DEVOPS

STEP1 :

The command sudo apt update, which updates the package lists for available updates and repositories on an Ubuntu system. The output includes information about the sources being updated, including the Jenkins repository and various components from the Ubuntu archives.

```
poojz@zZz:~/devops$ sudo apt update
[sudo] password for poojz:
Ign:1 https://pkg.jenkins.io/debian-stable binary/ InRelease
Hit:2 https://pkg.jenkins.io/debian-stable binary/ Release
Hit:4 http://archive.ubuntu.com/ubuntu noble InRelease
Get:5 http://archive.ubuntu.com/ubuntu noble-updates InRelease [126 kB]
Get:6 http://security.ubuntu.com/ubuntu noble-security InRelease [126 kB]
Get:7 http://archive.ubuntu.com/ubuntu noble-backports InRelease [126 kB]
Get:8 http://archive.ubuntu.com/ubuntu noble-updates/main amd64 Components [151 kB]
Get:9 http://archive.ubuntu.com/ubuntu noble-updates/universe amd64 Components [364 kB]
Get:10 http://archive.ubuntu.com/ubuntu noble-updates/restricted amd64 Components [212 B]
Get:11 http://archive.ubuntu.com/ubuntu noble-updates/multiverse amd64 Components [940 B]
Get:12 http://archive.ubuntu.com/ubuntu noble-backports/main amd64 Components [208 B]
Get:13 http://archive.ubuntu.com/ubuntu noble-backports/universe amd64 Components [20.0 kB]
Get:14 http://archive.ubuntu.com/ubuntu noble-backports/restricted amd64 Components [216 B]
Get:15 http://archive.ubuntu.com/ubuntu noble-backports/multiverse amd64 Components [212 B]
Get:16 http://security.ubuntu.com/ubuntu noble-security/main amd64 Components [9004 B]
Get:17 http://security.ubuntu.com/ubuntu noble-security/universe amd64 Components [52.0 kB]
```

STEP 2:

The terminal output indicates that the user attempted to install Docker on Ubuntu using `sudo apt install -y docker.io`. It shows that Docker is already at the newest version, with no upgrades available for other packages.

```
poojz@zZz:~$ sudo apt install -y docker.io
[sudo] password for poojz:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
docker.io is already the newest version (26.1.3-0ubuntu1~24.04.1).
0 upgraded, 0 newly installed, 0 to remove and 130 not upgraded.
```

STEP 3 :

The commands enabling and starting the Docker service (systemctl enable/start docker) and verifying the installation using docker --version, confirming Docker 26.1.3 on Ubuntu 24.04.

```
poojz@zZz:~$ sudo systemctl enable docker
poojz@zZz:~$ sudo systemctl start docker
```

```
poojz@zZz:~/devops$ docker --version
Docker version 26.1.3, build 26.1.3-0ubuntu1~24.04.1
```

STEP 4 :

The terminal command using curl to download the latest Docker Compose binary from GitHub and save it to /usr/local/bin/docker-compose. The progress bar indicates a successful download of **71.4MB** at a speed of **756kB/s**.

```
poojz@zZz:~$ sudo curl -L "https://github.com/docker/compose/releases/latest/download/docker-compose-$(uname -s)-$(uname -m)" -o /usr
/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 71.4M  100 71.4M    0     0   526k      0  0:02:18  0:02:18 --:--:--  756k
```

STEP 5 :

The commands making Docker Compose executable (chmod +x), verifying its installation (docker-compose --version), creating a ~/devops directory, and navigating into it.

```
poojz@zZz:~$ sudo chmod +x /usr/local/bin/docker-compose
poojz@zZz:~$ docker-compose --version
Docker Compose version v2.34.0
```

```
poojz@zZz:~$ mkdir ~/devops
poojz@zZz:~$ cd ~/devops
```

STEP 6 :

The Flask application (app.py) being created using the nano editor, defining a simple web server that returns "Hello, World! Running inside Docker!" on port 90.

```
poojz@zZz:~/devops$ nano app.py
```

```
poojz@zZz: ~/devops
  GNU nano 7.2                                                    app.py *
from flask import Flask
app = Flask(__name__)
@app.route("/")
def hello():
        return "Hello, World! Running inside Docker!"
if __name__ == "__main__":
        app.run(host="0.0.0.0",port=90)
```

