

Akilandeshwari Srinivasan(451036)

Lab 4 – CLCM3404

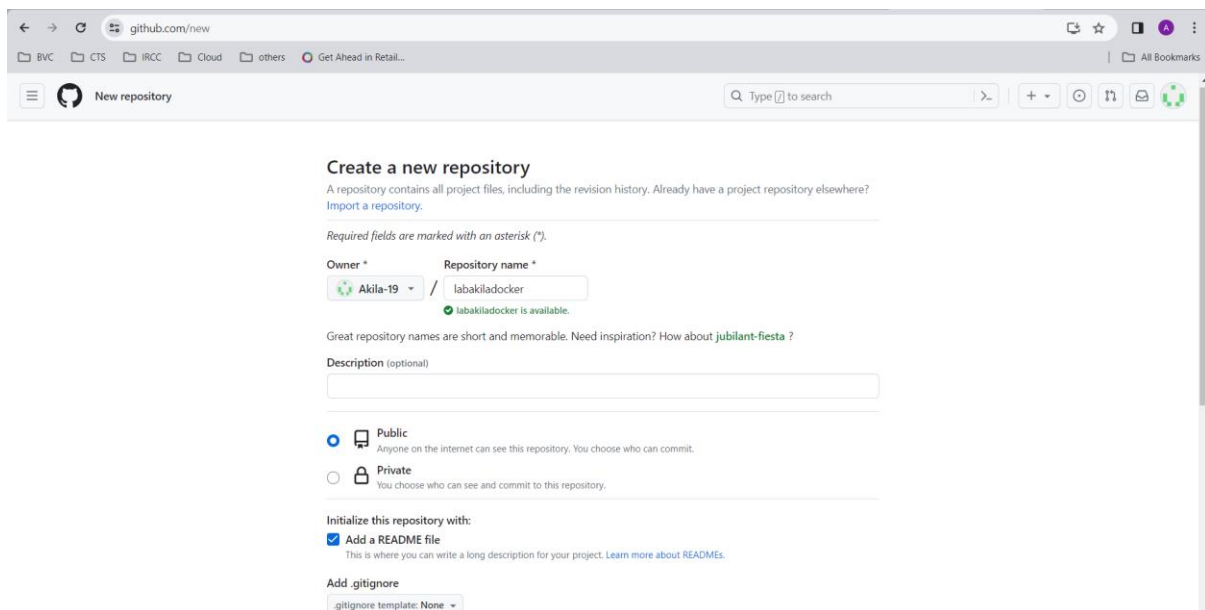
Due by : 09-12-2023

Preparation:

In this lab the task given is deploy an website on AWS EC2 using Docker and implement CI/CD.

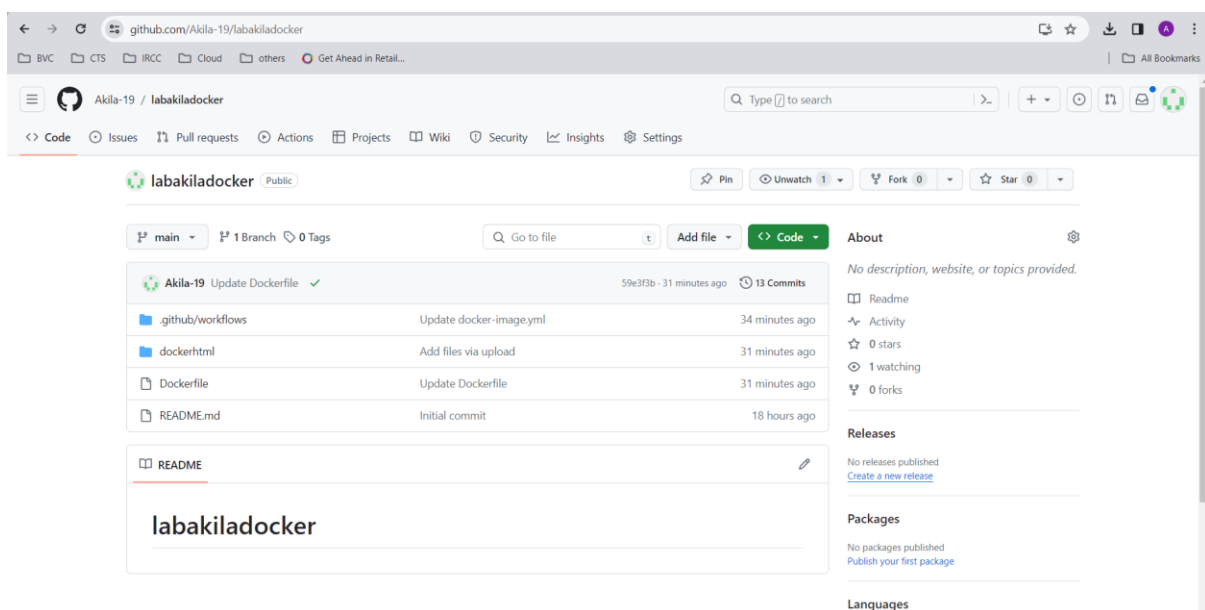
Screenshots:

