**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:**

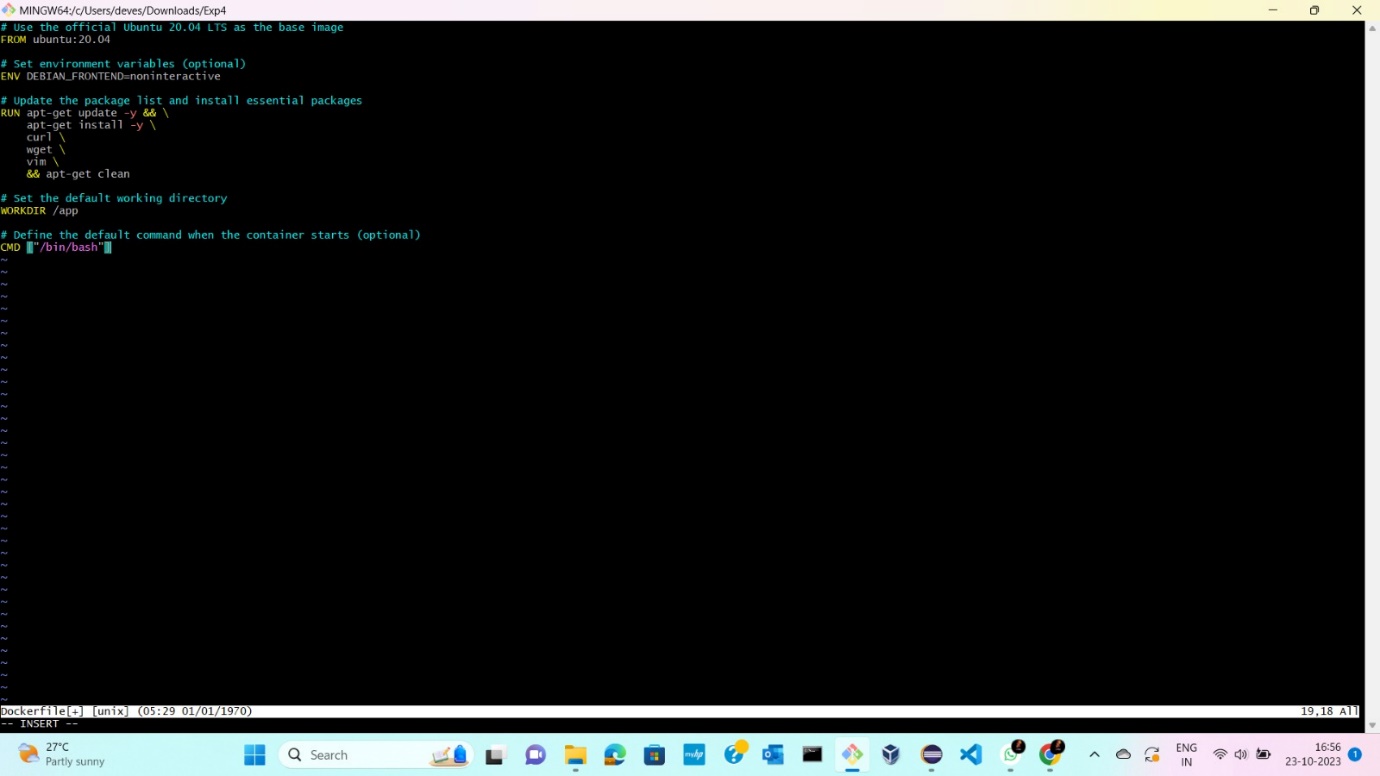
GitHub account

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

**Exercise Steps:**

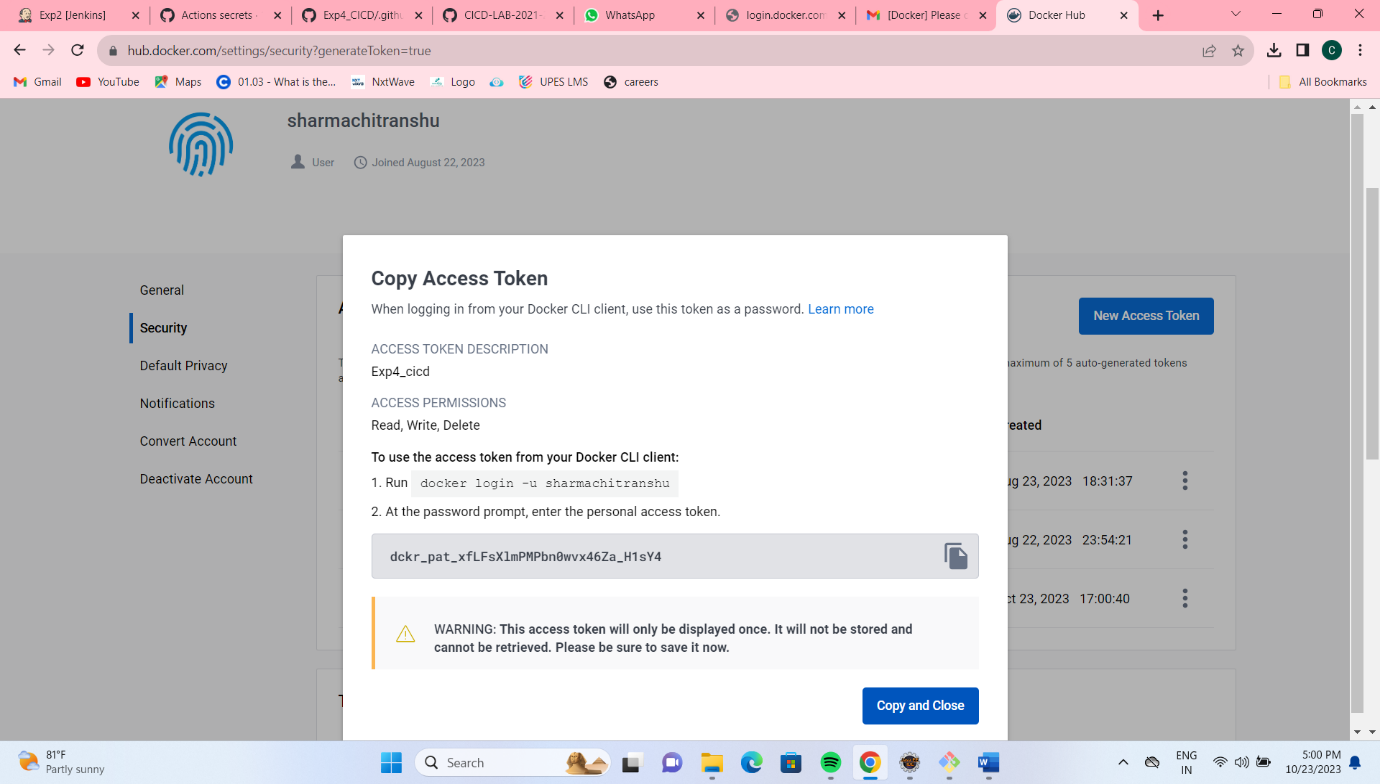
**Step 1: Fork and Clone the Repository**

* 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.



