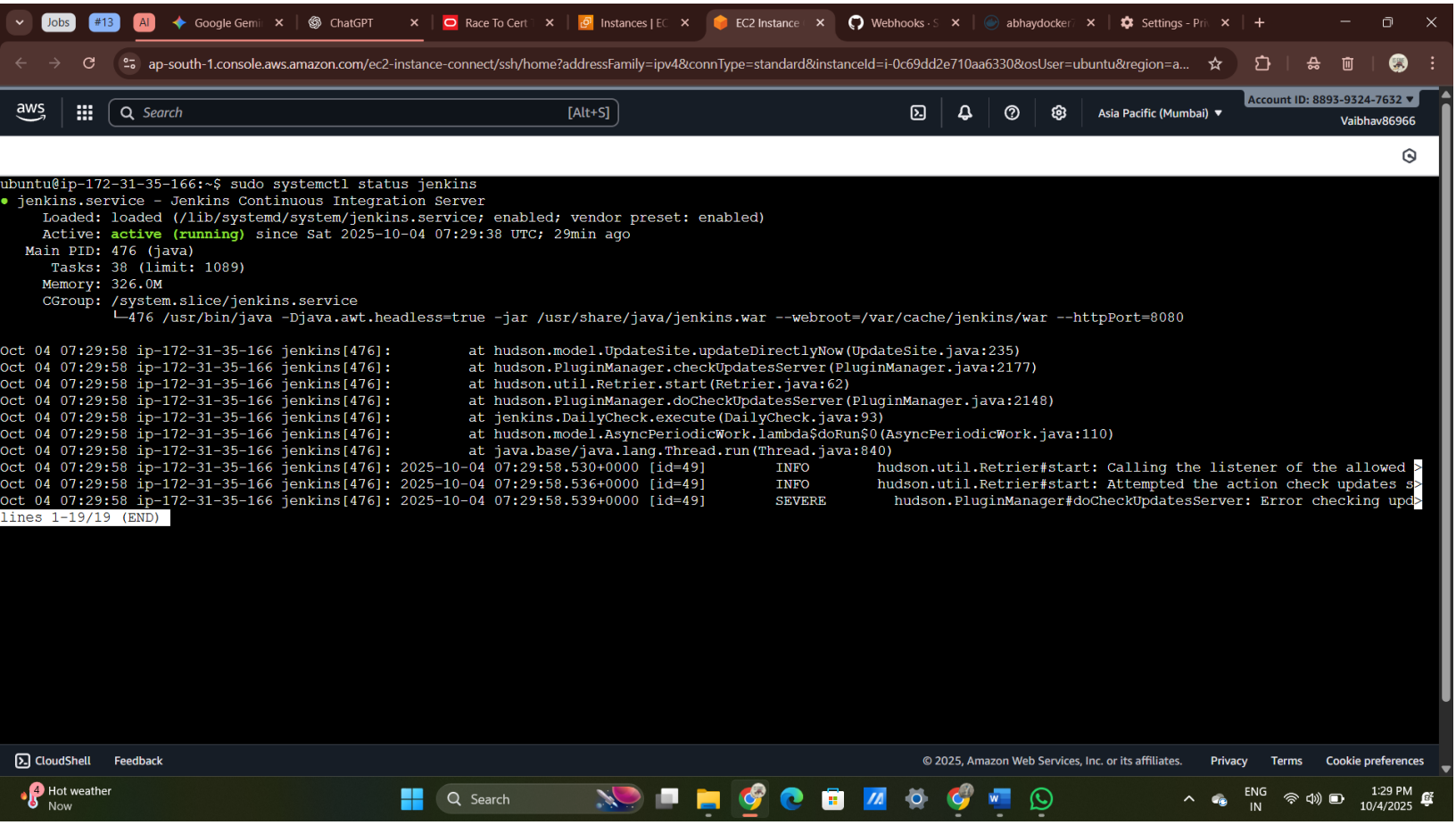
- *Final webapp container running on the server.*