Create a repository and include README file. Also make the file as public.



The screenshot shows the GitHub 'Create a new repository' page. The 'Repository name' field is filled with 'labakiladocker' and is marked as available. The 'Owner' is 'Akila-19'. The 'Description' field is empty. The 'Public' radio button is selected, indicating the repository will be public. The 'Add a README file' checkbox is checked. The 'Initialize this repository with' section shows 'Add a README file' as the selected option. The 'Add .gitignore' section shows '.gitignore template: None'.

Create a folder and add the index.html. Also create a Dockerfile and add the respective commands to it.

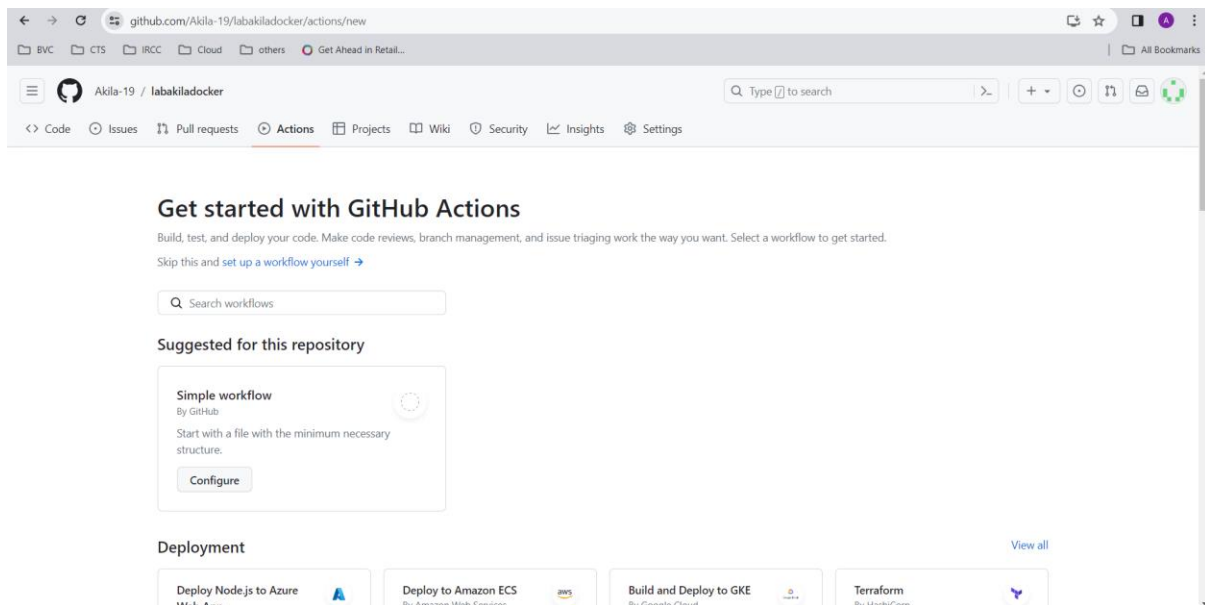


The screenshot shows the GitHub repository page for 'labakiladocker'. The repository is public and has 13 commits. The commit history shows the following files and their last commit times:

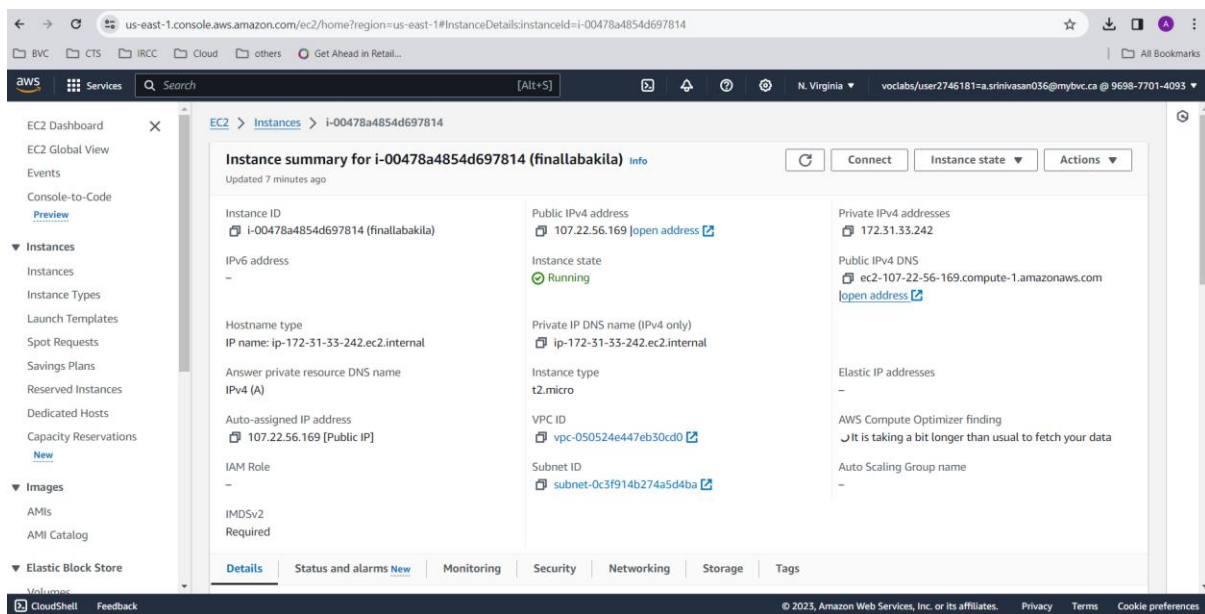
File	Commit Message	Time Ago
.github/workflows	Update docker-image.yml	34 minutes ago
dockerhtml	Add files via upload	31 minutes ago
Dockerfile	Update Dockerfile	31 minutes ago
README.md	Initial commit	18 hours ago

The README file is visible and contains the text 'labakiladocker'.

Create the workflow by going to the Actions->search for docker->configure



Parallely create the EC2 manually in the AWS academy. Once it is created, the connect it






















Post that create the repository secrets which we mentioned in the code.

These are the variables we are using in the yaml file.