**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.



**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 }}

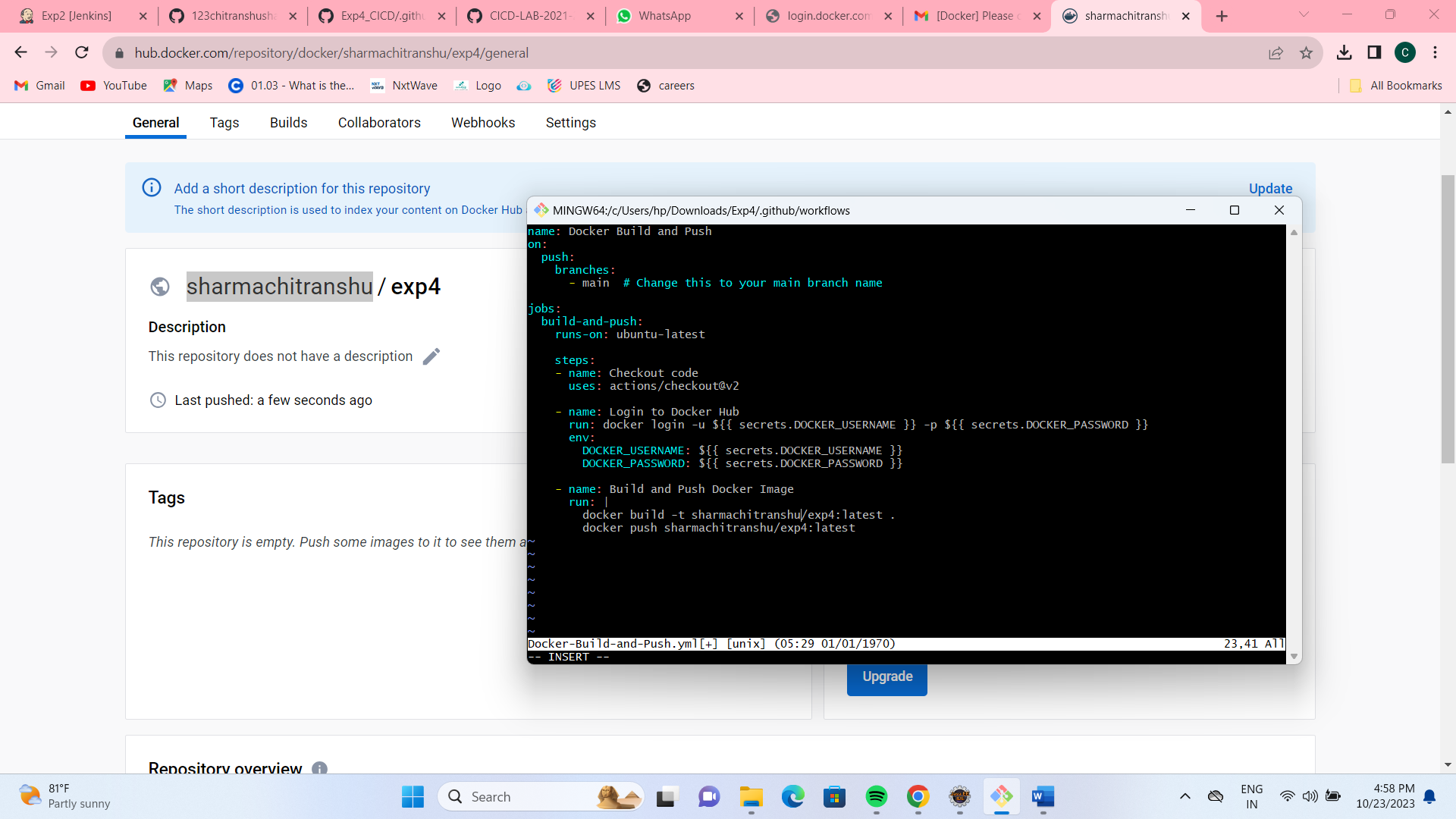
- name: Build and Push Docker Image

