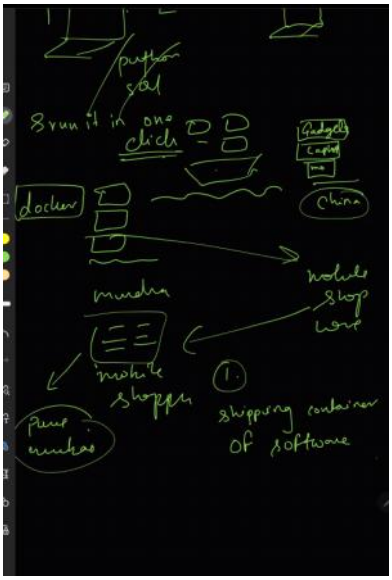
Docker

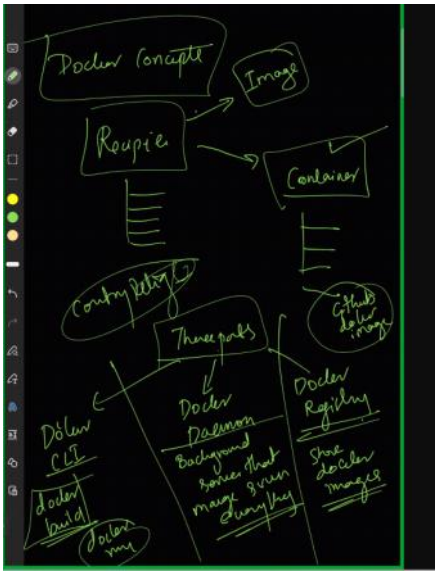
24 February 2025 09:14

Shipping of container of software



Docker creates a package of your application into a container

Package <- container and containers are isolated aka into a virtual kinda environment..(a light weight virtual machine)
So be it a new software or any packages u create a virtual container
U need something that works anywhere, example all machines in a work team doesn't run the same, so docker is useful to run these typa things on any environment. You don't have to reinstall every single thing instead u can use these virtual containers for ur use.



DOCKER CLI
DOCKER DAEMON
DOCCKER REGISTRY

There are some spelling mistakes in your text. Here's what it should be:

- **DOCKER CLI**: The Docker Command Line Interface (CLI) is the tool used to interact with Docker from the terminal. It allows users to manage containers, images, networks, and volumes using commands like docker run, docker build, and docker ps.
- **DOCKER DAEMON**: The Docker Daemon (dockerd) is the background service that manages Docker containers, images, and networks on the host machine. It listens for API requests and handles container lifecycle management.
- **DOCKER REGISTRY**: A Docker Registry is a storage system for Docker images. Public registries like Docker Hub and private registries allow users to push, pull, and manage container images.

Few commands

1. **docker pull <image>**
    - This command downloads a Docker image from a registry (e.g., Docker Hub) to your local

machine.
    - Example: docker pull nginx will pull the official Nginx image.
2. **docker build -t myapp**
    - This command builds a Docker image from a Dockerfile in the current directory (.).
    - The -t myapp flag tags the image with the name myapp.
3. **docker push <image>**
    - This command pushes a locally built image to a remote Docker registry (like Docker Hub).
    - Example: docker push myusername/myapp (requires login with docker login).
4. **docker run myapp** (should be docker run myapp without space in "my app")
    - This command creates and starts a new container from the myapp image.
    - Example: docker run -d -p 8080:80 myapp runs the container in detached mode, mapping port 8080 on the host to port 80 in the container

In docker run -p 8080:80 myapp, the -p flag is used to **map ports** between the host machine and the container. Here's what 8080:80 means:

- 8080 → The **host machine's port** (your local computer).
- 80 → The **container's port** (inside the Docker container).

