TITLE: CI-CD PIPELINE USING JENKINS, ANSIBLE AND WEBHOOKS

Name: Dipanshu Raj
Reg. No: 12210839
Roll No: 64
Sub Code: INT 333
T. Name: Priyanka Gupta

INTRODUCTION:

This architecture is a modern, efficient, and automated deployment pipeline designed to support teams in delivering containerized applications quickly and reliably. By combining the strengths of **GitHub**, **GitHub Actions**, **Docker Hub**, and **Ansible**, it enables continuous integration, seamless image management, and streamlined deployment.

The workflow begins with developers collaborating on **GitHub**, where source code and configuration files are maintained. Changes trigger **GitHub Actions**, which automate tasks such as testing, building, and packaging the application into Docker images. These images are then stored in **Docker Hub**, a trusted registry for containerized applications, ensuring easy access for deployment.

Ansible brings powerful orchestration and deployment capabilities to this setup, fetching the images from Docker Hub and deploying them on target servers or environments using **Docker containers**. This ensures scalability, reproducibility, and consistent application performance across different environments.

This architecture is ideal for organizations embracing DevOps and modern software delivery practices. It offers a structured, end-to-end solution for building, managing, and deploying containerized applications, enabling faster delivery cycles, reduced downtime, and effortless scaling.

SPECIFIC REQUIREMENTS:

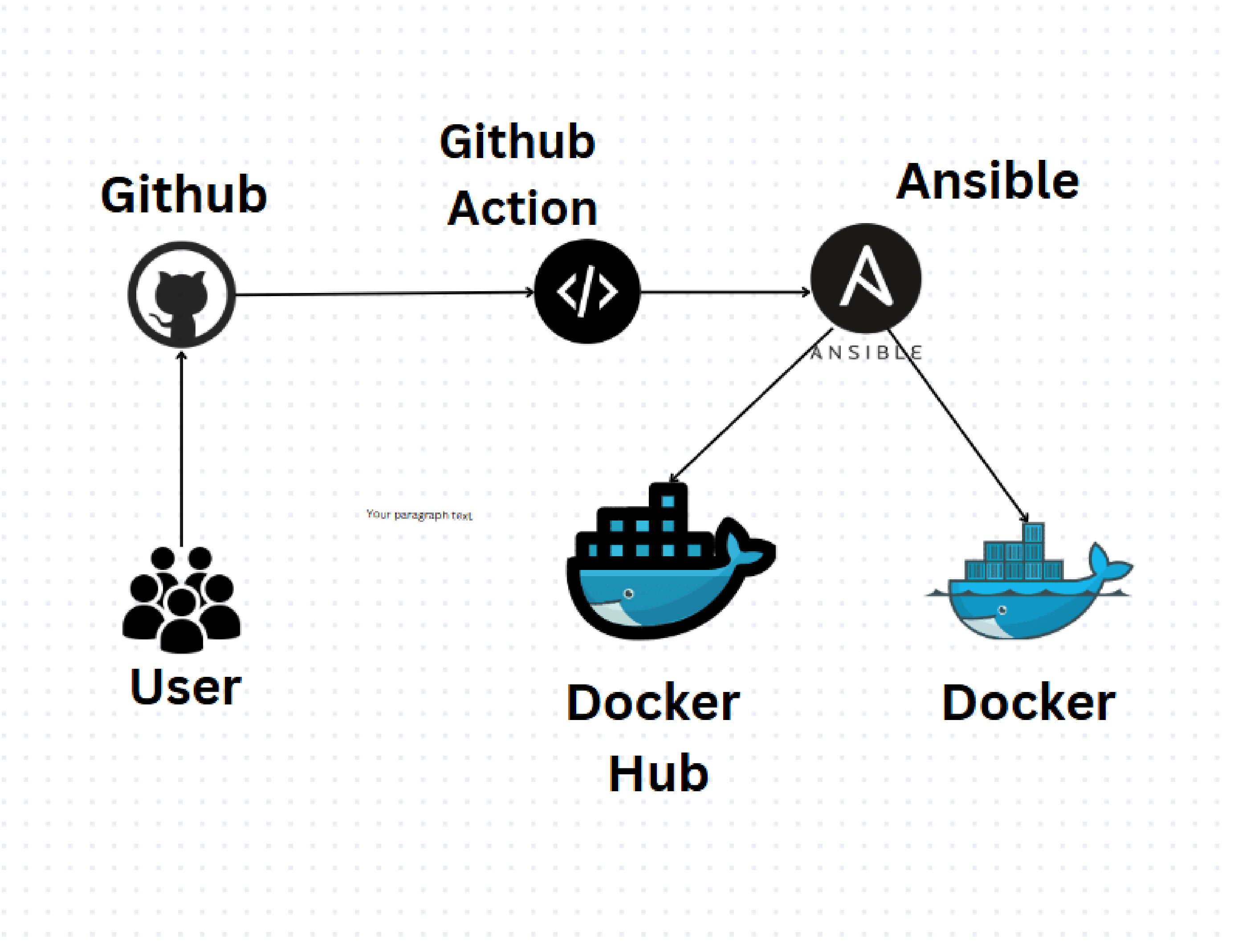
- Functional Requirements
  Ansible:
  1. Manage configuration of servers.
  2. Automate application deployment.
  3. Execute scripts to set up necessary infrastructure.

  Docker:
  1. Build and manage application containers.
  2. Ensure the application runs consistently across different environments.

  GitHub Actions:
  1. Trigger automated workflows upon code commits or pull requests.
  2. Build and test code in the pipeline.
  3. Deploy successful builds to production or staging environments.
- Non-Functional Requirements
  Scalability:
  1. The pipeline should handle increasing codebase size or additional environments without performance degradation.
- Reliability:
  Ensure minimal downtime during deployment.
  Automate rollback in case of failed deployments.
- Usability:
  Provide clear logs and feedback during pipeline execution.
  Maintain simplicity in pipeline configuration files.
  Hardware Requirements
- Development Environment:
  A system with Docker and GitHub CLI installed.
  Minimum 4 GB RAM and 20 GB storage.
- Target Environment:
  Servers capable of running Docker containers.
  Ansible-compatible OS (e.g., Ubuntu, CentOS).

- **SOFTWARE REQUIREMENTS:**
- Operating Systems:
  Development: Linux/Windows/macOS.
  Deployment: Linux (Ubuntu, CentOS).
- Tools and Frameworks:
  Docker 24.0 or higher.
  Ansible 2.13 or higher.
  GitHub Actions for CI/CD workflows.
- Codebase Dependencies:
  Python 3.8 or higher for Ansible playbooks.
  Application-specific libraries and dependencies.

Architecture:



This architecture represents a comprehensive **CI/CD (Continuous Integration/Continuous Deployment) pipeline** designed to streamline the process of building, testing, managing, and deploying containerized applications. By integrating tools like **GitHub Actions**, **Docker Hub**, and **Ansible**, this pipeline supports professional teams in achieving scalable, efficient, and automated workflows.

Key Components and Workflow:

1. **GitHub: The Source of Truth**
    - **Centralized Repository**: GitHub acts as the core platform for managing application code, Dockerfiles, and Ansible playbooks.

- **Collaboration**: Developers use GitHub to collaborate seamlessly, ensuring all updates, commits, and pull requests are synchronized in a structured manner.

- **Trigger Point**: Each commit or pull request serves as the starting point for automation in the CI/CD pipeline.

2. **GitHub Actions: Workflow Automation**
    - **Build Automation**: GitHub Actions automates processes such as running tests, checking code quality (linting), and building Docker images.

    - **Integration**: Upon a successful build, the resulting Docker image is pushed to **Docker Hub**.

    - **Custom Workflows**: Tailored workflows can be defined to handle various stages like notifications, environment setup, or advanced testing scenarios.

3. **Docker Hub: Container Registry**
    - **Image Management**: Docker Hub serves as a secure repository for storing, managing, and distributing Docker images.

    - **Versioning**: Teams can manage image versions efficiently, ensuring the latest or specific tagged versions are available for deployment.

    - **Access Control**: Role-based access and secure sharing ensure proper control over who can pull or push images.

4. **Ansible: Deployment Automation**
    - **Pulling Docker Images**: Ansible automates the process of fetching Docker images from Docker Hub for deployment.

    - **Orchestration**: Using playbooks, Ansible defines the logic for deployment, ensuring repeatable and consistent results across environments.

- **Configuration Management**: Handles configurations, container scaling, and environment-specific customizations effortlessly.

5. **Docker Host: Application Execution**
    - **Runtime Environment**: Deployed containers run on Docker hosts, which can be physical servers, virtual machines, or cloud instances.

    - **Isolation**: Each container operates as an independent unit, ensuring high reliability and minimizing conflicts.

    - **Portability**: Applications can seamlessly run across environments without dependency issues.

Key Features of This Architecture:

- **Continuous Integration and Deployment**: Automates the entire pipeline from code commits to application deployment, reducing human intervention and errors.

- **Portability Through Containers**: Docker ensures applications behave consistently across development, testing, and production environments.