STEP 7:

The creation of a requirements.txt file using the nano editor, listing flask as a dependency for the Python project. Note: There's a typo in the filename (requiremnts.txt).

```
poojz@zZz:~/devops$ nano requiremnts.txt
```

```
poojz@zZz: ~/devops
  GNU nano 7.2                                          requirements.txt *
flask
```

STEP 8:

The `Dockerfile` being created using `nano`, defining a containerized environment for a Python 3.11 Flask app by copying dependencies, installing them, exposing port 90, and running `app.py`.

```
poojz@zZz:~/devops$ nano dockerfile
```

```
poojz@zZz: ~/devops
  GNU nano 7.2                                          dockerfile *
FROM python:3.11
WORKDIR /app
COPY requirements.txt .
RUN pip install --no-cache-dir -r requirements.txt
COPY . .
EXPOSE 90
CMD ["python", "app.py"]
```

STEP 9 :

The docker-compose.yml file being created using nano, defining a service named web that builds from the current directory, maps port 5000:5000, mounts a volume, and restarts automatically.

```
poojz@zZz:~/devops$ nano docker-compose.yml
```

```
poojz@zZz: ~/devops
  GNU nano 7.2                                     docker-compose.yml *
version: '3.8'

services:
  web:
    build: .
    ports:
      - "5000:5000"
    volumes:
      - .:/app
    restart: always
```

STEP 10 :

The image shows the output of sudo docker images, listing three Docker images: test (1.03GB, created 20 hours ago), nginx (192MB, 5 weeks old), and python:3.11 (1.01GB, 3 months old)

```
poojz@zZz:~/devops$ sudo docker images
REPOSITORY   TAG       IMAGE ID       CREATED        SIZE
test         latest    b597cf24bcdc   20 hours ago   1.03GB
nginx        latest    53a18edff809   5 weeks ago    192MB
python       3.11      18c0f2265fd9   3 months ago   1.01GB
```

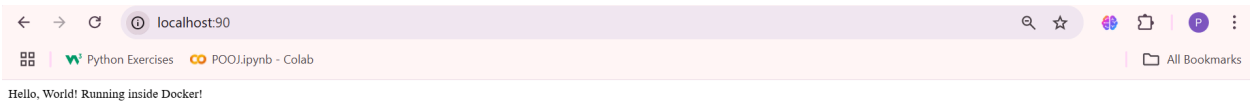
STEP 11 :

The image shows the execution of sudo docker-compose up --build, where Docker Compose is building an image from a docker-compose.yml file. A warning indicates that the version attribute is obsolete, and the build steps confirm that the required files and dependencies are successfully cached.

```
poojz@zZz:~/devops$ sudo docker-compose up --build
WARN[0000] /home/poojz/devops/docker-compose.yml: the attribute `version` is obsolete, it will be ignored, please remove it to avoid
potential confusion
Compose can now delegate builds to bake for better performance.
 To do so, set COMPOSE_BAKE=true.
[+] Building 0.2s (11/11) FINISHED                                                                          docker:default
 => [web internal] load build definition from dockerfile                                                              0.0s
 => => transferring dockerfile: 188B                                                                                  0.0s
 => [web internal] load metadata for docker.io/library/python:3.11                                                    0.0s
 => [web internal] load .dockerignore                                                                                 0.0s
 => => transferring context: 2B                                                                                       0.0s
 => [web 1/5] FROM docker.io/library/python:3.11                                                                      0.0s
 => [web internal] load build context                                                                                 0.0s
 => => transferring context: 164B                                                                                     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
```

OUTPUT:

The final output shows a web browser displaying the message "Hello, World! Running inside Docker!" at localhost:90, confirming that the Flask application successfully runs inside a Docker container. This validates the containerization process, including building the Docker image, running the container, and exposing the correct port.



Hello, World! Running inside Docker!