

# Staff Graded Assessment - 2

Name: Yashra Sunilj Kupekar

Student ID: 2023EBCS754

## Section: Git

In this section, I created a local Git repository for my simple web application consisting of three HTML pages: `index.html`, `about.html`, and `contact.html`.

The repository was initialized using `git init`, after which I used commands like `git status`, `git add`, and `git commit` to stage and commit my files.

I explored branching using `git branch` and switched branches with `git switch` to simulate working on different features.

The commit history was viewed using `git log`, which displayed a record of all commits made to the project.

This process demonstrated version control fundamentals and how Git tracks changes efficiently in a project.

```
yashrajkupkar@Yashrajs-MacBook-Air Bits assignmeny % mkdir devops-assignment
yashrajkupkar@Yashrajs-MacBook-Air Bits assignmeny % cd devops-assignment
yashrajkupkar@Yashrajs-MacBook-Air devops-assignment % mkdir src
yashrajkupkar@Yashrajs-MacBook-Air devops-assignment % pwd
/Users/yashrajkupkar/Bits assignmeny/devops-assignment
```

Initialized a new Git repository in the project folder using `git init`. This creates a `.git` folder to track changes.

```
yashrajkupkar@Yashrajs-MacBook-Air devops-assignment % git init
git status
git add src
git status
git commit -m "Initial commit: add web app HTML pages"
Initialized empty Git repository in /Users/yashrajkupkar/Bits assignmeny/devops-assignment/.git/
On branch main

No commits yet

Untracked files:
  (use "git add <file>..." to include in what will be committed)
    src/

nothing added to commit but untracked files present (use "git add" to track)
On branch main

No commits yet

Changes to be committed:
  (use "git rm --cached <file>..." to unstage)
    new file:   src/about.html
    new file:   src/contact.html
    new file:   src/index.html

[main (root-commit) bbd27e3] Initial commit: add web app HTML pages
3 files changed, 27 insertions(+)
create mode 100644 src/about.html
create mode 100644 src/contact.html
create mode 100644 src/index.html
```

```

● yashrajkupekar@Yashrajs-MacBook-Air devops-assignment % git log --oneline
bbd27e3 (HEAD -> main) Initial commit: add web app HTML pages
● yashrajkupekar@Yashrajs-MacBook-Air devops-assignment % git branch
git switch -c feature/add-readme
* main
Switched to a new branch 'feature/add-readme'
● yashrajkupekar@Yashrajs-MacBook-Air devops-assignment % echo "# DevOps Assignment" > README.md
git add README.md
git commit -m "Add README"
git log --oneline
git switch main
[feature/add-readme f18852e] Add README
1 file changed, 1 insertion(+)
create mode 100644 README.md
f18852e (HEAD -> feature/add-readme) Add README
bbd27e3 (main) Initial commit: add web app HTML pages
Switched to branch 'main'

● yashrajkupekar@Yashrajs-MacBook-Air devops-assignment % git remote add origin git@github.com:YashrajKupekar17/devops-assignment.git
git branch -M main
git push -u origin main
Enumerating objects: 6, done.
Counting objects: 100% (6/6), done.
Delta compression using up to 8 threads
Compressing objects: 100% (5/5), done.
Writing objects: 100% (6/6), 876 bytes | 876.00 KiB/s, done.
Total 6 (delta 0), reused 0 (delta 0), pack-reused 0
To github.com:YashrajKupekar17/devops-assignment.git
 * [new branch]      main -> main
branch 'main' set up to track 'origin/main'.