- **Scalability and Flexibility**: Ansible simplifies scaling applications horizontally or vertically based on demand.

- **Secure Image Management**: Docker Hub offers secure storage and sharing capabilities, safeguarding the integrity of containerized applications.

- **Rapid Delivery**: With automated workflows, new features and updates can be deployed much faster, aligning with agile development practices.

- **Reliability**: Ansible's idempotent approach ensures deployments are predictable and error-free, even during repetitive executions.

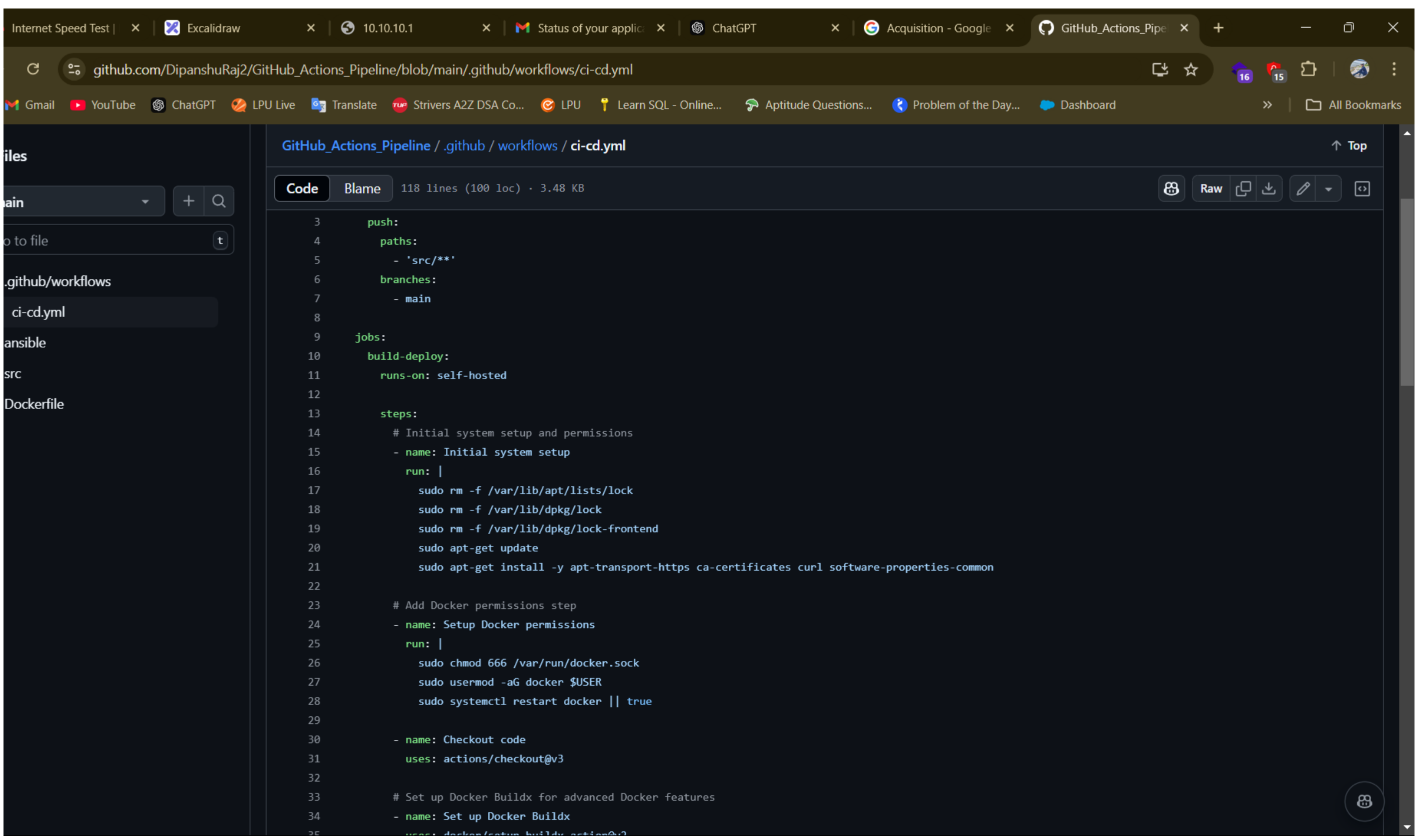
Ideal Use Cases:

- **DevOps Teams**: For building and maintaining a reliable CI/CD pipeline.

- **Cloud-Native Applications**: Ideal for deploying containerized apps in hybrid or multi-cloud environments.

PSEUDOCODE:

In this project, the implementation relies on logical workflows and processes that are represented through pseudocode. Pseudocode serves as a bridge between conceptual understanding and actual code, providing a clear outline of the steps involved in executing key functionalities. Below is a segment of pseudocode from the project, designed to highlight the structure and logic driving its functionality.
The pseudocode outlines the **core processes** involved, ensuring that the logic is easy to interpret and implement across different programming languages. It provides a step-by-step explanation of how the application handles specific tasks, making the development process more efficient and accessible for future enhancements.
Here is the pseudocode for the project:



github.com/DipanshuRaj2/GitHub_Actions_Pipeline/blob/main/.github/workflows/ci-cd.yml

GitHub_Actions_Pipeline / .github / workflows / ci-cd.yml

Code | Blame — 118 lines (100 loc) · 3.48 KB

```yaml
        uses: dawidd6/action-ansible-playbook@v2
        with:
          inventory: ansible/inventory
          playbook: ansible/playbook.yml
          options: |
            --become
            --become-method=sudo
            --become-user=root
            -vvvv
          sudo: true

      - name: Pull Docker image from Docker Hub
        run: |
          docker pull ${{ secrets.DOCKER_USERNAME }}/my-static-website:latest

      - name: Run Docker container
        run: |
          docker run -d -p 80:80 --name my-static-website ${{ secrets.DOCKER_USERNAME }}/my-static-website:latest


      # Clean up after deployment
      - name: Clean up
        if: always()
        run: |
          sudo apt-get clean
          sudo rm -rf /var/lib/apt/lists/*

      # Notify on failure
      - name: Notify on failure
        if: failure()
        run: |
          echo "Workflow failed! Check the logs for more details."
          # Add your notification commands here (e.g., email, Slack)
```

github.com/DipanshuRaj2/GitHub_Actions_Pipeline/blob/main/ansible/playbook.yml

Code | Blame — 69 lines (60 loc) · 1.9 KB

```yaml
---
- hosts: localhost
  become: true  # Run with sudo privileges
  become_user: root
  tasks:
    # Remove any apt locks
    - name: Ensure no apt lock exists
      file:
        path: /var/lib/apt/lists/lock
        state: absent
      ignore_errors: yes

    - name: Ensure no apt lock exists for dpkg
      file:
        path: /var/lib/dpkg/lock
        state: absent
      ignore_errors: yes

    - name: Ensure no apt lock exists for dpkg-frontend
      file:
        path: /var/lib/dpkg/lock-frontend
        state: absent
      ignore_errors: yes

    - name: Wait for lock release
      command: sleep 5

    # Configure apt sources correctly
    - name: Configure apt repositories
      copy:
        dest: /etc/apt/sources.list
        content: |
          deb http://us-east-1.ec2.archive.ubuntu.com/ubuntu/ jammy main restricted universe multiverse
          deb http://us-east-1.ec2.archive.ubuntu.com/ubuntu/ jammy-updates main restricted universe multiverse
```

GitHub_Actions_Pipeline / ansible / playbook.yml

Code Blame 69 lines (60 loc) · 1.9 KB

```yaml
            deb http://us-east-1.ec2.archive.ubuntu.com/ubuntu/ jammy-updates main restricted universe multiverse
            deb http://us-east-1.ec2.archive.ubuntu.com/ubuntu/ jammy-backports main restricted universe multiverse
            deb http://security.ubuntu.com/ubuntu jammy-security main restricted universe multiverse
          mode: '0644'

    - name: Clean apt cache
      command: apt-get clean

    - name: Update apt packages
      apt:
        update_cache: yes
        upgrade: dist
        cache_valid_time: 3600
      register: apt_update_result

    # - name: Install required packages
    #   apt:
    #     name: "{{ item }}"
    #     state: present
    #     update_cache: no
    #   loop:
    #     # - docker.io
    #     # - docker-compose
    #     - python3-pip

    - name: Ensure Docker service is started and enabled
      service:
        name: docker
        state: started
        enabled: yes

    - name: Add current user to docker group
      user:
```

DipanshuRaj2 / GitHub_Actions_Pipeline

Code · Issues · Pull requests · Actions · Projects · Wiki · Security · Insights · Settings

