Experiment 4

Docker Build and Push using GitHub Actions

Objective: Set up a GitHub Actions workflow to automatically build a Docker image from a Dockerfile in your GitHub repository and push it to a container registry (e.g., Docker Hub).

Prerequisites:

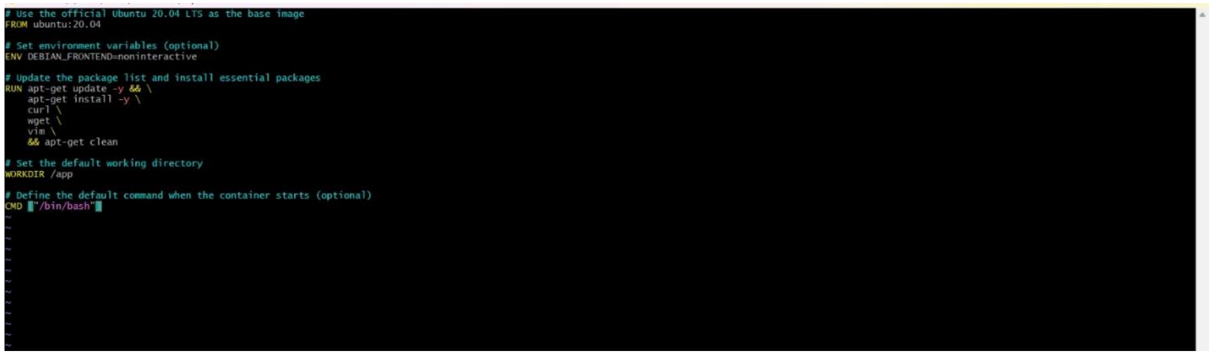
GitHub account

- Docker installed on your local machine
- A Dockerfile in your GitHub repository
- A Docker Hub account (or any other container registry)

Exercise Steps:

Step 1: Fork and Clone the Repository

- Fork a sample GitHub repository containing a Dockerfile or create a new repository and add a Dockerfile to it.
- Clone the forked repository to your local machine.

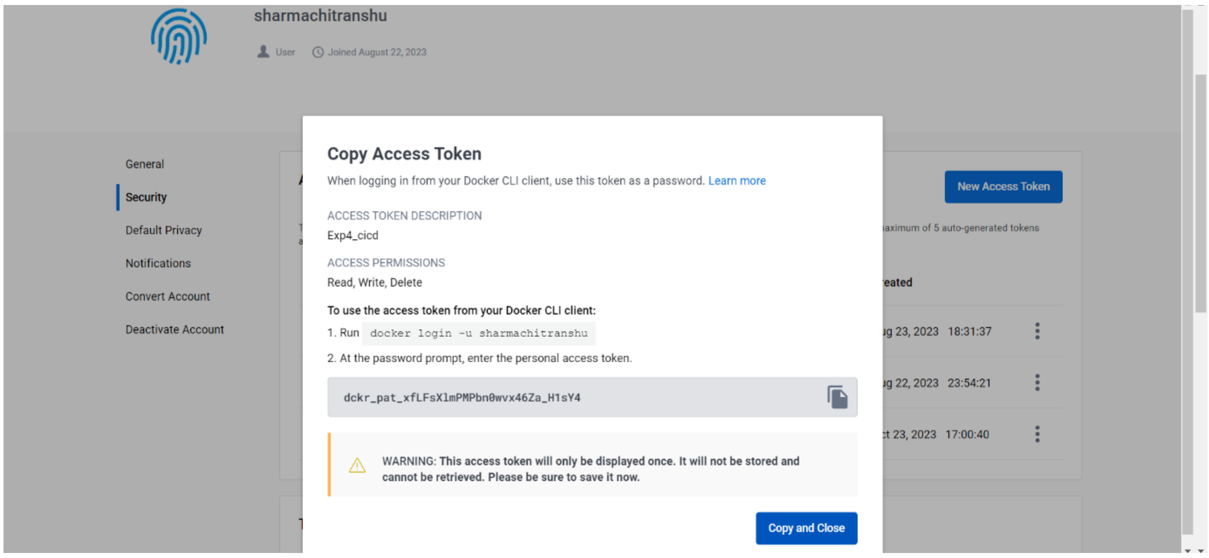


```
# Use the official Ubuntu 20.04 LTS as the base image
FROM ubuntu:20.04

# Set environment variables (optional)
ENV DEBIAN_FRONTEND=noninteractive

# Update the package list and install essential packages
RUN apt-get update -y && \
    apt-get install -y \
    curl \
    wget \
    vim \
    && apt-get clean

# Set the default working directory
WORKDIR /app

# Define the default command when the container starts (optional)
CMD ["/bin/bash"]
~
~
~
~
~
~
~
~
~
~
```

Step 2: Create Docker Hub Access Token

- Log in to your Docker Hub account.
- Go to your account settings and click on the "Security" tab.
- Under "Access Tokens," click "New Access Token." Give it a name, select the required permissions (e.g., "Write" for pushing Docker images), and click "Create."