run: |

docker build -t your-dockerhub-username/your-repo-name:latest .

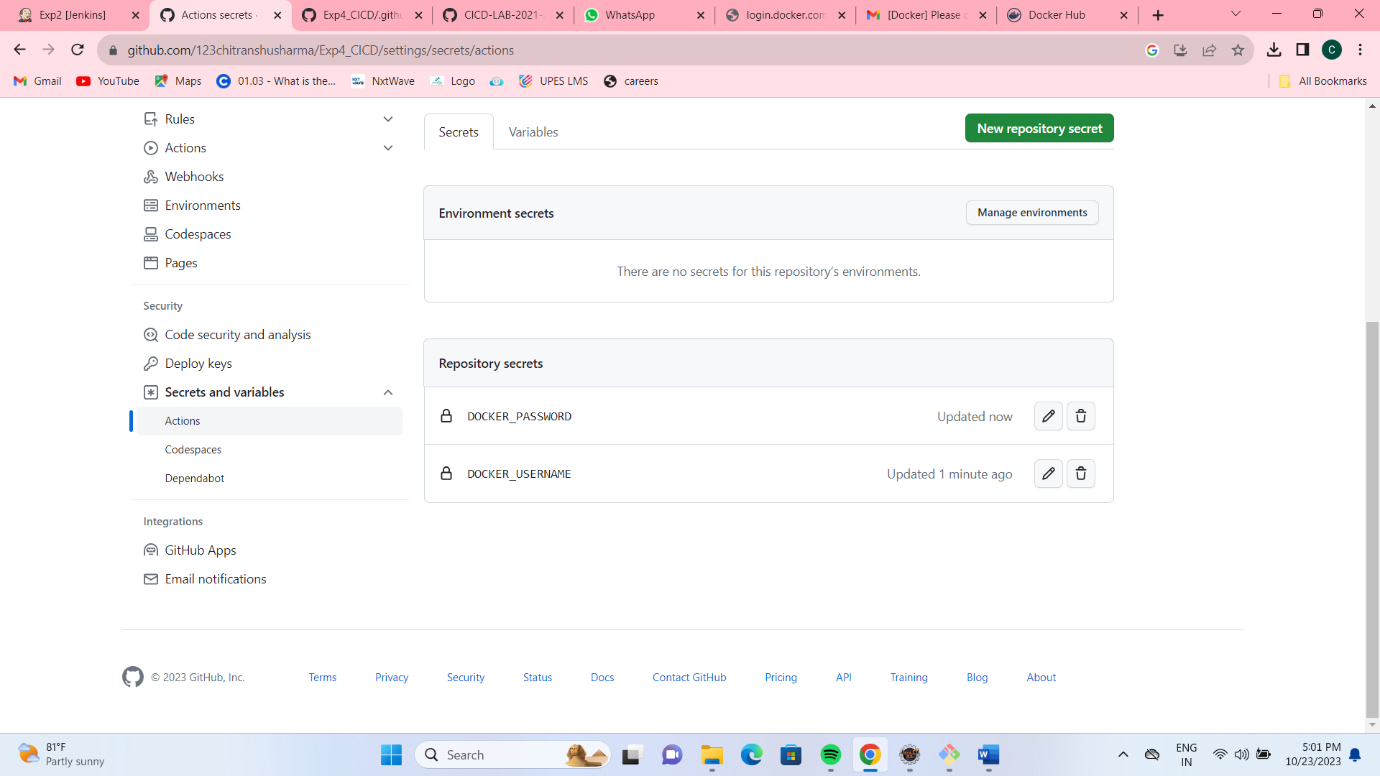
docker push your-dockerhub-username/your-repo-name:latest