Files: main · .github · ansible · src · Dockerfile

GitHub_Actions_Pipeline / Dockerfile

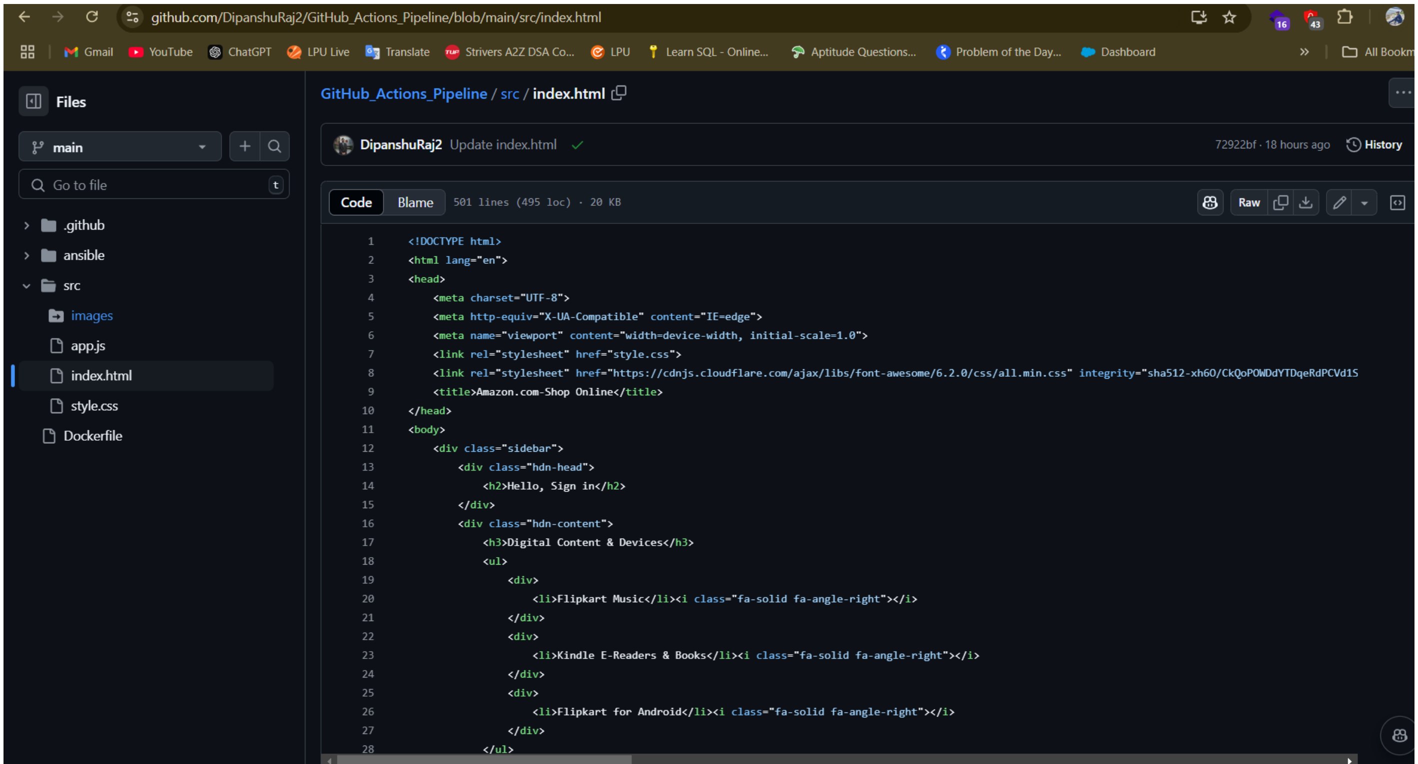
DipanshuRaj2 Update Dockerfile — 3f0f7f0 · 3 days ago · History

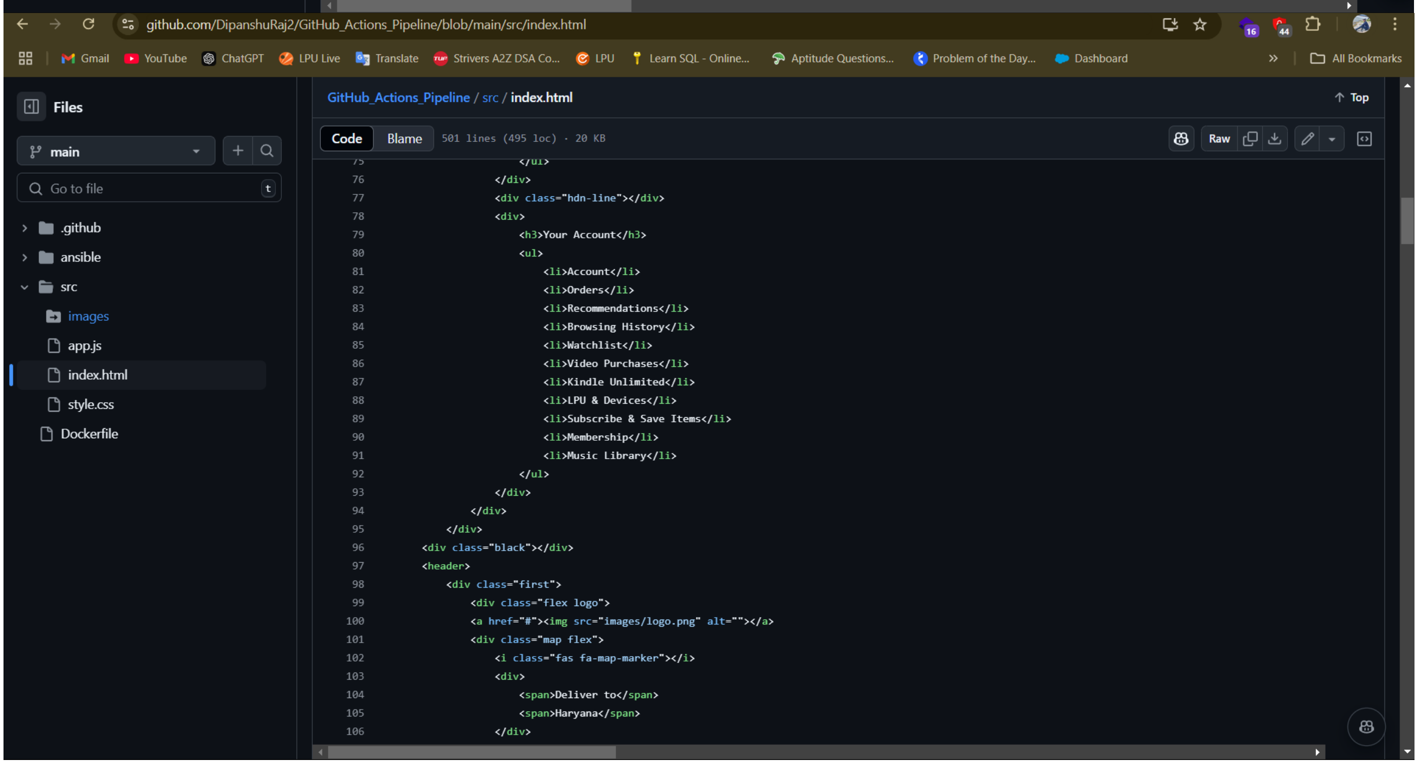
Code Blame 8 lines (6 loc) · 185 Bytes

```dockerfile
# Use NGINX to serve the static content
FROM nginx:alpine

# Copy the website content to the NGINX container
COPY src/ /usr/share/nginx/html/

# Expose the default NGINX port
EXPOSE 80
```





This representation is crucial for understanding the underlying operations of the project, including key algorithms, workflows, and their interactions within the system. By analyzing the pseudocode, one can gain a deeper insight into the problem-solving approach and the overall design of the application.

IMPLEMENTATION

Phase 1: Setting Up the Development Environment

The implementation of the CI/CD pipeline project involved setting up, configuring, and integrating tools and workflows to achieve a seamless continuous integration and delivery system.The project was structured in the following phase :

● **Environment Preparation**:
Installed necessary tools including Docker, Ansible, and GitHub CLI on the development machine (AWS EC2)
Configured access to the repository hosting the application code.
● Infrastructure Configuration:
● Deployed target servers with Linux (Ubuntu) as the operating system to support Docker containers and Ansible.

Phase 2: Containerization with Docker

DockerSetup:
● Created Dockerfile to define the application environment.
● Configured Docker Compose to manage multi-container applications if required.
● Testing Docker Images:
● Built the Docker image using docker build and tested its functionality locally.
● Published the image to a Docker registry for deployment

Phase 3: Configuration Management with Ansible
● PlaybookCreation:
● Developed Ansible playbooks to automate server setup, such as installing Docker, pulling container images, and deploying containers.
● Inventory Management:
● Created an inventory file specifying the IP addresses or hostnames of target servers.
● Execution and Testing:
● Executed playbooks locally using ansible-playbook commands and ensured proper deployment on target servers

Phase 4: CI/CD Workflow with GitHub Actions

●WorkflowDefinition:

● Addeda.github/workflows/deploy.yml file to the repository to define the CI/CD process. Included steps for:

● Pulling the latest code from the repository.

● Building the Docker image

● Runningautomated tests.

● Deploying the updated application using Ansible.

● Secrets Management:

● Secured credentials such as SSH keys and Docker registry tokens using GitHub Actions secrets.

Phase 5: Self-Hosted GitHub Runner

● Toenhanceflexibility and control over the CI/CD pipeline, a self-hosted GitHub runner was configured. The following steps were performed:

● Installed and registered the runner on a dedicated server.