- Copy the generated access token. You will need it to authenticate with Docker Hub in your GitHub Actions workflow.



sharmachitranshu

User Joined August 22, 2023

General
Security
Default Privacy
Notifications
Convert Account
Deactivate Account

Copy Access Token

When logging in from your Docker CLI client, use this token as a password. Learn more

ACCESS TOKEN DESCRIPTION
Exp4_cicd

ACCESS PERMISSIONS
Read, Write, Delete

To use the access token from your Docker CLI client:

1. Run `docker login -u sharmachitranshu`
2. At the password prompt, enter the personal access token.

```
dckr_pat_xfLFsXlmPMPbn0wvx46Za_H1sY4
```

WARNING: This access token will only be displayed once. It will not be stored and cannot be retrieved. Please be sure to save it now.

Step 3: Create a GitHub Actions Workflow

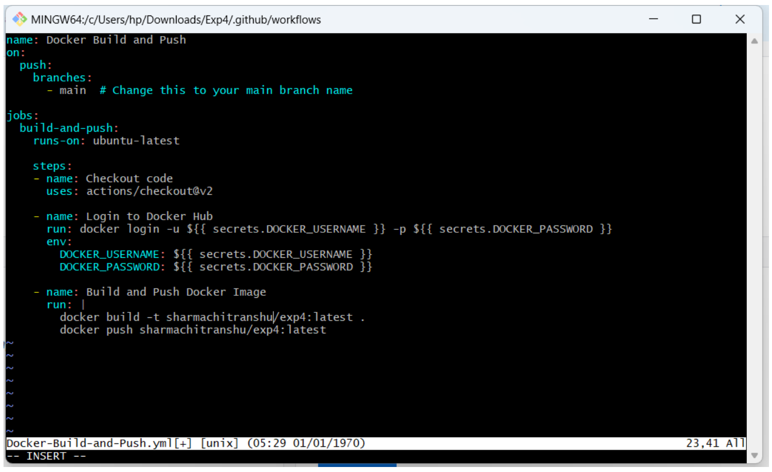
- In your cloned repository, create a directory named .github/workflows if it doesn't exist.

- Inside the .github/workflows directory, create a YAML file (e.g., docker-build-and-push.yml) to define your GitHub Actions workflow. You can use any text editor to create the file.
- Edit docker-build-and-push.yml and add the following content:

```
name: Docker Build and Push

on:

 push:

  branches:

   - main  # Change this to your main branch name


jobs:

 build-and-push:

  runs-on: ubuntu-latest


  steps:

  - name: Checkout code

   uses: actions/checkout@v2
```

```
- name: Login to Docker Hub
  run: docker login -u ${{ secrets.DOCKER_USERNAME }} -p ${{ secrets.DOCKER_PASSWORD }}
  env:
    DOCKER_USERNAME: ${{ secrets.DOCKER_USERNAME }}
    DOCKER_PASSWORD: ${{ secrets.DOCKER_PASSWORD }}

- name: Build and Push Docker Image
  run: |
    docker build -t your-dockerhub-username/your-repo-name:latest .
    docker push your-dockerhub-username/your-repo-name:latest
```

Replace your-dockerhub-username and your-repo-name with your Docker Hub username and repository name.