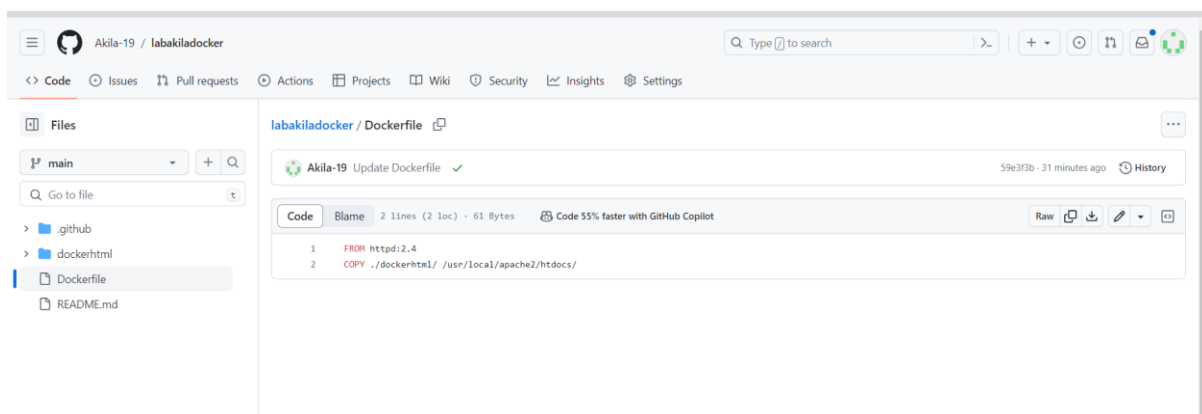
- DOCKER_PASSWORD → your docker desktop password
- DOCKER_USERNAME → your docker desktop username
- EC2_SSH_KEY → The .pem file contents which is downloaded while creating EC2
- HOST_DNS → the public DNS IP address
- TARGET_DIR → home(destination directory)
- USERNAME → ec2-user

Repository secrets

[New repository secret](#)

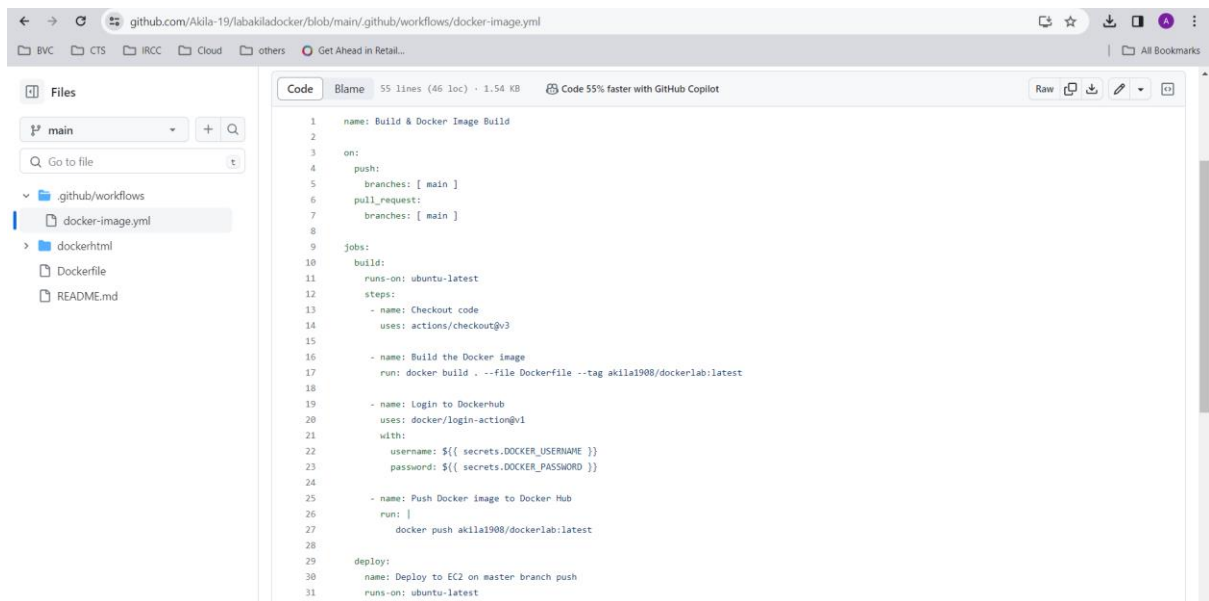
Name 	Last updated	
 DOCKER_PASSWORD	3 hours ago	 
 DOCKER_USERNAME	3 hours ago	 
 EC2_SSH_KEY	3 hours ago	 
 HOST_DNS	3 hours ago	 
 TARGET_DIR	3 hours ago	 
 USERNAME	3 hours ago	 

In the Dockerfile add these two lines. Httpd is the docker image and copy that from dockerhtml folder to the htdocs directory.