● Integrated the runner with the GitHub repository.

● Optimized runner performance for executing workflows and handling build artifacts.

Phase 6: Testing and Validation

● UnitTesting:

● Ensured all test cases passed successfully during the build process.

● End-to-End Testing:

● Validated the CI/CD pipeline by making changes to the application code and observing automated deployment to the staging environment.

● ErrorHandling:

● Implemented rollback mechanisms in case of deployment failures.

Phase 7: Deployment and Monitoring

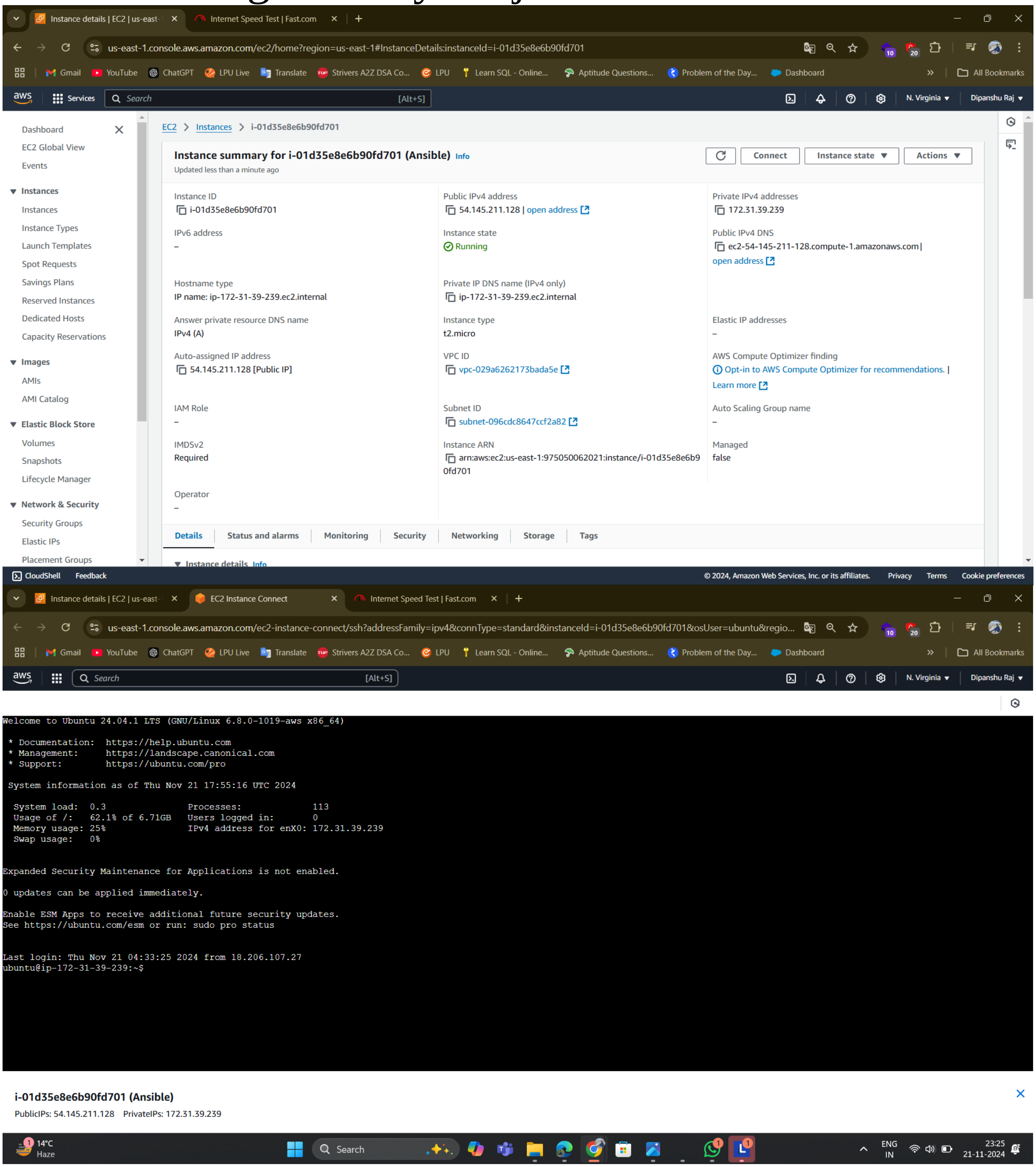
● Production Deployment:

● Configured GitHub Actions workflows to deploy successfully tested builds to the production environment.

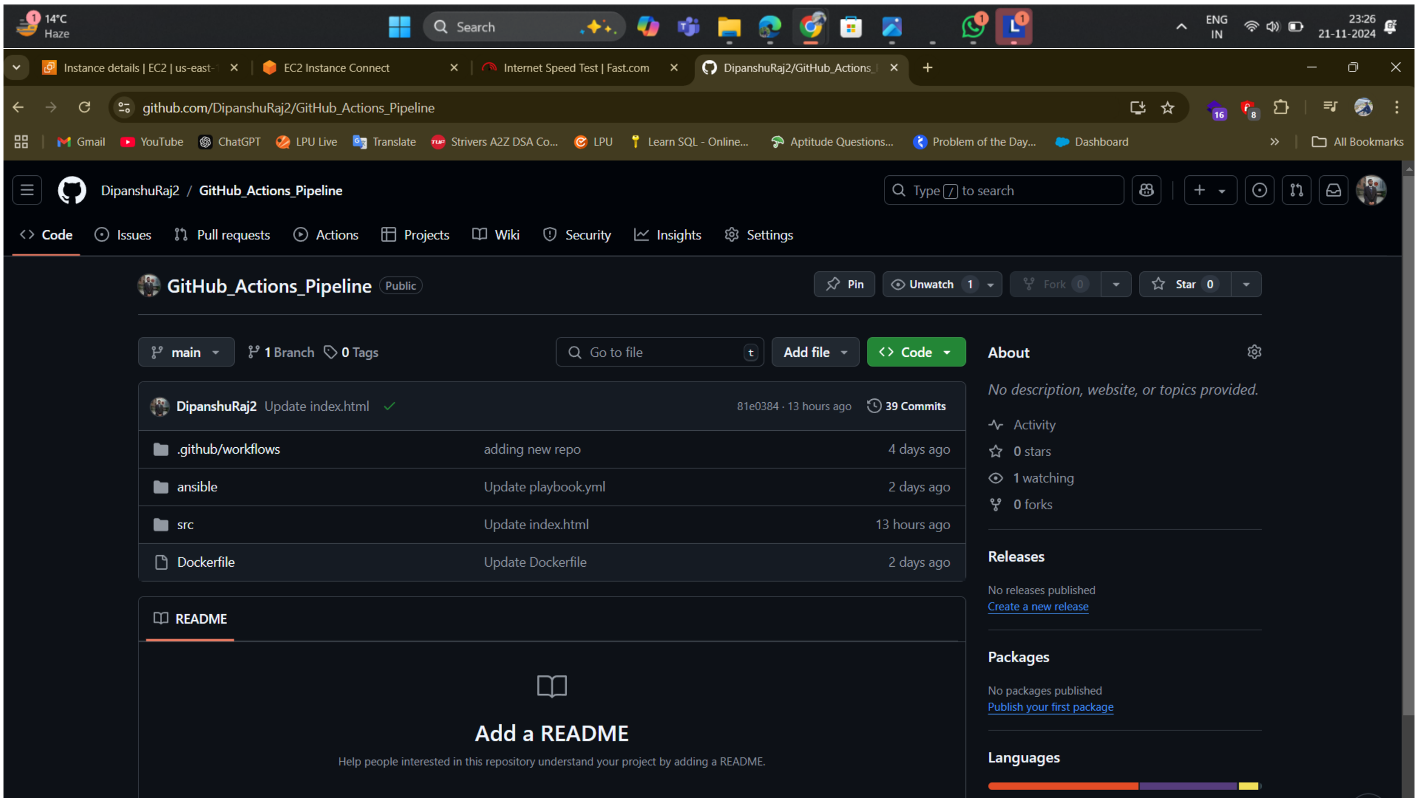
● Monitoring:

● Integrated Docker logging and GitHub Actions build logs to monitor pipeline performance