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



**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.



**Step 5: Commit and Push Changes**

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

Commit the changes to your local repository:

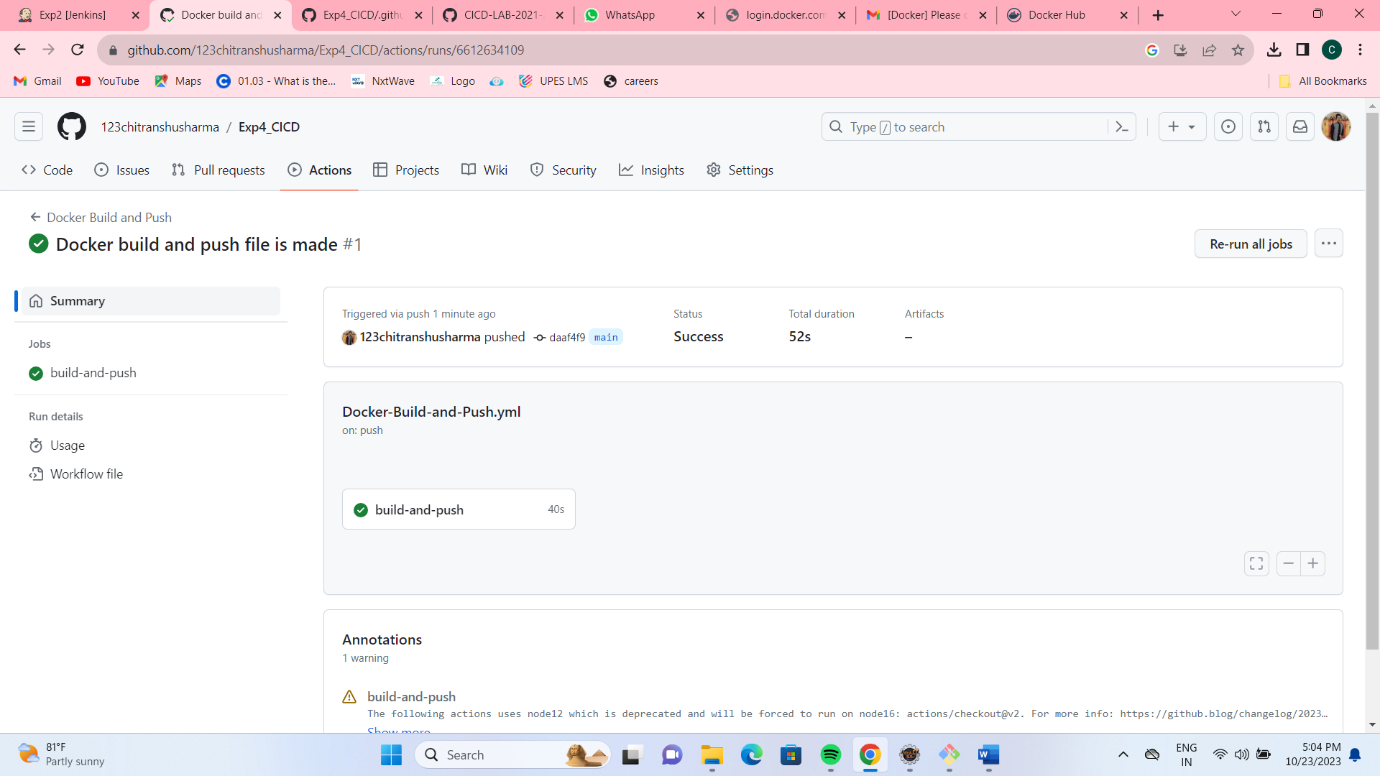
git add .

