Size matters: Inspecting Docker Images for Efficiency and Security

Introduction

Irena Grgic, Lead DevOps Engineer at Carl Zeiss AG

Follow me on:

irac.grgic @Medium

LinkedIn

pythonmonty @GitHub

The repository

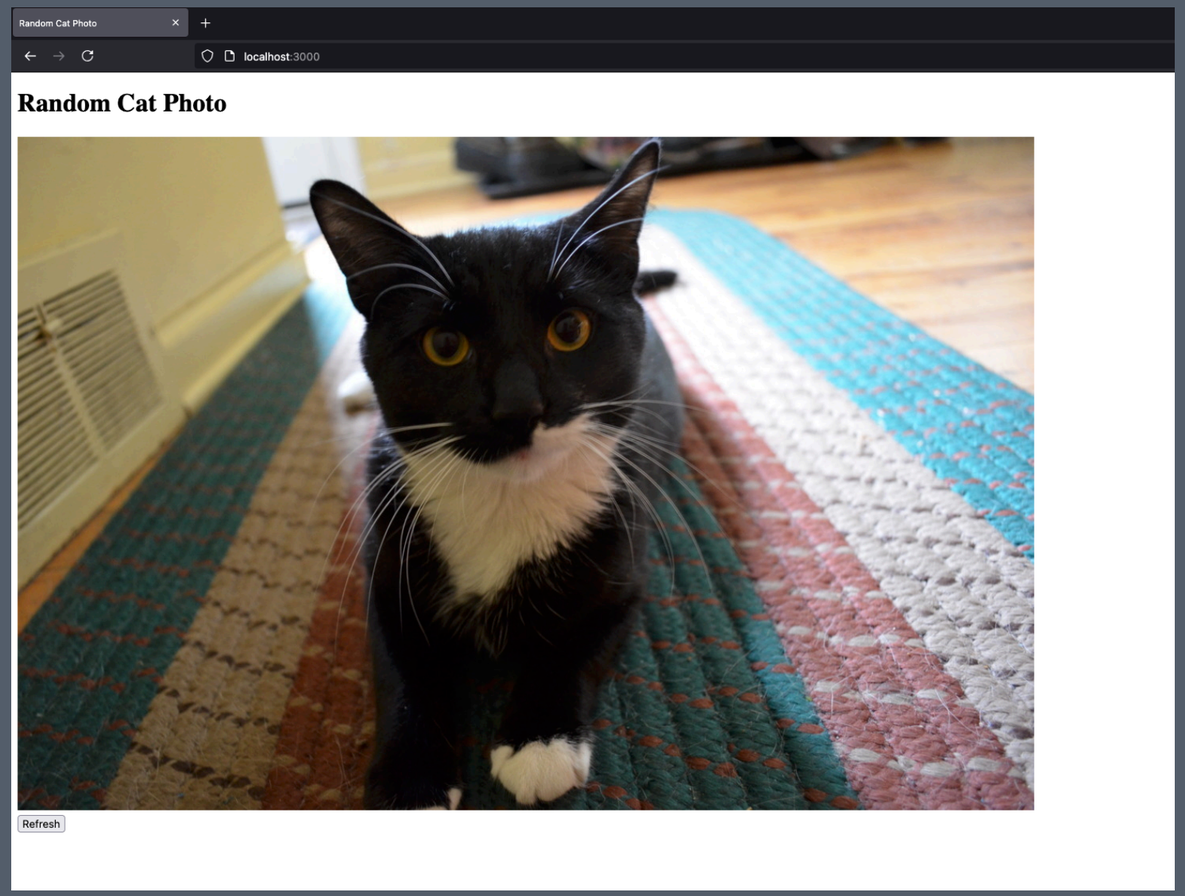
Clone the repository

https://github.com/pythonmonty/inspect-docker-images

The repository

- → A simple Flask app.
- → Sends requests to The Cat API and obtains a cat image.
- → Displays the cat image on a simple HTML page.

The application



Repository structure

```
.
├── README.md
├── cat_app
│   ├── __init__.py
│   └── app.py
├── docker
│   ├── 0_original.Dockerfile
│   ├── 1_base_image.Dockerfile
│   ├── 2_remove_packages.Dockerfile
│   ├── 3_layering.Dockerfile
│   ├── 4_poetry.Dockerfile
│   ├── 5_secrets.Dockerfile
│   ├── 6_non_root_user.Dockerfile
│   ├── 7_bind_mount.Dockerfile
│   ├── 8_multi_stage.Dockerfile
│   └── 9_nice_to_haves.Dockerfile
├── poetry.lock
├── pyproject.toml
└── tests
    └── test_app.py
```