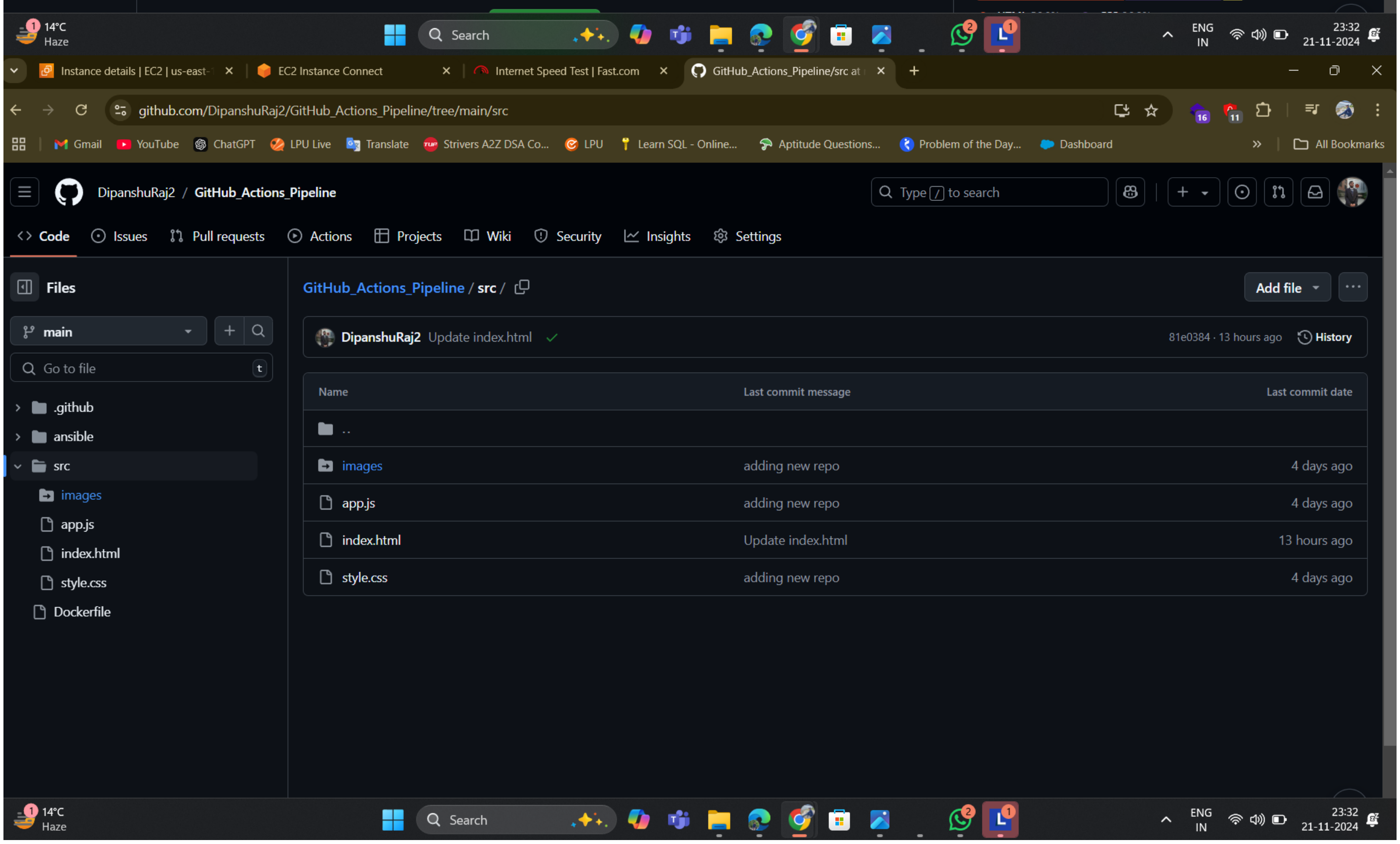
Here are Images of My Project:



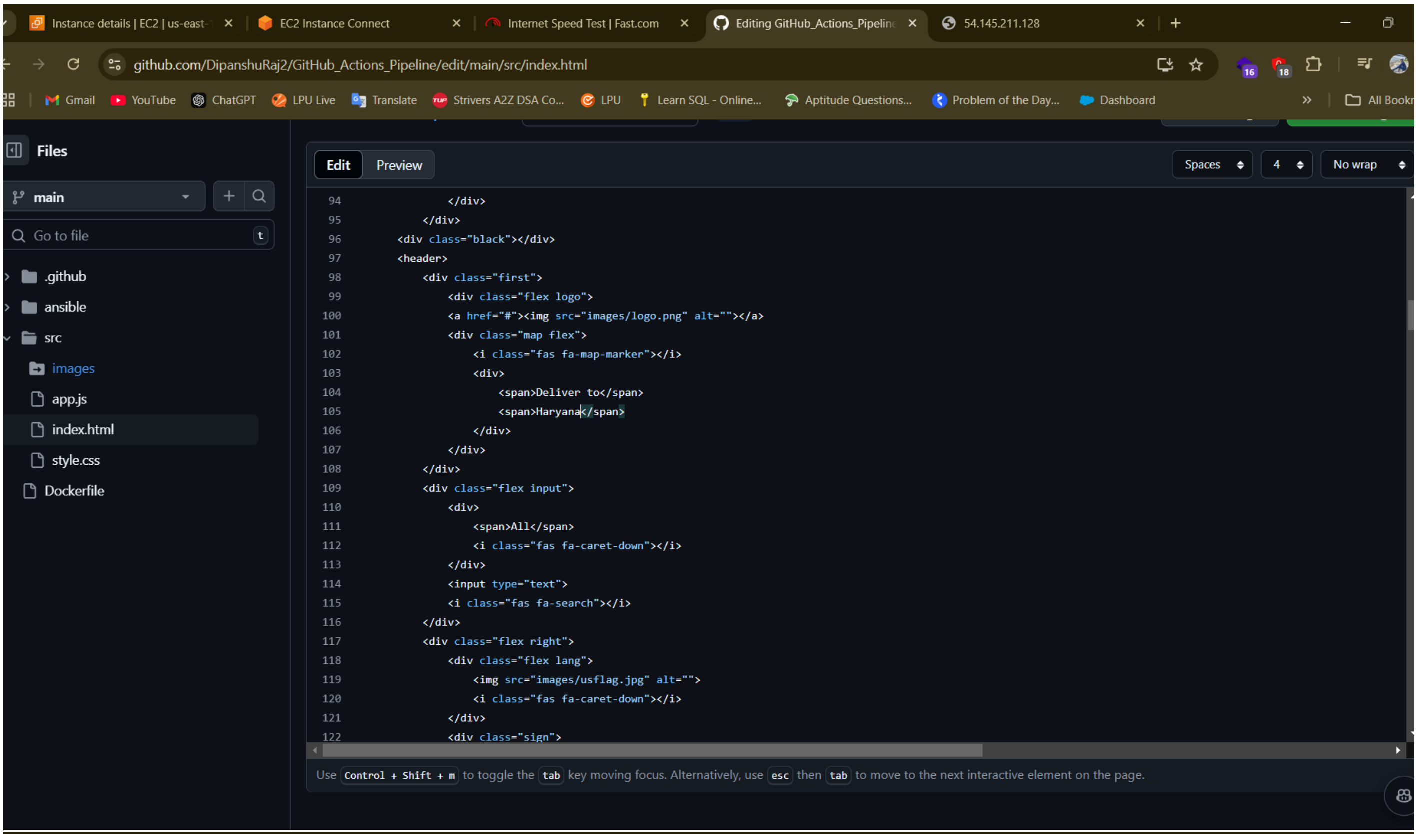
11 packages are up to date.
buntu@ip-172-31-39-239:~$ cd actions-runner/
buntu@ip-172-31-39-239:~/actions-runner$ ./run.sh
Connected to GitHub
urrent runner version: '2.320.0'
024-11-21 17:56:38Z: Listening for Jobs
i-01d35e8e6b90fd701 (Ansible)
PublicIPs: 54.145.211.128 PrivateIPs: 172.31.39.239