git commit -m "Add GitHub Actions workflow for Docker build and push"

git push origin main

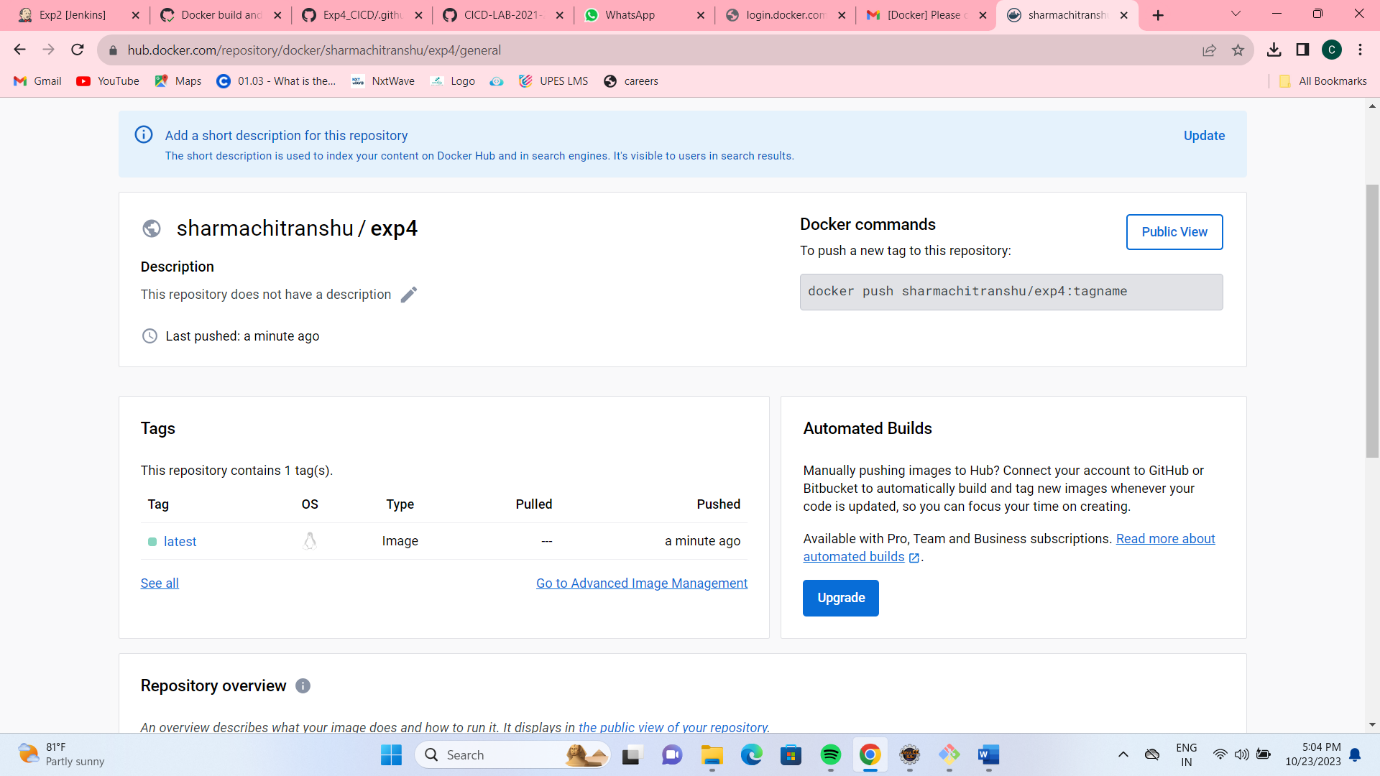
**Step 6: Check the Workflow Status**

* 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.



**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.



**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.