How it Works:

- When you access http://localhost:8080 on your browser, it forwards traffic to port 80 inside the container.
- This is useful when running web applications because most web servers inside containers (like Nginx, Apache) listen on port 80.

```
# Update package list
sudo apt update

# Install prerequisites
sudo apt install -y apt-transport-https ca-certificates curl software-properties-common

# Add Docker's official GPG key
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o
/usr/share/keyrings/docker-archive-keyring.gpg

# Add Docker repository
echo "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg]
https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | sudo tee
/etc/apt/sources.list.d/docker.list

# Update package list again
sudo apt update

# Install Docker
sudo apt install -y docker-ce docker-ce-cli containerd.io

# Add your user to docker group
sudo usermod -aG docker $USER

# Verify installation
docker --version
```

Now to create dockerfiles

1. mkdir python-docker-project
2. cd python-docker-project
3. mkdir src tests
4. touch src/__init__.py
5. touch src/main.py
6. touch requirements.txt
7. touch Dockerfile
8. touch .dockerignore
9. touch docker-compose.yml

In requirements.txt

```
flask==2.0.1
gunicorn==20.1.0
pytest==7.0.1
```

.dockerignore file

```
__pycache__
*.pyc
*.pyo
*.pyd
.Python
env/
venv/
```

.env
*.log
.git
.gitignore
Dockerfile
.dockerignore
tests/
README.md

To create python environment

1. Python3 -m venv enc
2. To activate
3. source env/bin/activate

And to install everything in requirements file
pip install -r requirements.txt

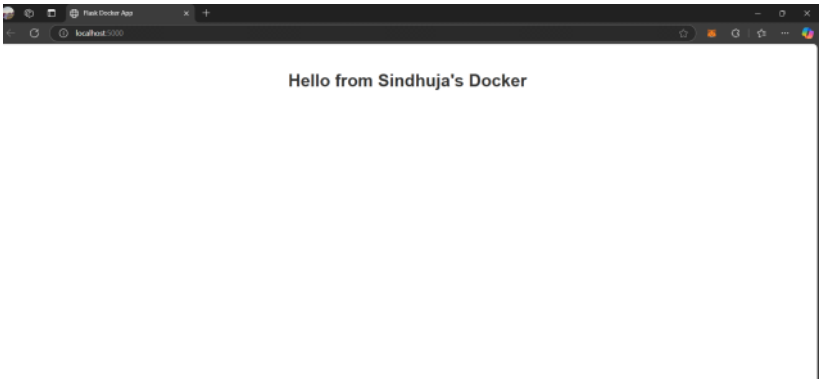
Now in main.py file

```python
from flask import Flask, jsonify

app = Flask(__name__)

@app.route("/health")
def health_check():
    return jsonify({"status": "healthy"})

@app.route("/")
def hello_world():
    return jsonify({"message": "Hello from Sindhuja's docker"})

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000)
```

Now to run this
- Python3 src/main.py

```python
from flask import Flask, jsonify, render_template_string

app = Flask(__name__)

@app.route("/health")
def health_check():
    return jsonify({"status": "healthy"})

@app.route("/")
def hello_world():
    html_content = """
    <!DOCTYPE html>
    <html lang="en">
    <head>
        <meta charset="UTF-8">
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <title>Flask Docker App</title>
        <style>
            body {
                font-family: Arial, sans-serif;
                text-align: center;
                margin-top: 50px;
            }
            h1 {
                color: #333;
            }
        </style>
    </head>
    <body>
        <h1>Hello from Sindhuja's Docker</h1>
    </body>
    </html>
    """
    return render_template_string(html_content)

if __name__ == '__main__':
    app.run(host='0.0.0.0', port=5000)
```

Output do be like



Now u have to build docker image and run it

This code in dockerfile

```
FROM python:3.9-slim
WORKDIR /app
COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt
COPY . .
ENV FLASK_APP=src/main.py
ENV FLASK_ENV=development
ENV PYTHONPATH=/app
EXPOSE 5000
CMD ["gunicorn","--bind","0.0.0.0:5000","src.main:app"]
```

And now this code in .yml

```
version: '3.8'
services:
  web:
    build: .
    ports:
      - "5000:5000"
    volumes:
      - .:/app
    environment:
      - FLASK_APP=src/main.py
      - FLASK_ENV=development
    command: flask run --host=0.0.0.0
```