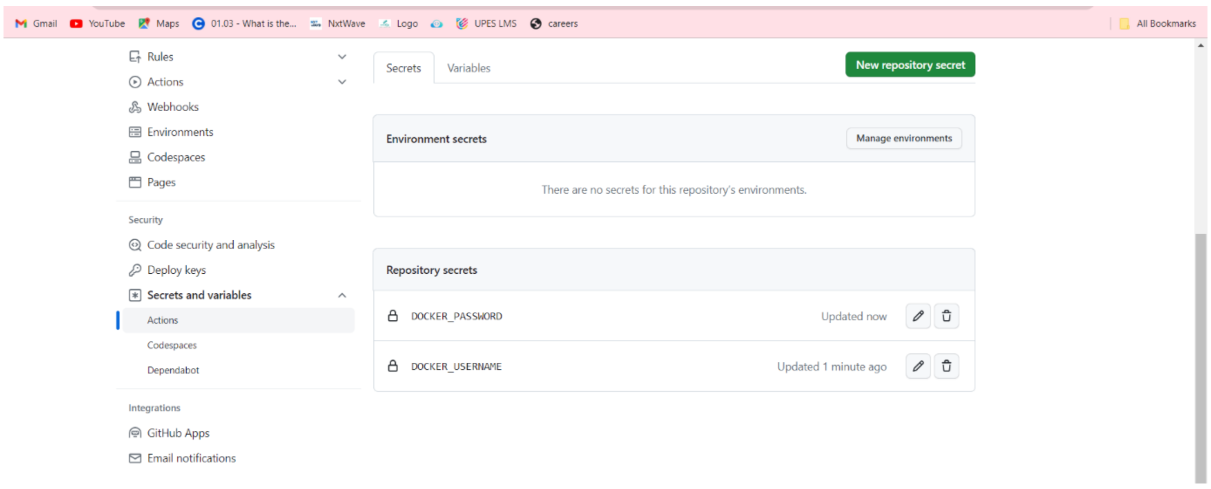


MINGW64:/c/Users/hp/Downloads/Exp4/.github/workflows

```
name: Docker Build and Push
on:
  push:
    branches:
      - main  # Change this to your main branch name

jobs:
  build-and-push:
    runs-on: ubuntu-latest

    steps:
    - name: Checkout code
      uses: actions/checkout@v2

    - name: Login to Docker Hub
      run: docker login -u ${{ secrets.DOCKER_USERNAME }} -p ${{ secrets.DOCKER_PASSWORD }}
      env:
        DOCKER_USERNAME: ${{ secrets.DOCKER_USERNAME }}
        DOCKER_PASSWORD: ${{ secrets.DOCKER_PASSWORD }}

    - name: Build and Push Docker Image
      run: |
        docker build -t sharmachitranshu/exp4:latest .
        docker push sharmachitranshu/exp4:latest
~
~
~
~
~
~
~
~
Docker-Build-and-Push.yml[+] [unix] (05:29 01/01/1970)                    23,41 All
-- INSERT --
```

Step 4: Add Docker Hub Credentials to GitHub Secrets

- Go to your GitHub repository on the GitHub website.
- Click on "Settings" and then "Secrets" in the left sidebar.
- Click on "New repository secret" and add two secrets:
- DOCKER_USERNAME: Set this to your Docker Hub username.
- DOCKER_PASSWORD: Set this to the Docker Hub access token you generated earlier.


Secrets | Variables | New repository secret

Environment secrets | Manage environments

There are no secrets for this repository's environments.

Repository secrets

DOCKER_PASSWORD	Updated now
DOCKER_USERNAME	Updated 1 minute ago

Step 5: Commit and Push Changes

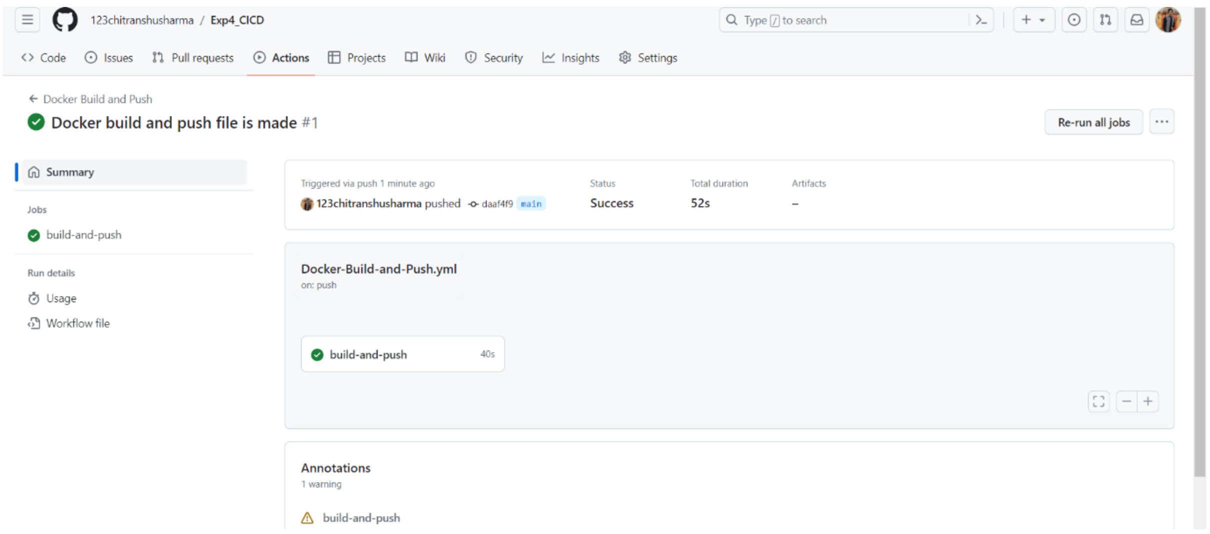
Save the docker-build-and-push.yml file.