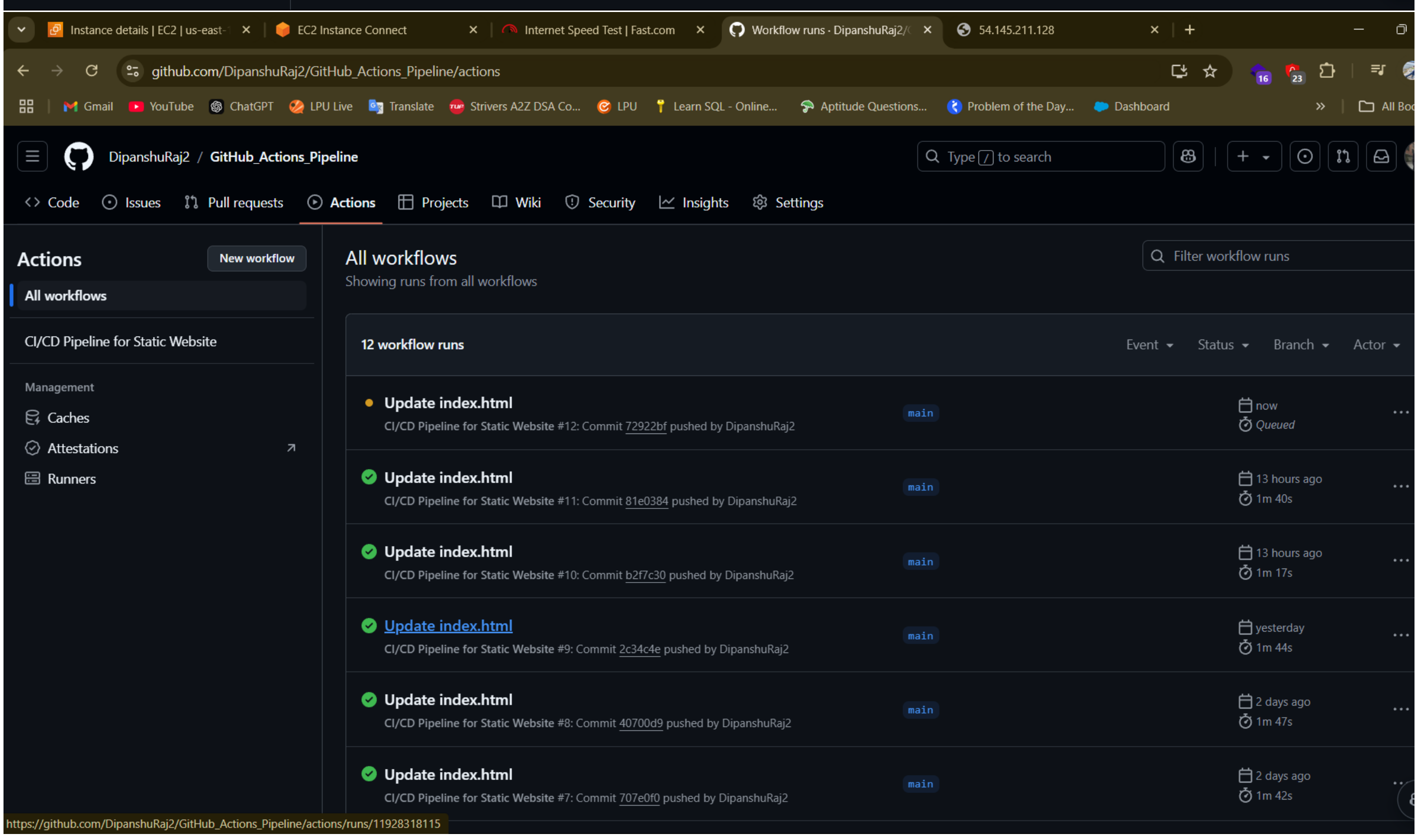
github.com/DipanshuRaj2/GitHub_Actions_Pipeline
DipanshuRaj2 / GitHub_Actions_Pipeline
Code Issues Pull requests Actions Projects Wiki Security Insights Settings
GitHub_Actions_Pipeline Public
main 1 Branch 0 Tags
Go to file
Add file
Code
DipanshuRaj2 Update index.html
81e0384 · 13 hours ago 39 Commits
.github/workflows adding new repo 4 days ago
ansible Update playbook.yml 2 days ago
src Update index.html 13 hours ago
Dockerfile Update Dockerfile 2 days ago
README
Add a README
Help people interested in this repository understand your project by adding a README.
About
No description, website, or topics provided.
Activity
0 stars
1 watching
0 forks
Releases
No releases published
Create a new release
Packages
No packages published
Publish your first package
Languages
23:26
21-11-2024



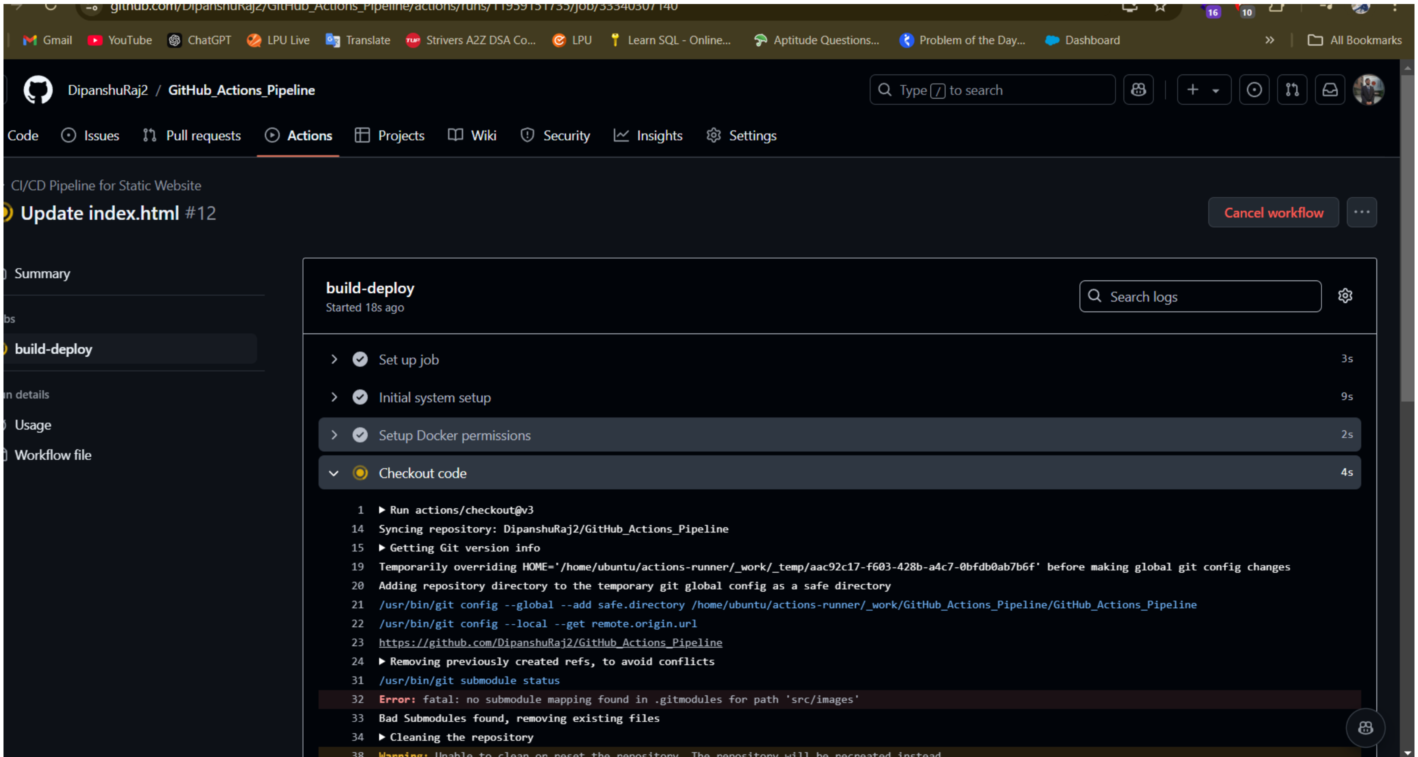
github.com/DipanshuRaj2/GitHub_Actions_Pipeline/tree/main/src
DipanshuRaj2 / GitHub_Actions_Pipeline
Code Issues Pull requests Actions Projects Wiki Security Insights Settings
Files
main
Go to file
.github
ansible
src
images
app.js
index.html
style.css
Dockerfile
GitHub_Actions_Pipeline / src /
Add file
DipanshuRaj2 Update index.html
81e0384 · 13 hours ago History
Name Last commit message Last commit date
images adding new repo 4 days ago
app.js adding new repo 4 days ago
index.html Update index.html 13 hours ago
style.css adding new repo 4 days ago
23:32
21-11-2024