And in order to run these, stay in env and then execute below two commands

1) # to create images
   docker build -t python-docker-app .
2) # To run it
   docker run -p 5000:5000 python-docker-app

   Change port number if already in use or kill the process
   docker run -p 5001:5000 python-docker-app

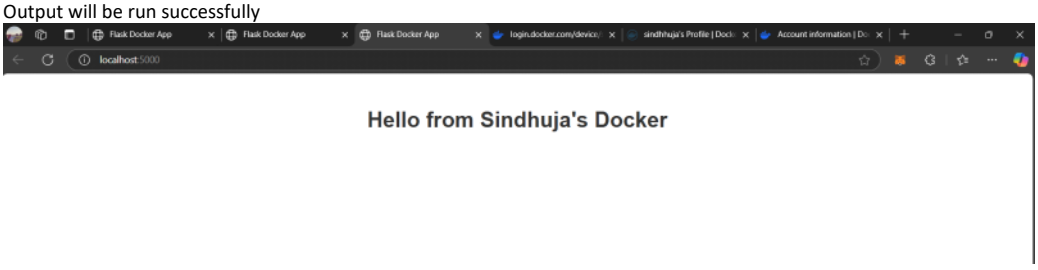
```
(env) sandy@sandy:~/python-docker-project$ docker build -t python-docker-app .
2025/02/24 06:48:32 in: []string{}
2025/02/24 06:48:32 Parsed entitlements: []
[+] Building 0.9s (10/10) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 296B
 => [internal] load metadata for docker.io/library/python:3.9-slim
 => [internal] load .dockerignore
 => => transferring context: 157B
 => [1/5] FROM docker.io/library/python:3.9-slim@sha256:f9364cd6e0c146966f8f23fc4fd85d53f2e604bdde74e3c06565194dc4a02f85
 => [internal] load build context
 => => transferring context: 182B
 => CACHED [2/5] WORKDIR /app
 => CACHED [3/5] COPY requirements.txt .
 => CACHED [4/5] RUN pip install --no-cache-dir -r requirements.txt
 => CACHED [5/5] COPY . .
 => exporting to image
 => => exporting layers
 => => writing image sha256:4cbe9ffb899c8a9393990e4e384588663fd3cac71141a82292a9350f0c56a0d9
 => => naming to docker.io/library/python-docker-app
(env) sandy@sandy:~/python-docker-project$ docker run -p 5001:5000 python-docker-app
[2025-02-24 06:48:46 +0000] [1] [INFO] Starting gunicorn 23.0.0
[2025-02-24 06:48:46 +0000] [1] [INFO] Listening at: http://0.0.0.0:5000 (1)
[2025-02-24 06:48:46 +0000] [1] [INFO] Using worker: sync
```

Now using docker compose :

1) sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
2) sudo chmod +x /usr/local/bin/docker-compose
3) docker-compose --version
4) docker-compose up --build

```
(env) sandy@sandy:~/python-docker-project$ sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)" -o /u
sr/local/bin/docker-compose
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
  0     0    0     0    0     0      0      0 --:--:-- --:--:-- --:--:--     0
100 70.2M  100 70.2M    0     0  2799k      0  0:00:25  0:00:25 --:--:-- 2570k
(env) sandy@sandy:~/python-docker-project$ sudo chmod +x /usr/local/bin/docker-compose
(env) sandy@sandy:~/python-docker-project$ docker-compose --version
Docker Compose version v2.33.1
(env) sandy@sandy:~/python-docker-project$ docker-compose up --build
WARN[0000] /home/sandy/python-docker-project/docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid potential con
fusion
Compose now can delegate build to bake for better performances
Just set COMPOSE_BAKE=true
[+] Building 1.9s (11/11) FINISHED                                                                                                   docker:default
 => [web internal] load build definition from Dockerfile                                                                                        0.0s
 => => transferring dockerfile: 296B                                                                                                            0.0s
 => [web internal] load metadata for docker.io/library/python:3.9-slim                                                                          1.8s
 => [web internal] load .dockerignore                                                                                                           0.0s
 => => transferring context: 157B                                                                                                               0.0s
 => [web 1/5] FROM docker.io/library/python:3.9-slim@sha256:f9364cd6e0c146966f8f23fc4fd85d53f2e604bdde74e3c06565194dc4a02f85                     0.0s
 => [web internal] load build context                                                                                                           0.0s
 => => transferring context: 411B                                                                                                               0.0s
 => CACHED [web 2/5] WORKDIR /app                                                                                                               0.0s
 => CACHED [web 3/5] COPY requirements.txt .                                                                                                    0.0s
 => CACHED [web 4/5] RUN pip install --no-cache-dir -r requirements.txt                                                                         0.0s
 => CACHED [web 5/5] COPY . .                                                                                                                   0.0s
 => [web] exporting to image                                                                                                                    0.0s
 => => exporting layers                                                                                                                         0.0s
 => => writing image sha256:f4009d669b27cdbccb22a09067ec5816f5e202806067078 50fdec6667487d6eb                                                   0.0s
 => => naming to docker.io/library/python-docker-project-web                                                                                    0.0s
 => [web] resolving provenance for metadata file                                                                                                0.0s
[+] Running 2/2
 ✓ web                                   Built                                                                                                  0.0s
 ✓ Container python-docker-project-web-1 Recreated                                                                                              0.1s
Attaching to web-1
web-1  |  * Serving Flask app 'src/main.py'
web-1  |  * Debug mode: off
web-1  | WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
web-1  |  * Running on all addresses (0.0.0.0)
web-1  |  * Running on http://127.0.0.1:5000
web-1  |  * Running on http://172.18.0.2:5000
                                                                                                                                   Ln 7, Col 13   Spaces: 2   UTF-8   LF
```

Output will be run successfully



Now docker login:

1. Log in to Docker Hub

docker login

- This prompts you to enter your Docker Hub username and password.
- If successful, you'll see **"Login Succeeded"**.

2. Tagging the Web Project Image

docker tag python-docker-project-app:latest sindhhuja/python-docker-project-app:v1

- This renames (tags) your **local** Docker image python-docker-project-app:latest to sindhhuja/python-docker-project-web:v1.
- The new tag allows you to push it to Docker Hub.

3. Tagging the App Image

docker tag python-docker-app:latest sindhhuja/python-docker-app:v1

- This renames (tags) your **local** Docker image python-docker-app:latest to sindhhuja/python-docker-app:v1.

4. Push the Web Project Image to Docker Hub

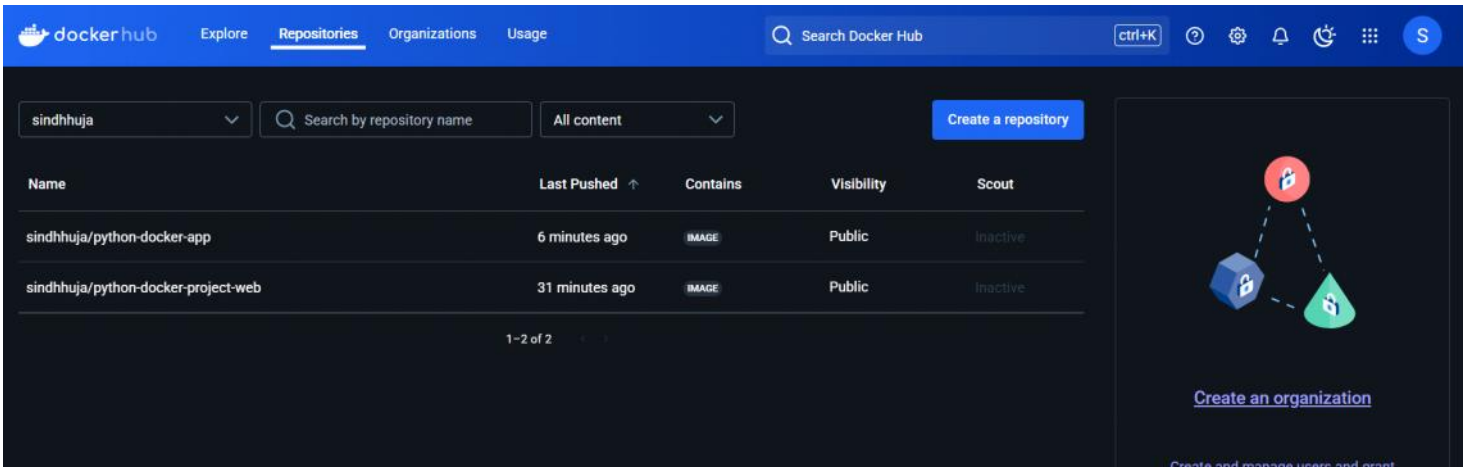
docker push sindhhuja/python-docker-project-app:v1