```

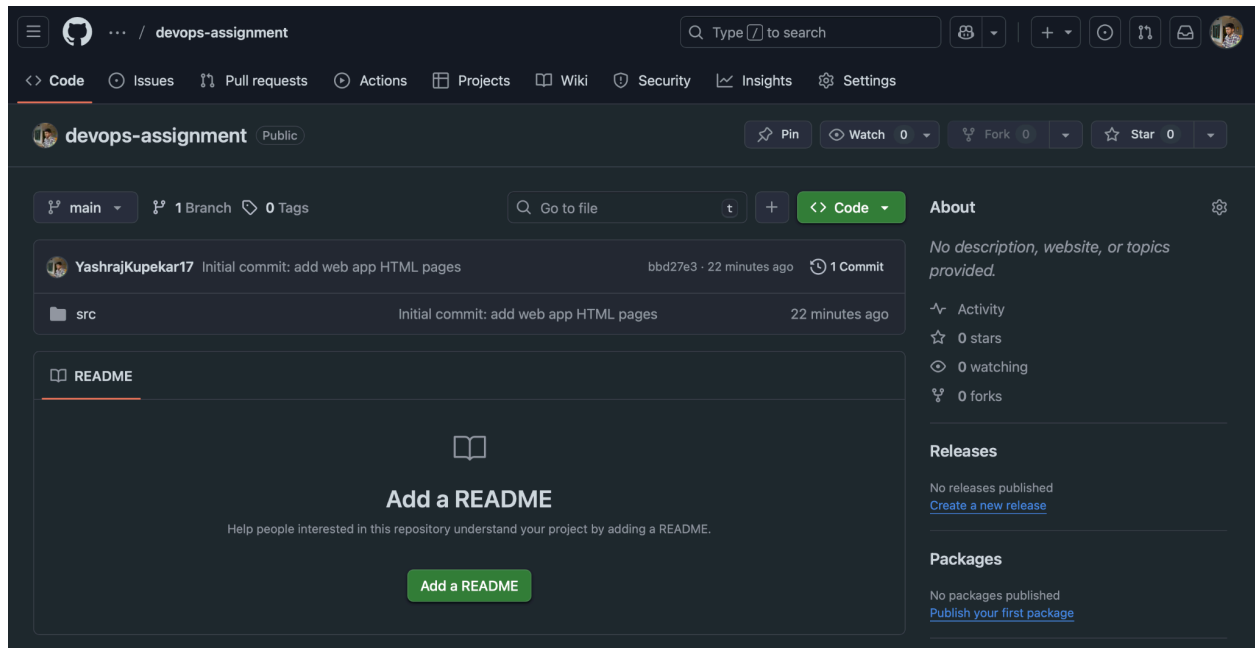
## Github

After setting up the local repository, I created a remote repository on GitHub named **devops-assignment**.

The local repository was linked to GitHub using the command `git remote add origin`, followed by pushing the commits with `git push -u origin main`.

This integration ensured my code was backed up online and version-controlled collaboratively. The project files (HTML pages, Dockerfile, and README) were successfully uploaded to GitHub, providing remote accessibility and version tracking.

GitHub also helps with future collaboration and acts as a central repository for development and deployment pipelines.



## Docker

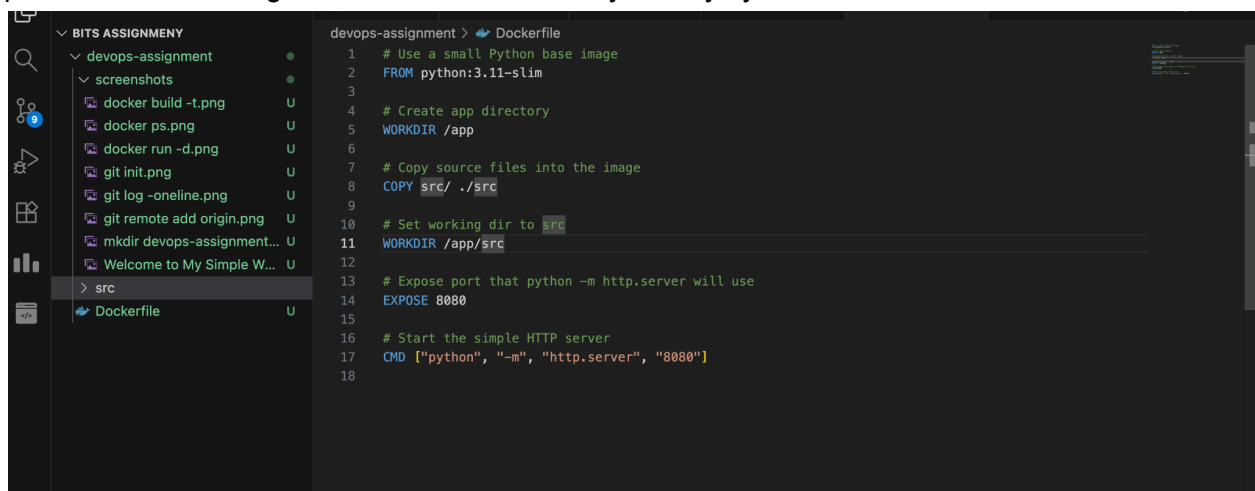
In this section, I containerized the web application using Docker.