Commit the changes to your local repository:

```
git add .
git commit -m "Add GitHub Actions workflow for Docker build and push"
git push origin main
```

Step 6: Check the Workflow Status

- Go to your GitHub repository on the GitHub website.
- Click on the "Actions" tab to see the workflow running. You should see a workflow named "Docker Build and Push" or the name you specified in the YAML file.
- Monitor the workflow's progress, and once it completes successfully, you should see a green checkmark indicating a successful build and push of the Docker image to Docker Hub.


123chitranshusharma / Exp4_CICD

Docker Build and Push

Docker build and push file is made #1

Triggered via push 1 minute ago: 123chitranshusharma pushed daaf4f9 main

Status: Success

Total duration: 52s

Artifacts: –

Docker-Build-and-Push.yml
on: push

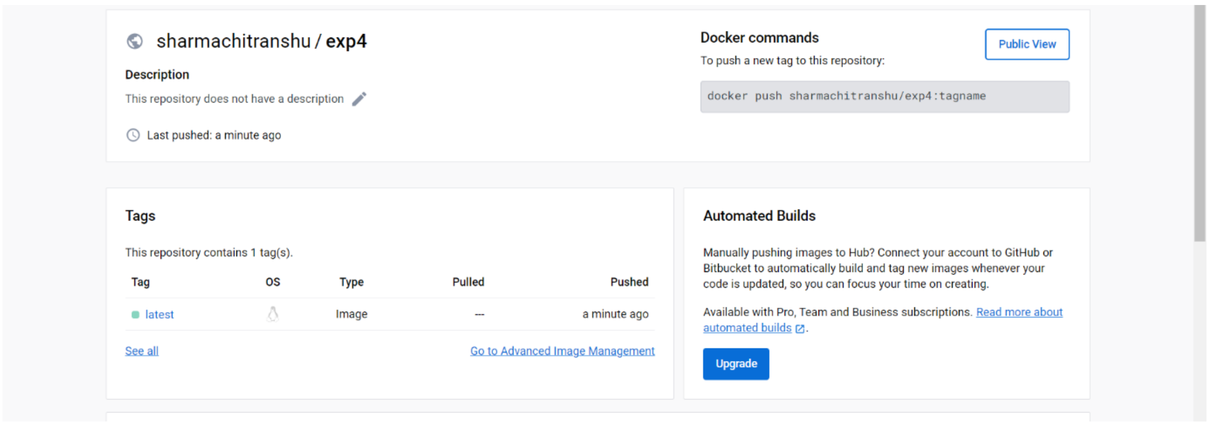
build-and-push 40s

Annotations
1 warning

build-and-push

Step 7: Verify the Docker Image on Docker Hub

- Log in to your Docker Hub account.
- Navigate to your Docker Hub repository, and you should see the Docker image you pushed from the GitHub Actions workflow.



sharmachitranshu / exp4

Description

This repository does not have a description

Last pushed: a minute ago

Docker commands

To push a new tag to this repository:

```
docker push sharmachitranshu/exp4:tagname
```

Tags

This repository contains 1 tag(s).

Tag	OS	Type	Pulled	Pushed
latest		Image	—	a minute ago

Automated Builds

Manually pushing images to Hub? Connect your account to GitHub or Bitbucket to automatically build and tag new images whenever your code is updated, so you can focus your time on creating.

Available with Pro, Team and Business subscriptions. Read more about automated builds.

Step 8: Optional - Trigger a Build

To test the workflow, make changes to your Dockerfile or application code, commit, and push them to the repository. This should trigger the GitHub Actions workflow automatically.

Conclusion:

In this lab exercise, you've set up a GitHub Actions workflow to build a Docker image from a Dockerfile and push it to Docker Hub. Participants should now have a basic understanding of how to automate Docker image creation and deployment using GitHub Actions. You can extend this exercise by exploring more advanced Docker features or integrating other container registries.