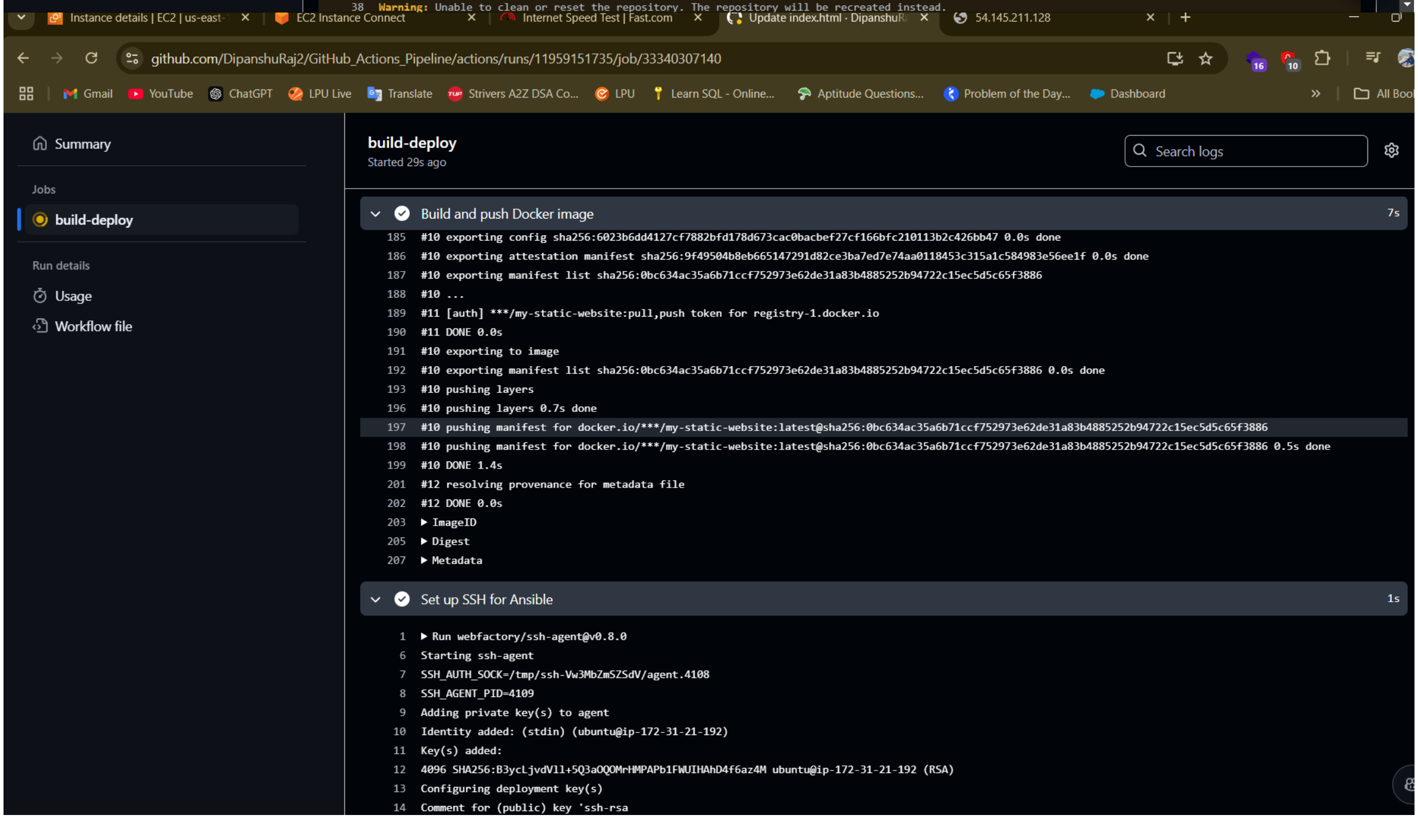


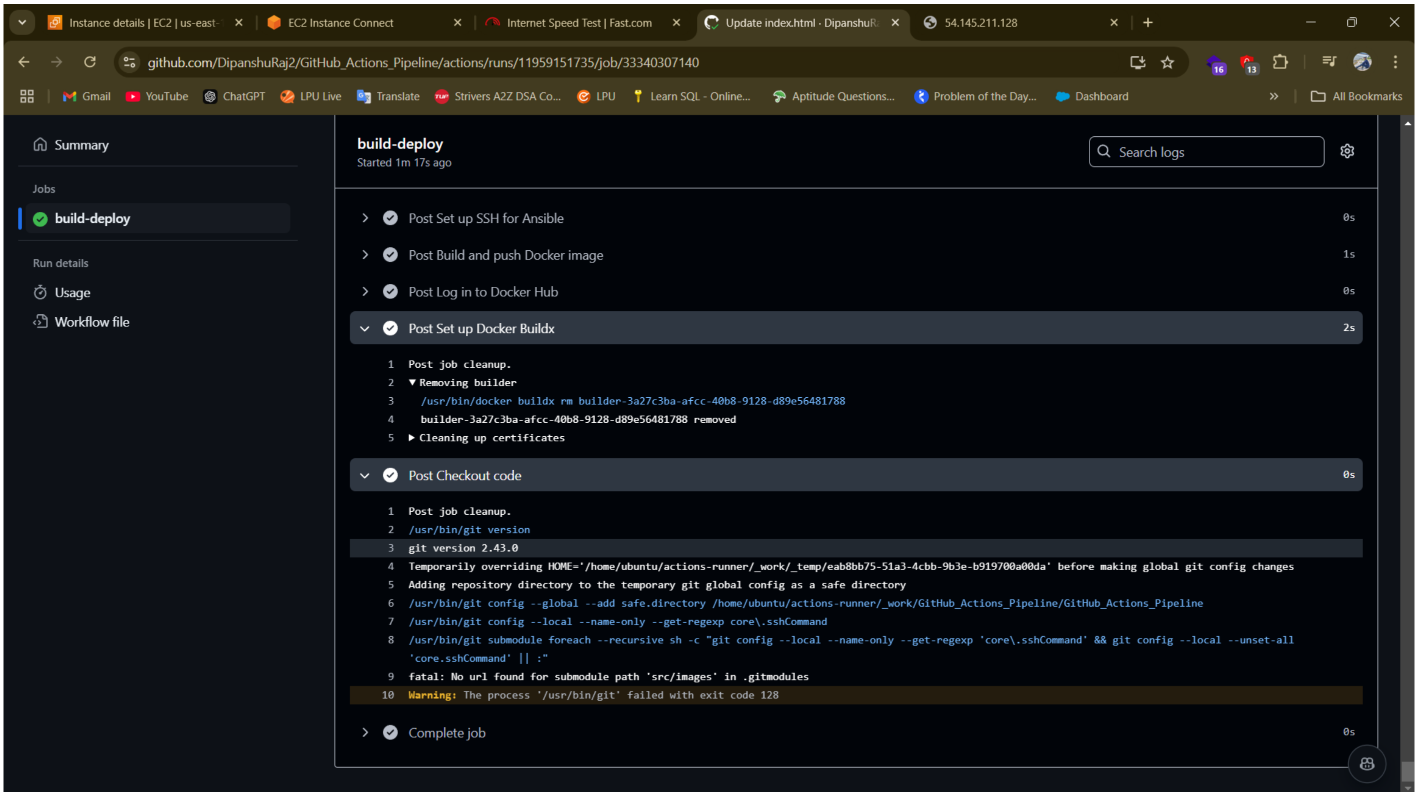
github.com/DipanshuRaj2/GitHub_Actions_Pipeline/edit/main/src/index.html
Files
main
Go to file
.github
ansible
src
images
app.js
index.html
style.css
Dockerfile
Edit
Preview
Spaces
4
No wrap
</div>
</div>
<div class="black"></div>
<header>
<div class="first">
<div class="flex logo">
<img src="images/logo.png" alt="">
<div class="map flex">
<i class="fas fa-map-marker"></i>
<div>
Deliver to
Haryana
</div>
</div>
</div>
<div class="flex input">
<div>
All
<i class="fas fa-caret-down"></i>
</div>
<input type="text">
<i class="fas fa-search"></i>
</div>
<div class="flex right">
<div class="flex lang">
<img src="images/usflag.jpg" alt="">
<i class="fas fa-caret-down"></i>
</div>
<div class="sign">
Use Control + Shift + m to toggle the tab key moving focus. Alternatively, use esc then tab to move to the next interactive element on the page.