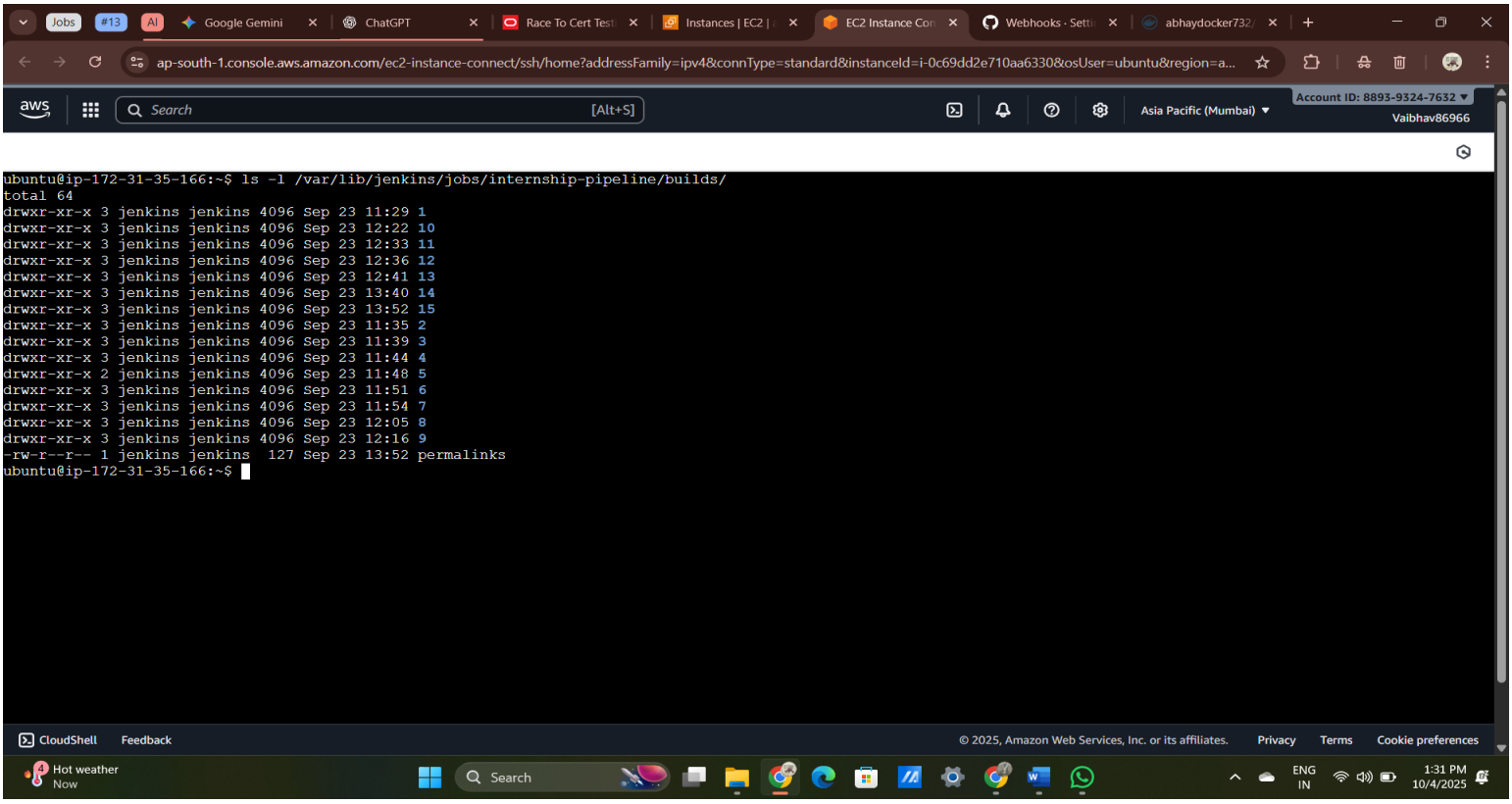
- *Pushed images visible in the Docker Hub repository.*