The screenshot shows a GitHub repository for 'Akila-19 / labakiladocker'. The 'Files' sidebar on the left shows the directory structure: main, .github, dockerhtml, Dockerfile, and README.md. The 'Dockerfile' file is selected and its content is displayed in the main area. The Dockerfile contains two lines: 'FROM httpd:2.4' and 'COPY ./dockerhtml/ /usr/local/apache2/htdocs/'. Above the code, a commit message 'Akila-19 Update Dockerfile' is visible, along with a green checkmark indicating a successful update. The commit hash '59e3f3b' and the time '31 minutes ago' are also shown. The 'Code' tab is active, and the file is 61 bytes in size.

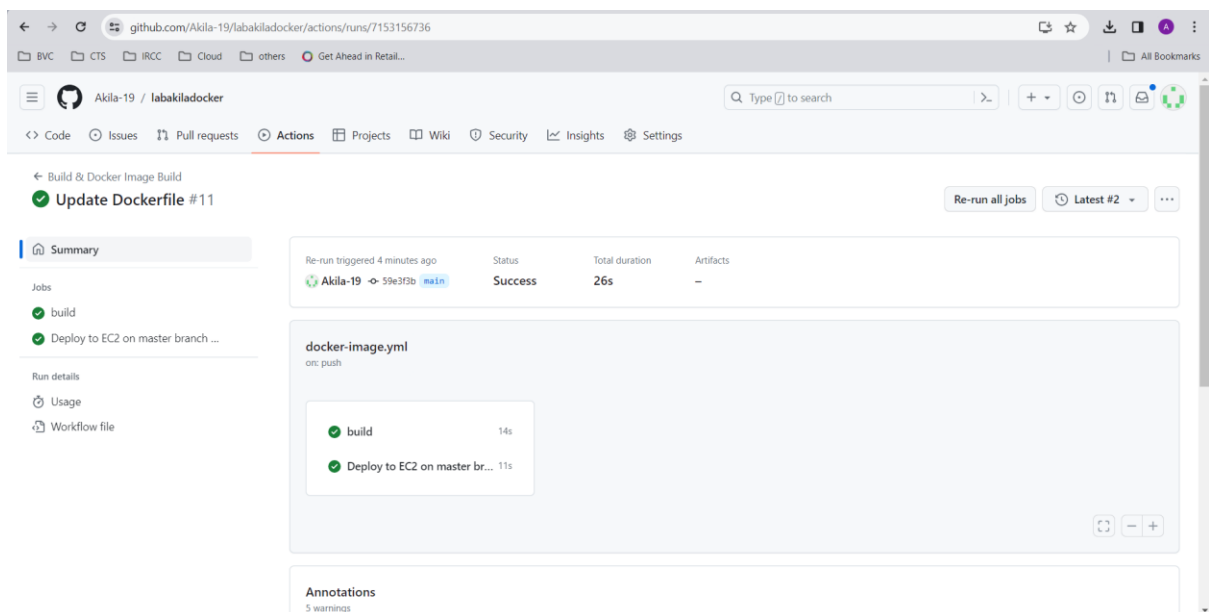
Now add the contents in the .yaml files and commits the changes



The screenshot shows a GitHub repository page for 'Akila-19/labakiladocker'. The file 'docker-image.yml' is selected in the 'Files' sidebar. The main content area displays the YAML code for the workflow. The workflow is named 'Build & Docker Image Build' and is triggered on 'push' to the 'main' branch. It consists of three jobs: 'build', 'login', and 'push'. The 'build' job uses 'ubuntu-latest' and runs 'docker build'. The 'login' job uses 'docker/login-action@v1' and sets environment variables for 'DOCKER_USERNAME' and 'DOCKER_PASSWORD'. The 'push' job uses 'docker/push-action@v1' and pushes the image to 'akila1908/dockerlab:latest'.

```
1 name: Build & Docker Image Build
2
3 on:
4   push:
5     branches: [ main ]
6   pull_request:
7     branches: [ main ]
8
9 jobs:
10  build:
11    runs-on: ubuntu-latest
12    steps:
13      - name: Checkout code
14        uses: actions/checkout@v3
15
16      - name: Build the Docker image
17        run: docker build . --file Dockerfile --tag akila1908/dockerlab:latest
18
19      - name: Login to Dockerhub
20        uses: docker/login-action@v1
21        with:
22          username: ${{ secrets.DOCKER_USERNAME }}
23          password: ${{ secrets.DOCKER_PASSWORD }}
24
25      - name: Push Docker image to Docker Hub
26        run: |
27          docker push akila1908/dockerlab:latest
28
29  deploy:
30    name: Deploy to EC2 on master branch push
31    runs-on: ubuntu-latest
```