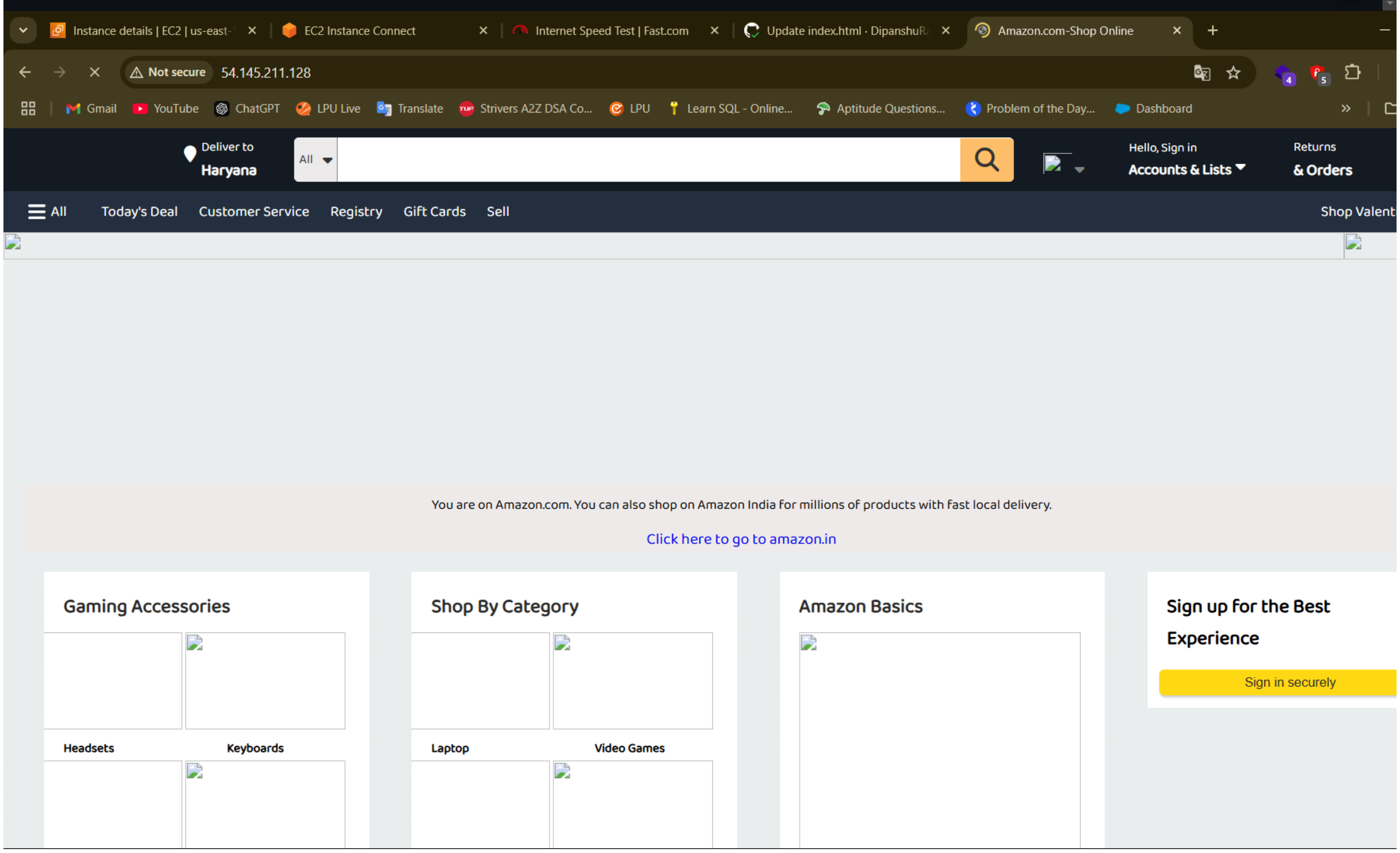


github.com/DipanshuRaj2/GitHub_Actions_Pipeline/actions
DipanshuRaj2 / GitHub_Actions_Pipeline
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
Actions
New workflow
All workflows
CI/CD Pipeline for Static Website
Management
Caches
Attestations
Runners
All workflows
Showing runs from all workflows
Filter workflow runs
12 workflow runs
Event
Status
Branch
Actor
Update index.html
CI/CD Pipeline for Static Website #12: Commit 72922bf pushed by DipanshuRaj2
main
now
Queued
Update index.html
CI/CD Pipeline for Static Website #11: Commit 81e0384 pushed by DipanshuRaj2
main
13 hours ago
1m 40s
Update index.html
CI/CD Pipeline for Static Website #10: Commit b2f7c30 pushed by DipanshuRaj2
main
13 hours ago
1m 17s
Update index.html
CI/CD Pipeline for Static Website #9: Commit 2c34c4e pushed by DipanshuRaj2
main
yesterday
1m 44s
Update index.html
CI/CD Pipeline for Static Website #8: Commit 40700d9 pushed by DipanshuRaj2
main
2 days ago
1m 47s
Update index.html
CI/CD Pipeline for Static Website #7: Commit 707e0f0 pushed by DipanshuRaj2
main
2 days ago
1m 42s
https://github.com/DipanshuRaj2/GitHub_Actions_Pipeline/actions/runs/11928318115


DipanshuRaj2 / GitHub_Actions_Pipeline
Code
Issues
Pull requests
Actions
Projects
Wiki
Security
Insights
Settings
CI/CD Pipeline for Static Website
Update index.html #12
Cancel workflow
Summary
build-deploy
Usage
Workflow file
build-deploy
Started 18s ago
Search logs
Set up job 3s
Initial system setup 9s
Setup Docker permissions 2s
Checkout code 4s
1 ▶ Run actions/checkout@v3
14 Syncing repository: DipanshuRaj2/GitHub_Actions_Pipeline
15 ▶ Getting Git version info
19 Temporarily overriding HOME='/home/ubuntu/actions-runner/_work/_temp/aac92c17-f603-428b-a4c7-0bfdb0ab7b6f' before making global git config changes
20 Adding repository directory to the temporary git global config as a safe directory
21 /usr/bin/git config --global --add safe.directory /home/ubuntu/actions-runner/_work/GitHub_Actions_Pipeline/GitHub_Actions_Pipeline
22 /usr/bin/git config --local --get remote.origin.url
23 https://github.com/DipanshuRaj2/GitHub_Actions_Pipeline
24 ▶ Removing previously created refs, to avoid conflicts
31 /usr/bin/git submodule status
32 Error: fatal: no submodule mapping found in .gitmodules for path 'src/images'
33 Bad Submodules found, removing existing files
34 ▶ Cleaning the repository
38 Warning: Unable to clean or reset the repository. The repository will be recreated instead.


github.com/DipanshuRaj2/GitHub_Actions_Pipeline/actions/runs/11959151735/job/33340307140
Summary
Jobs
build-deploy
Run details
Usage
Workflow file
build-deploy
Started 29s ago
Search logs
Build and push Docker image 7s
185 #10 exporting config sha256:6023b6dd4127cf7882bfd178d673cac0bacbef27cf166bfc210113b2c426bb47 0.0s done
186 #10 exporting attestation manifest sha256:9f49504b8eb665147291d82ce3ba7ed7e74aa0118453c315a1c584983e56ee1f 0.0s done
187 #10 exporting manifest list sha256:0bc634ac35a6b71ccf752973e62de31a83b4885252b94722c15ec5d5c65f3886
188 #10 ...
189 #11 [auth] ***/my-static-website:pull,push token for registry-1.docker.io
190 #11 DONE 0.0s
191 #10 exporting to image
192 #10 exporting manifest list sha256:0bc634ac35a6b71ccf752973e62de31a83b4885252b94722c15ec5d5c65f3886 0.0s done
193 #10 pushing layers
196 #10 pushing layers 0.7s done
197 #10 pushing manifest for docker.io/***/my-static-website:latest@sha256:0bc634ac35a6b71ccf752973e62de31a83b4885252b94722c15ec5d5c65f3886
198 #10 pushing manifest for docker.io/***/my-static-website:latest@sha256:0bc634ac35a6b71ccf752973e62de31a83b4885252b94722c15ec5d5c65f3886 0.5s done
199 #10 DONE 1.4s
201 #12 resolving provenance for metadata file
202 #12 DONE 0.0s
203 ▶ ImageID
205 ▶ Digest
207 ▶ Metadata
Set up SSH for Ansible 1s
1 ▶ Run webfactory/ssh-agent@v0.8.0
6 Starting ssh-agent
7 SSH_AUTH_SOCK=/tmp/ssh-Vw3MbZmSZSdV/agent.4108
8 SSH_AGENT_PID=4109
9 Adding private key(s) to agent
10 Identity added: (stdin) (ubuntu@ip-172-31-21-192)
11 Key(s) added:
12 4096 SHA256:B3ycLjvdV1l+5Q3aOQOMrHMPAPb1FWUIHAhD4f6az4M ubuntu@ip-172-31-21-192 (RSA)
13 Configuring deployment key(s)
14 Comment for (public) key 'ssh-rsa


Instance details | EC2 | us-east-
EC2 Instance Connect
Internet Speed Test | Fast.com
Update index.html · DipanshuR
54.145.211.128
github.com/DipanshuRaj2/GitHub_Actions_Pipeline/actions/runs/11959151735/job/33340307140
Gmail YouTube ChatGPT LPU Live Translate Strivers A2Z DSA Co... LPU Learn SQL - Online... Aptitude Questions... Problem of the Day... Dashboard All Bookmarks
Summary
Jobs
build-deploy
Run details
Usage
Workflow file
build-deploy
Started 1m 17s ago
Search logs
Post Set up SSH for Ansible 0s
Post Build and push Docker image 1s
Post Log in to Docker Hub 0s
Post Set up Docker Buildx 2s
1 Post job cleanup.
2 ▼ Removing builder
3 /usr/bin/docker buildx rm builder-3a27c3ba-afcc-40b8-9128-d89e56481788
4 builder-3a27c3ba-afcc-40b8-9128-d89e56481788 removed
5 ▶ Cleaning up certificates
Post Checkout code 0s
1 Post job cleanup.
2 /usr/bin/git version
3 git version 2.43.0
4 Temporarily overriding HOME='/home/ubuntu/actions-runner/_work/_temp/eab8bb75-51a3-4cbb-9b3e-b919700a00da' before making global git config changes
5 Adding repository directory to the temporary git global config as a safe directory
6 /usr/bin/git config --global --add safe.directory /home/ubuntu/actions-runner/_work/GitHub_Actions_Pipeline/GitHub_Actions_Pipeline
7 /usr/bin/git config --local --name-only --get-regexp core\.sshCommand
8 /usr/bin/git submodule foreach --recursive sh -c "git config --local --name-only --get-regexp 'core\.sshCommand' && git config --local --unset-all 'core.sshCommand' || :"
9 fatal: No url found for submodule path 'src/images' in .gitmodules
10 Warning: The process '/usr/bin/git' failed with exit code 128
Complete job 0s


Instance details | EC2 | us-east-
EC2 Instance Connect
Internet Speed Test | Fast.com
Update index.html · DipanshuR
Amazon.com-Shop Online
Not secure 54.145.211.128
Gmail YouTube ChatGPT LPU Live Translate Strivers A2Z DSA Co... LPU Learn SQL - Online... Aptitude Questions... Problem of the Day... Dashboard
Deliver to
Haryana
All
Hello, Sign in
Accounts & Lists
Returns
& Orders
All Today's Deal Customer Service Registry Gift Cards Sell
Shop Valent
You are on Amazon.com. You can also shop on Amazon India for millions of products with fast local delivery.
Click here to go to amazon.in
Gaming Accessories
Headsets
Keyboards
Shop By Category
Laptop
Video Games
Amazon Basics
Sign up for the Best Experience
Sign in securely