Inspecting the Docker image

Inspecting the Docker image

1. The Dockerfile

```dockerfile
# 0_original.Dockerfile
FROM python:3.12
ARG TOKEN

ENV DEBIAN_FRONTEND=noninteractive
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"

RUN apt-get update
RUN apt-get install -y --no-install-recommends software-properties-common \
    imagemagick \
    build-essential \
    python3-distutils \
    python3-venv \
    python3-pip \
    python3-wheel \
    python3-dev \
    && curl -sS https://bootstrap.pypa.io/get-pip.py | python3

RUN apt-get update
RUN apt-get install --no-install-recommends -y libsm6 \
    libxext6 \
    libxrender-dev \
    libglib2.0-0 \
    libgl1

WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"
RUN poetry install

CMD ["python", "cat_app/app.py"]
```

```dockerfile
# 0_original.Dockerfile
FROM python:3.12
ARG TOKEN

ENV DEBIAN_FRONTEND=noninteractive
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"

RUN apt-get update
RUN apt-get install -y --no-install-recommends software-properties-common \
    imagemagick \
    build-essential \
    python3-distutils \
    python3-venv \
    python3-pip \
    python3-wheel \
    python3-dev \
    && curl -sS https://bootstrap.pypa.io/get-pip.py | python3

RUN apt-get update
RUN apt-get install --no-install-recommends -y libsm6 \
    libxext6 \
    libxrender-dev \
    libglib2.0-0 \
    libgl1

WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"
RUN poetry install

CMD ["python", "cat_app/app.py"]
```

```dockerfile
# 0_original.Dockerfile
FROM python:3.12
ARG TOKEN

ENV DEBIAN_FRONTEND=noninteractive
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"

RUN apt-get update
RUN apt-get install -y --no-install-recommends software-properties-common \
    imagemagick \
    build-essential \
    python3-distutils \
    python3-venv \
    python3-pip \
    python3-wheel \
    python3-dev \
    && curl -sS https://bootstrap.pypa.io/get-pip.py | python3

RUN apt-get update
RUN apt-get install --no-install-recommends -y libsm6 \
    libxext6 \
    libxrender-dev \
    libglib2.0-0 \
    libgl1

WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"
RUN poetry install

CMD ["python", "cat_app/app.py"]
```

```dockerfile
# 0_original.Dockerfile
FROM python:3.12
ARG TOKEN

ENV DEBIAN_FRONTEND=noninteractive
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"

RUN apt-get update
RUN apt-get install -y --no-install-recommends software-properties-common \
    imagemagick \
    build-essential \
    python3-distutils \
    python3-venv \
    python3-pip \
    python3-wheel \
    python3-dev \
    && curl -sS https://bootstrap.pypa.io/get-pip.py | python3

RUN apt-get update
RUN apt-get install --no-install-recommends -y libsm6 \
    libxext6 \
    libxrender-dev \
    libglib2.0-0 \
    libgl1

WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"
RUN poetry install

CMD ["python", "cat_app/app.py"]
```

```dockerfile
# 0_original.Dockerfile
FROM python:3.12
ARG TOKEN

ENV DEBIAN_FRONTEND=noninteractive
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"

RUN apt-get update
RUN apt-get install -y --no-install-recommends software-properties-common \
    imagemagick \
    build-essential \
    python3-distutils \
    python3-venv \
    python3-pip \
    python3-wheel \
    python3-dev \
    && curl -sS https://bootstrap.pypa.io/get-pip.py | python3

RUN apt-get update
RUN apt-get install --no-install-recommends -y libsm6 \
    libxext6 \
    libxrender-dev \
    libglib2.0-0 \
    libgl1

WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"
RUN poetry install

CMD ["python", "cat_app/app.py"]
```

```dockerfile
# 0_original.Dockerfile
FROM python:3.12
ARG TOKEN

ENV DEBIAN_FRONTEND=noninteractive
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"

RUN apt-get update
RUN apt-get install -y --no-install-recommends software-properties-common \
    imagemagick \
    build-essential \
    python3-distutils \
    python3-venv \
    python3-pip \
    python3-wheel \
    python3-dev \
    && curl -sS https://bootstrap.pypa.io/get-pip.py | python3

RUN apt-get update
RUN apt-get install --no-install-recommends -y libsm6 \
    libxext6 \
    libxrender-dev \
    libglib2.0-0 \
    libgl1

WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"
RUN poetry install

CMD ["python", "cat_app/app.py"]
```

```dockerfile
# 0_original.Dockerfile
FROM python:3.12
ARG TOKEN

ENV DEBIAN_FRONTEND=noninteractive
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"

RUN apt-get update
RUN apt-get install -y --no-install-recommends software-properties-common \
    imagemagick \
    build-essential \
    python3-distutils \
    python3-venv \
    python3-pip \
    python3-wheel \
    python3-dev \
    && curl -sS https://bootstrap.pypa.io/get-pip.py | python3

RUN apt-get update
RUN apt-get install --no-install-recommends -y libsm6 \
    libxext6 \
    libxrender-dev \
    libglib2.0-0 \
    libgl1

WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"
RUN poetry install

CMD ["python", "cat_app/app.py"]
```

```dockerfile
# 0_original.Dockerfile
FROM python:3.12
ARG TOKEN

ENV DEBIAN_FRONTEND=noninteractive
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"

RUN apt-get update
RUN apt-get install -y --no-install-recommends software-properties-common \
    imagemagick \
    build-essential \
    python3-distutils \
    python3-venv \
    python3-pip \
    python3-wheel \
    python3-dev \
    && curl -sS https://bootstrap.pypa.io/get-pip.py | python3

RUN apt-get update
RUN apt-get install --no-install-recommends -y libsm6 \
    libxext6 \
    libxrender-dev \
    libglib2.0-0 \
    libgl1

WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"
RUN poetry install

CMD ["python", "cat_app/app.py"]
```

```dockerfile
# 0_original.Dockerfile
FROM python:3.12
ARG TOKEN

ENV DEBIAN_FRONTEND=noninteractive
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"

RUN apt-get update
RUN apt-get install -y --no-install-recommends software-properties-common \
    imagemagick \
    build-essential \
    python3-distutils \
    python3-venv \
    python3-pip \
    python3-wheel \
    python3-dev \
    && curl -sS https://bootstrap.pypa.io/get-pip.py | python3

RUN apt-get update
RUN apt-get install --no-install-recommends -y libsm6 \
    libxext6 \
    libxrender-dev \
    libglib2.0-0 \
    libgl1

WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"
RUN poetry install

CMD ["python", "cat_app/app.py"]
```

```
# 0_original.Dockerfile
FROM python:3.12
ARG TOKEN

ENV DEBIAN_FRONTEND=noninteractive
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"

RUN apt-get update
RUN apt-get install -y --no-install-recommends software-properties-common \
    imagemagick \
    build-essential \
    python3-distutils \
    python3-venv \
    python3-pip \
    python3-wheel \
    python3-dev \
    && curl -sS https://bootstrap.pypa.io/get-pip.py | python3

RUN apt-get update
RUN apt-get install --no-install-recommends -y libsm6 \
    libxext6 \
    libxrender-dev \
    libglib2.0-0 \
    libgl1

WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"
RUN poetry install

CMD ["python", "cat_app/app.py"]
```