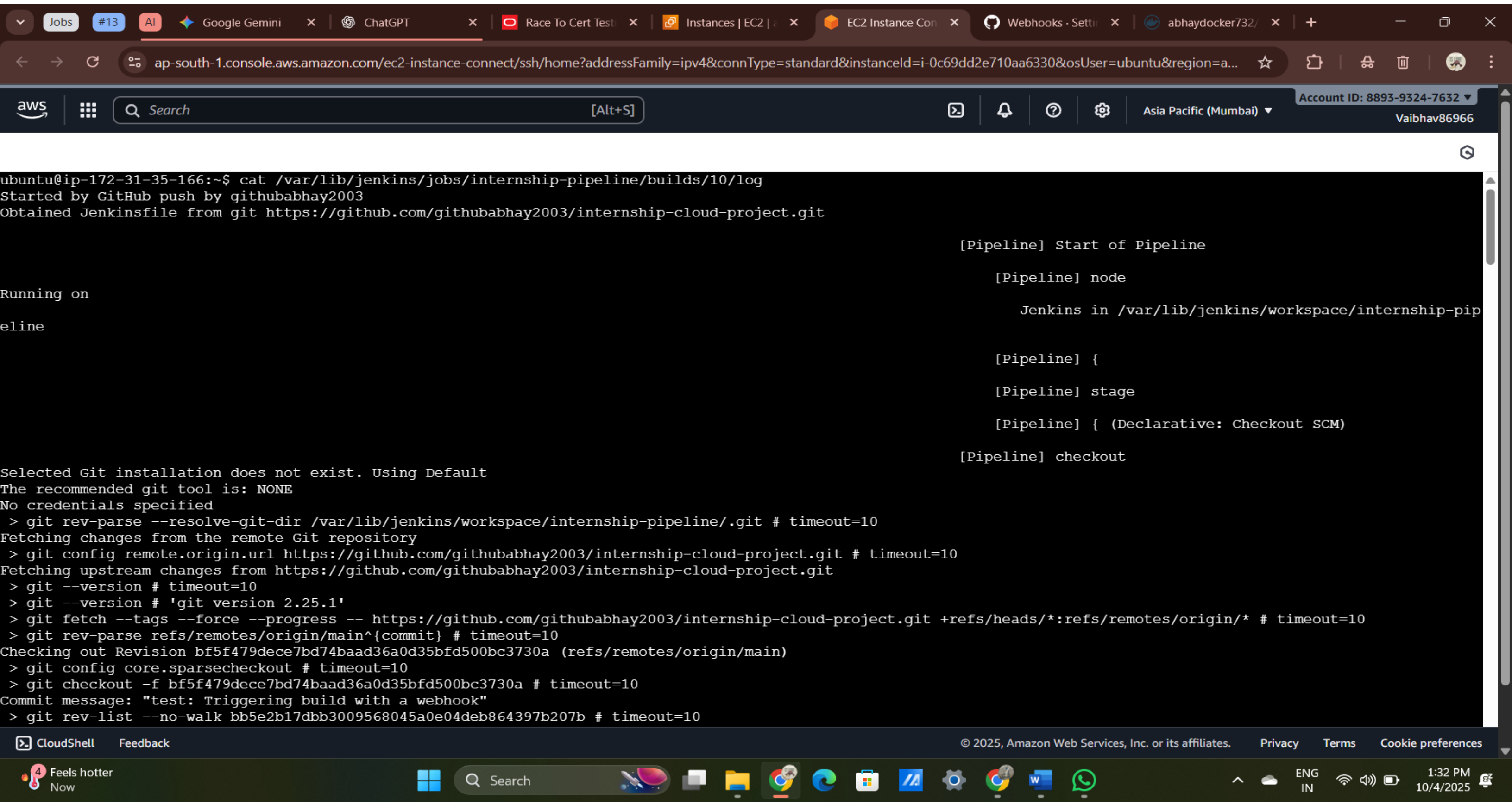


- *Jenkins pipeline dashboard showing a history of successful builds.*
- **Note:** Due to a persistent network routing issue preventing GUI access, the following command-line output is provided as proof that the Jenkins service is active and running correctly on the server.