I created a **Dockerfile** that uses a lightweight Python base image and serves the HTML files via Python's built-in HTTP server on port 8080.

Using the command `docker build -t yourdockerhubusername/webapp-demo:1.0 .`, I built a custom image from the Dockerfile.

After successful image creation, I logged into Docker Hub using `docker login` and pushed the image using `docker push`.

This step demonstrates containerization—packaging the application and its dependencies into a portable Docker image that can run consistently on any system.



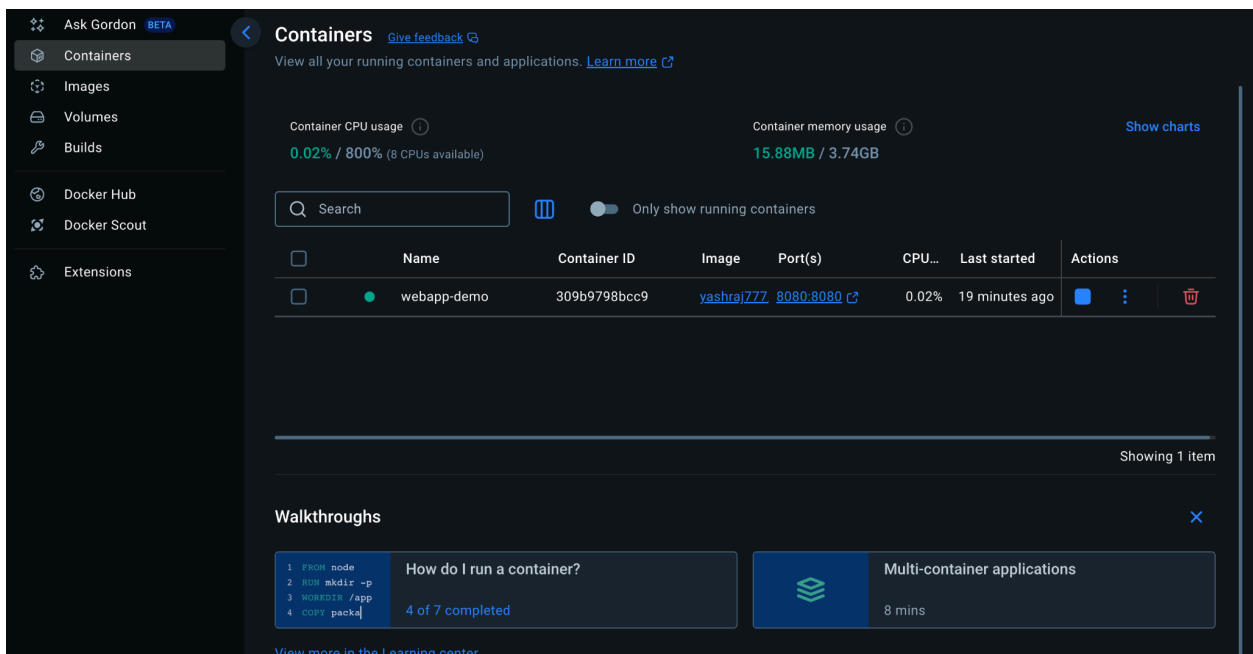
```
yashrajkupekar@Yashrajs-MacBook-Air devops-assignment % docker build -t yashraj777/webapp-demo:1.0 .

[+] Building 2.4s (10/10) FINISHED                                docker:desktop-linux
=> [internal] load build definition from Dockerfile                0.0s
=> => transferring dockerfile: 364B                               0.0s
=> [internal] load metadata for docker.io/library/python:3.11-slim 2.3s
=> [auth] library/python:pull token for registry-1.docker.io      0.0s
=> [internal] load .dockerignore                                  0.0s
=> => transferring context: 2B                                     0.0s
=> [1/4] FROM docker.io/library/python:3.11-slim@sha256:e4676722fba839e2e5cdb844a52262b43e90e56dbd55b7ad953ee36 0.0s
=> => resolve docker.io/library/python:3.11-slim@sha256:e4676722fba839e2e5cdb844a52262b43e90e56dbd55b7ad953ee36 0.0s
=> [internal] load build context                                  0.0s
=> => transferring context: 129B                                    0.0s
=> CACHED [2/4] WORKDIR /app                                       0.0s
=> CACHED [3/4] COPY src/ ./src                                    0.0s
=> CACHED [4/4] WORKDIR /app/src                                   0.0s
=> exporting to image                                             0.0s
=> => exporting layers                                             0.0s
=> => exporting manifest sha256:ffeab335345fbb54b39a8c0cb1ad36bdb3da9d7edfb0ae7acfb81cc6ac7bbc91 0.0s