- Uploads python-docker-project-web:v1 to Docker Hub under your username.

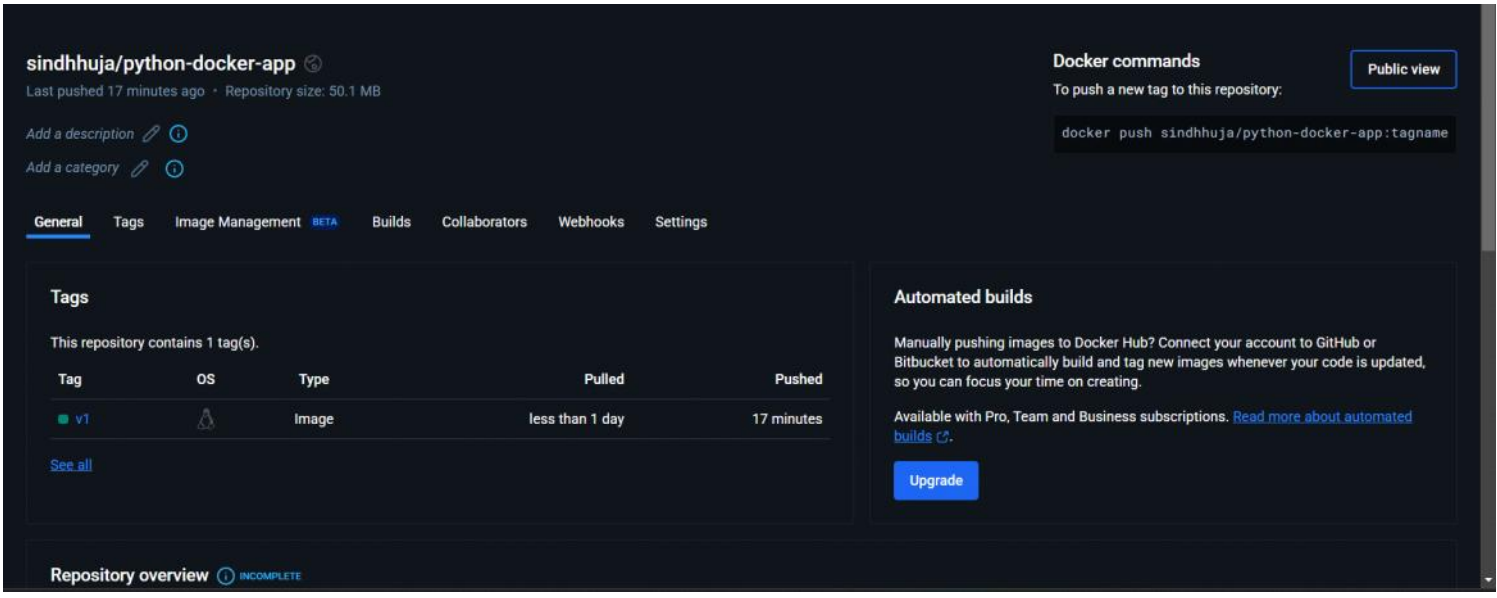
5. Push the App Image to Docker Hub

docker push sindhhuja/python-docker-app:v1

- Uploads python-docker-app:v1 to Docker Hub under your username.