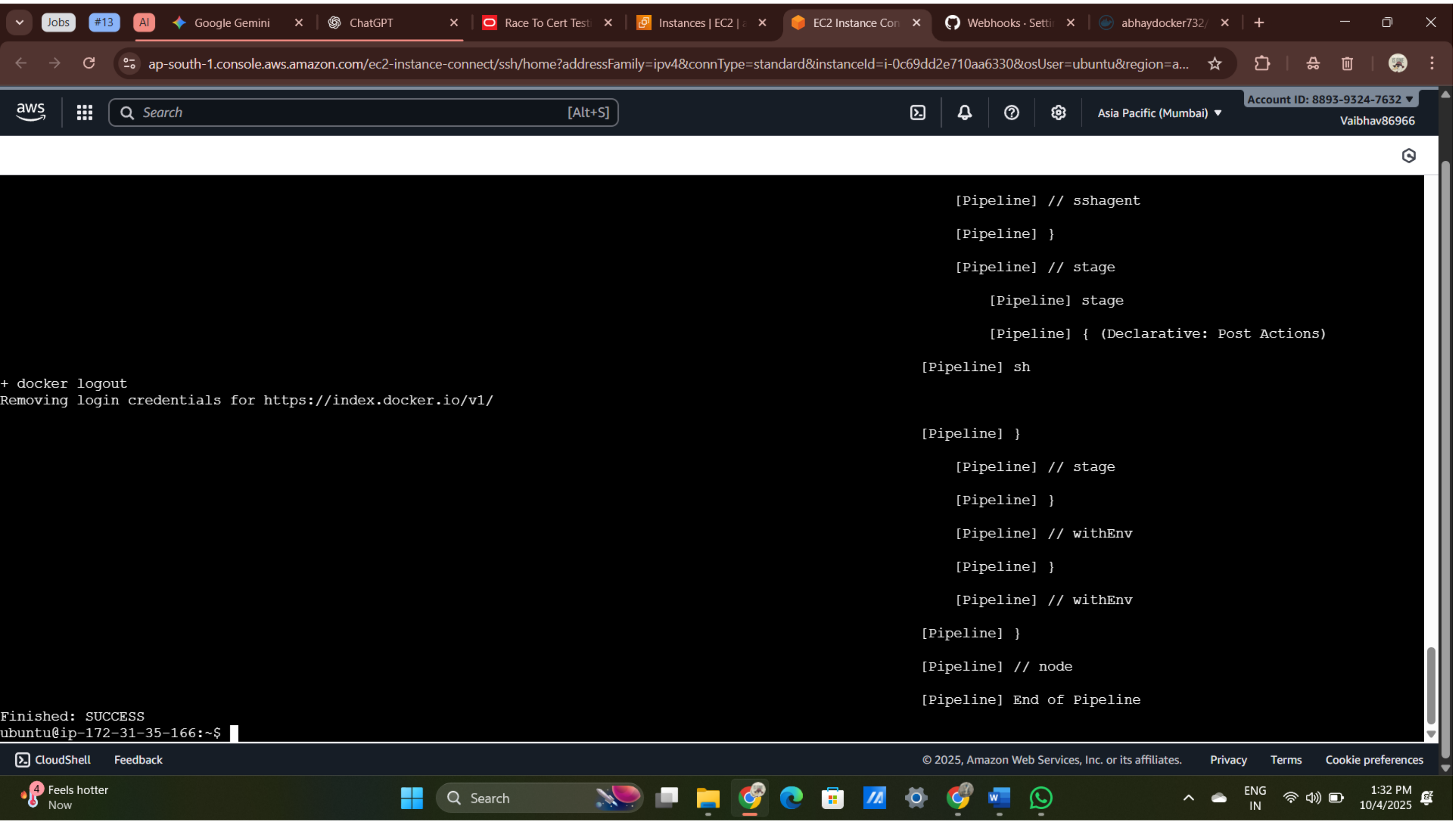


- *Detailed stage view of a successful pipeline run.*
- **Note:** The following is the console log output from the last successful build, confirming that all stages of the pipeline completed successfully.





```
ubuntu@ip-172-31-35-166:~$ cat /var/lib/jenkins/jobs/internship-pipeline/builds/10/log
Started by GitHub push by githubabhay2003
Obtained Jenkinsfile from git https://github.com/githubabhay2003/internship-cloud-project.git

                                                                                                            [Pipeline] Start of Pipeline

                                                                                                                [Pipeline] node

Running on                                                                                                         Jenkins in /var/lib/jenkins/workspace/internship-pip

eline

                                                                                                                [Pipeline] {

                                                                                                                [Pipeline] stage

                                                                                                                [Pipeline] { (Declarative: Checkout SCM)

                                                                                                            [Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
No credentials specified
 > git rev-parse --resolve-git-dir /var/lib/jenkins/workspace/internship-pipeline/.git # timeout=10
Fetching changes from the remote Git repository
 > git config remote.origin.url https://github.com/githubabhay2003/internship-cloud-project.git # timeout=10
Fetching upstream changes from https://github.com/githubabhay2003/internship-cloud-project.git
 > git --version # timeout=10
 > git --version # 'git version 2.25.1'
 > git fetch --tags --force --progress -- https://github.com/githubabhay2003/internship-cloud-project.git +refs/heads/*:refs/remotes/origin/* # timeout=10
 > git rev-parse refs/remotes/origin/main^{commit} # timeout=10
Checking out Revision bf5f479dece7bd74baad36a0d35bfd500bc3730a (refs/remotes/origin/main)
 > git config core.sparsecheckout # timeout=10
 > git checkout -f bf5f479dece7bd74baad36a0d35bfd500bc3730a # timeout=10
Commit message: "test: Triggering build with a webhook"