=> => exporting config sha256:d5a6bda64ead890e6cca37345a7c60862d36eae7804fc9e8f994ee84b5f11e 0.0s
=> => exporting attestation manifest sha256:0d16c8a90ba7a79f37ae1919b3788a3d11d4e7f0c621a20f20949fd0e1af9868 0.0s
=> => exporting manifest list sha256:ee66f7dfd4a27535d90e9517d6278e6924a6976f2bdaa225801b994c11226e7 0.0s
=> => naming to docker.io/yashraj777/webapp-demo:1.0             0.0s
=> => unpacking to docker.io/yashraj777/webapp-demo:1.0          0.0s

yashrajkupekar@Yashrajs-MacBook-Air devops-assignment % docker run -d --name webapp-demo -p 8080:8080 yashraj777/webapp-demo:1.0
309b9798bcc9bb9f59b0f4beda1f317ac9b722bfc3a6dc6e687eea02e1932eff

yashrajkupekar@Yashrajs-MacBook-Air devops-assignment % docker ps
# to view logs (optional)
docker logs webapp-demo

CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS                               NAMES
309b9798bcc9   yashraj777/webapp-demo:1.0         "python -m http.serv..." 2 minutes ago  Up 2 minutes  0.0.0.0:8080->8080/tcp             webapp-demo
zsh: unknown sort specifier
```



## Running the App

Once the Docker image was uploaded to Docker Hub, I ran the container locally using `docker run -d -p 8080:8080 yashraj777/webapp-demo:1.0`. The container started successfully, serving the web application at `http://localhost:8080/`.

Accessing this URL in a browser displayed the homepage, along with navigation links to the About and Contact pages.

This validated that the Docker image was functional and the container replicated the environment reliably.

Running containers locally also demonstrated the efficiency and consistency of containerized deployments.

## Welcome to My Simple Web App

This is the Home page.

[About](#) | [Contact](#)

## References and Links

### References and Links

- **GitHub Repository:** <https://github.com/YashrajKupekar17/devops-assignment>
- **Tools Used:** Git, GitHub, Docker, Visual Studio Code / Terminal
- **Base Image:** `python:3.11-slim`