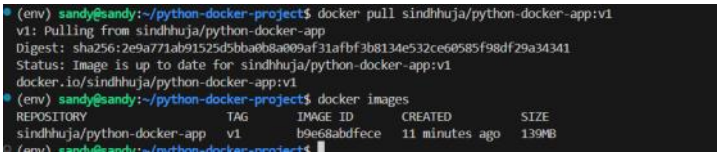


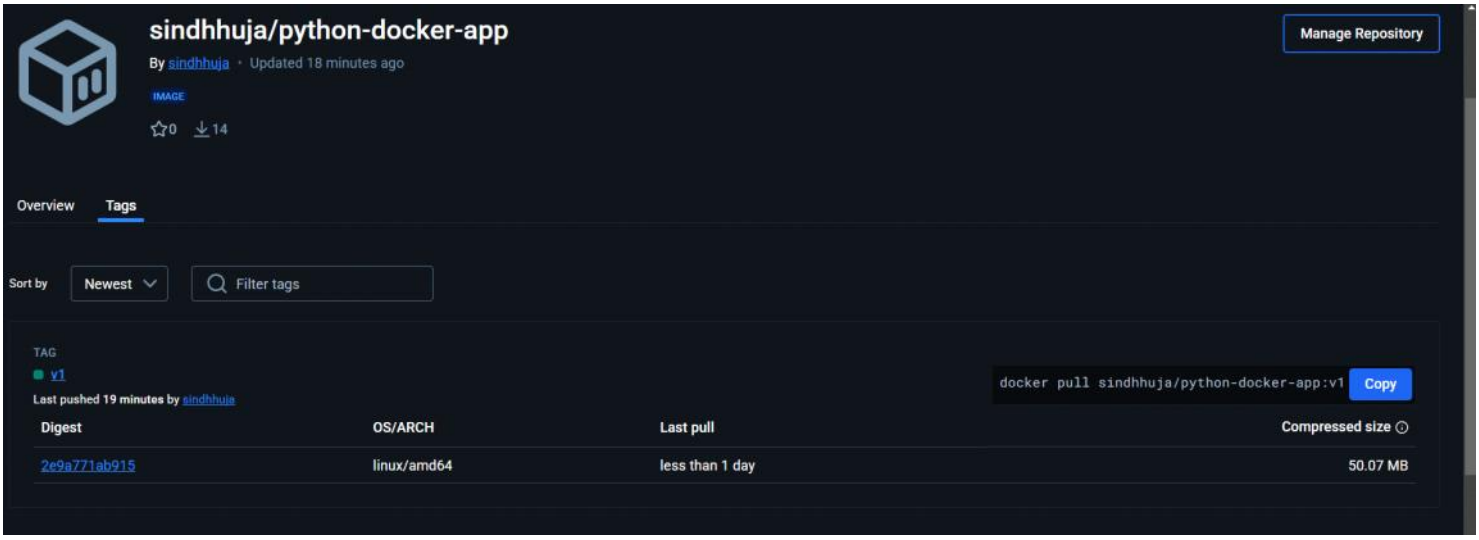
This is how it looks in docker and click on the name of ur docker file to see more details



You can use pull command to check if u r up to date or need to add more :

--> docker pull sindhhuja/python-docker-app:v1




sindhhuja/python-docker-app
By sindhhuja · Updated 18 minutes ago
IMAGE
0 14
Manage Repository
Overview
Tags
Sort by
Newest
Filter tags
TAG
v1
Last pushed 19 minutes by sindhhuja
docker pull sindhhuja/python-docker-app:v1
Copy
Digest
OS/ARCH
Last pull
Compressed size
2e9a771ab915
linux/amd64
less than 1 day
50.07 MB