 > git rev-list --no-walk bb5e2b17dbb3009568045a0e04deb864397b207b # timeout=10
```



```
                                                                                                                [Pipeline] // sshagent

                                                                                                                [Pipeline] }

                                                                                                                [Pipeline] // stage

                                                                                                                    [Pipeline] stage

                                                                                                                    [Pipeline] { (Declarative: Post Actions)

                                                                                                            [Pipeline] sh

+ docker logout
Removing login credentials for https://index.docker.io/v1/

                                                                                                            [Pipeline] }

                                                                                                                [Pipeline] // stage

                                                                                                                [Pipeline] }

                                                                                                                [Pipeline] // withEnv

                                                                                                                [Pipeline] }

                                                                                                                [Pipeline] // withEnv

                                                                                                            [Pipeline] }

                                                                                                            [Pipeline] // node

                                                                                                            [Pipeline] End of Pipeline

Finished: SUCCESS
ubuntu@ip-172-31-35-166:~$ 
```

Task 4 & 5: Automation & Deployment

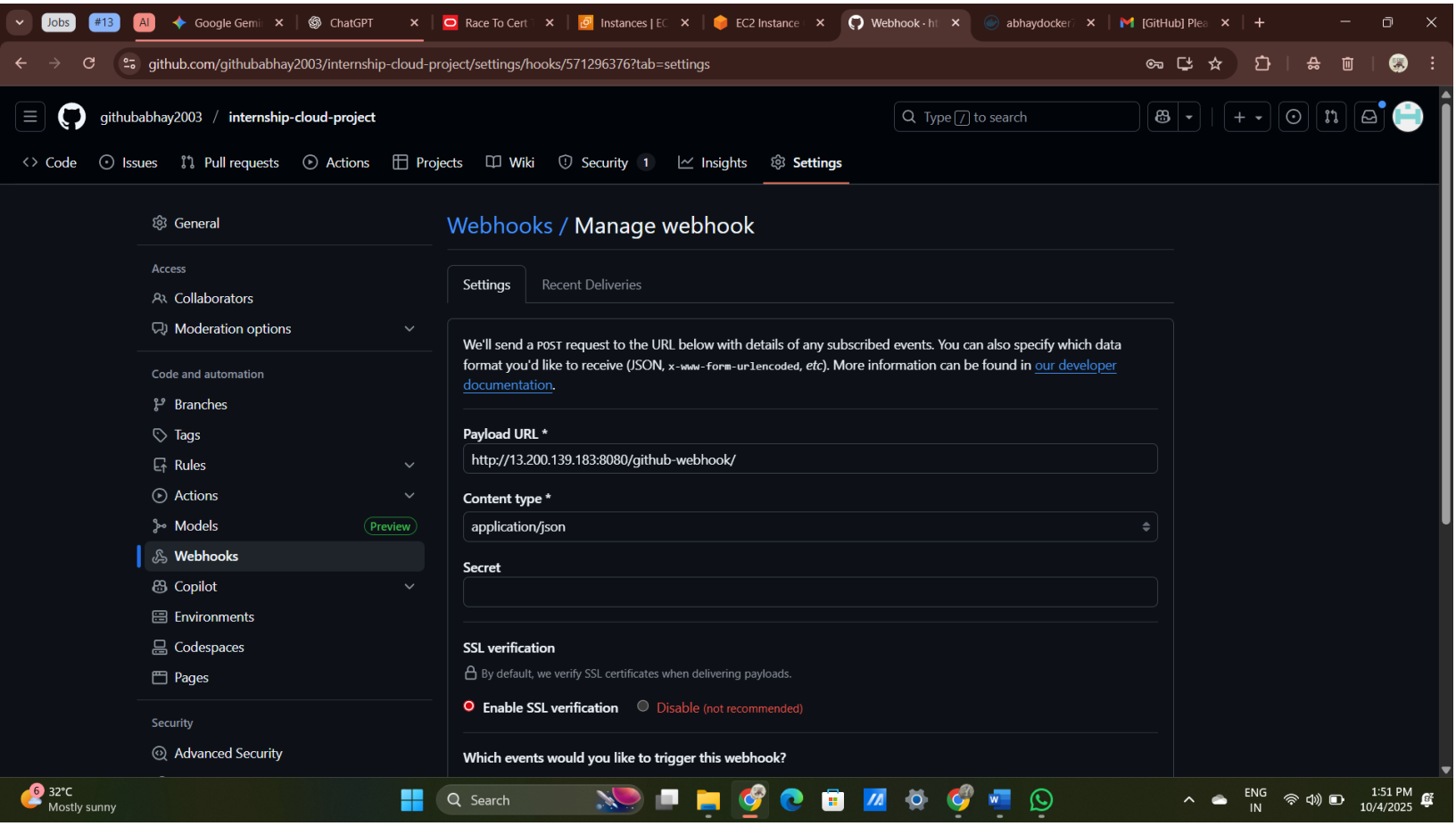
True Continuous Integration was achieved by linking GitHub to Jenkins with a **webhook**. The final deployment stage in the Jenkins pipeline handles the live release of the application.