Check whether it is deployed properly or not in the tab actions and corresponding yaml files.



The screenshot shows the GitHub Actions page for the 'Update Dockerfile #11' workflow. The workflow is in a 'Success' state, triggered by a push to the 'main' branch. The page shows a summary of the workflow, including the jobs 'build' and 'Deploy to EC2 on master branch ...'. The 'build' job took 14s and the 'Deploy to EC2 on master branch ...' job took 11s. The page also shows the 'docker-image.yml' file and the 'Annotations' section with 5 warnings.

Build & Docker Image Build

Update Dockerfile #11

Re-run triggered 4 minutes ago

Job	Status	Total duration	Artifacts
Akila-19 - 59e3f3b - main	Success	26s	-

docker-image.yml

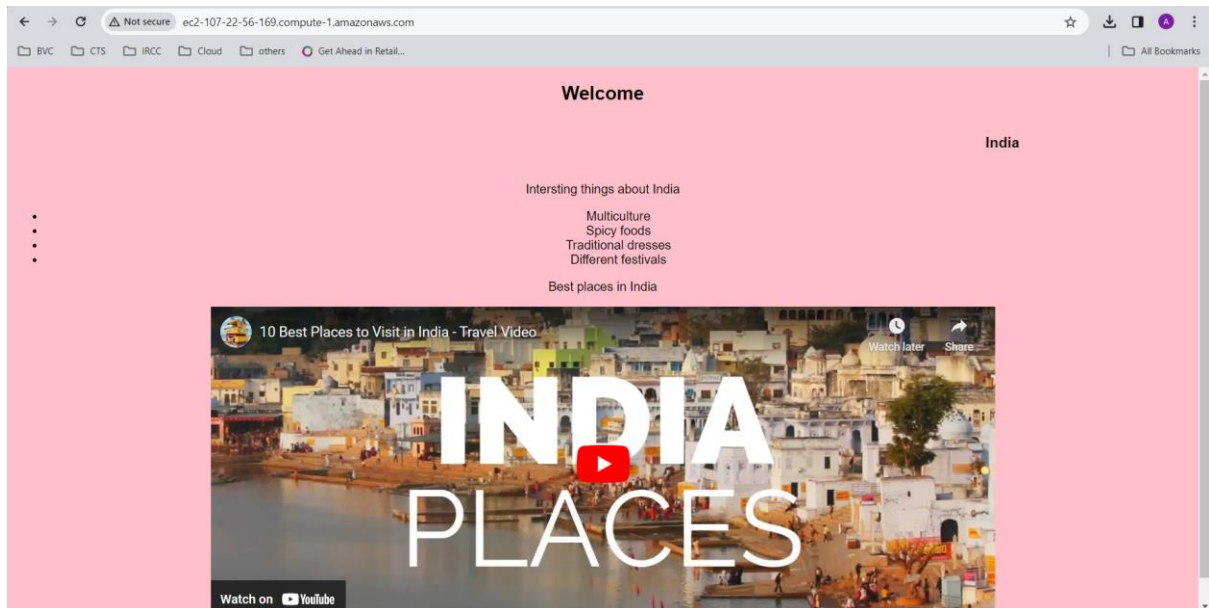
on: push

- build 14s
- Deploy to EC2 on master br... 11s

Annotations

5 warnings

Once its deployed successfully, then go the respective EC2 and click the ip address where you can see the deployed html here.



Reflections:

When I started the lab my deployment is failed due to some errors in yml file. Then I have referred the yml file which is in the d2l then it worked fine.