Key Features:

- A GitHub webhook automatically triggers the Jenkins pipeline on every git push to the main branch.
- The deployment script in the Jenkinsfile securely connects to the EC2 server, pulls the latest image from Docker Hub, and restarts the container using Docker Compose.

Visual Evidence:

Successfully configured webhook in GitHub.



Verifying Web Server Functionality Locally

Note: *As conclusive evidence that the application is running correctly on the server, the curl http://localhost command was executed directly from the EC2 instance's command line.*

This command effectively simulates a browser visit from the server to itself, bypassing any external network or firewall issues. The successful return of the full HTML content from the index.html file, as seen in the screenshot below, provides definitive proof that:

1. *The Docker container is running successfully.*
2. *The Nginx web server inside the container is active and correctly configured.*

3. *The application's source code was properly deployed and is being served as expected.*

This test confirms that the application itself is fully functional, isolating the previously observed timeout issue to external network routing factors.

```
ubuntu@ip-172-31-35-166:~$ curl http://localhost
<!DOCTYPE html>
<html lang="en">

<head>
  <!-- Basic -->
  <meta charset="utf-8" />
  <meta http-equiv="X-UA-Compatible" content="IE=edge" />
  <!-- Mobile Metas -->
  <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no" />
  <!-- Site Metas -->
  <meta name="keywords" content="" />
  <meta name="description" content="" />
  <meta name="author" content="" />

  <title>Aseem</title>

  <!-- slider stylesheet -->
  <link rel="stylesheet" type="text/css" href="https://cdnjs.cloudflare.com/ajax/libs/OwlCarousel2/2.1.3/assets/owl.carousel.min.css" />

  <!-- bootstrap core css -->
  <link rel="stylesheet" type="text/css" href="css/bootstrap.css" />

  <!-- fonts style -->
  <link href="https://fonts.googleapis.com/css?family=Open+Sans:400,700|Poppins:400,700|Roboto:400,700&display=swap" rel="stylesheet" />

  <!-- Custom styles for this template -->
  <link href="css/style.css" rel="stylesheet" />
  <!-- responsive style -->
  <link href="css/responsive.css" rel="stylesheet" />
</head>

<body>
  <div class="hero_area">
```

```
            Newsletter
          </h4>
          <form action="#">
            <input type="text" placeholder="Enter Your Email" />
            <button type="submit">
              Subscribe
            </button>
          </form>
        </div>
      </div>
    </div>
  </div>
</section>

<!-- end info_section -->

<!-- footer section -->
<footer class="container-fluid footer_section">
  <p>
    &copy; 2019 All Rights Reserved By
    <a href="https://html.design/">Free Html Templates</a>
  </p>
</footer>
<!-- footer section -->

<script src="js/jquery-3.4.1.min.js"></script>
<script src="js/bootstrap.js"></script>
<script src="js/circles.min.js"></script>
<script src="js/custom.js"></script>


</body>

</html>ubuntu@ip-172-31-35-166:~$
```