Build the image

```
docker build --platform linux/amd64 \
             --build-arg TOKEN="$PAT" \
             -f docker/0_original.Dockerfile \
             -t 0-original .
```

Inspecting the Docker image

2. Docker history

Inspect the image with `docker history`

```
docker history 0-original
```

```
IMAGE          CREATED         CREATED BY                                      SIZE      COMMENT
07fc1ed22e72   25 hours ago    CMD ["python" "cat_app/app.py"]                 0B        buildkit.dockerfile.v0
<missing>      25 hours ago    RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDA…   5.02MB    buildkit.dockerfile.v0
<missing>      25 hours ago    RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDA…   113MB     buildkit.dockerfile.v0
<missing>      25 hours ago    COPY . . # buildkit                             10.1MB    buildkit.dockerfile.v0
<missing>      25 hours ago    WORKDIR /app                                    0B        buildkit.dockerfile.v0
<missing>      25 hours ago    RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDA…   173MB     buildkit.dockerfile.v0
<missing>      25 hours ago    RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDA…   254kB     buildkit.dockerfile.v0
<missing>      25 hours ago    RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDA…   125MB     buildkit.dockerfile.v0
<missing>      25 hours ago    RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDA…   19.6MB    buildkit.dockerfile.v0
<missing>      25 hours ago    RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDA…   0B        buildkit.dockerfile.v0
<missing>      25 hours ago    ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=D…   0B        buildkit.dockerfile.v0
<missing>      25 hours ago    ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=d…   0B        buildkit.dockerfile.v0
<missing>      25 hours ago    ENV POETRY_VIRTUALENVS_IN_PROJECT=0             0B        buildkit.dockerfile.v0
<missing>      25 hours ago    ENV POETRY_VIRTUALENVS_CREATE=0                 0B        buildkit.dockerfile.v0
<missing>      25 hours ago    ENV POETRY_NO_INTERACTION=1                     0B        buildkit.dockerfile.v0
<missing>      25 hours ago    ENV POETRY_VERSION=2.1.0                        0B        buildkit.dockerfile.v0
<missing>      25 hours ago    ENV DEBIAN_FRONTEND=noninteractive              0B        buildkit.dockerfile.v0
<missing>      25 hours ago    ARG TOKEN                                       0B        buildkit.dockerfile.v0
<missing>      10 days ago     CMD ["python3"]                                 0B        buildkit.dockerfile.v0
<missing>      10 days ago     RUN /bin/sh -c set -eux;  for src in idle3 p…   36B       buildkit.dockerfile.v0
<missing>      10 days ago     RUN /bin/sh -c set -eux;   wget -O python.ta…   68.5MB    buildkit.dockerfile.v0
<missing>      10 days ago     ENV PYTHON_SHA256=07ab697474595e06f06647417d…   0B        buildkit.dockerfile.v0
<missing>      10 days ago     ENV PYTHON_VERSION=3.12.10                      0B        buildkit.dockerfile.v0
<missing>      10 days ago     ENV GPG_KEY=7169605F62C751356D054A26A821E680…   0B        buildkit.dockerfile.v0
<missing>      10 days ago     RUN /bin/sh -c set -eux;  apt-get update;  a…   17.8MB    buildkit.dockerfile.v0
<missing>      10 days ago     ENV LANG=C.UTF-8                                0B        buildkit.dockerfile.v0
<missing>      10 days ago     ENV PATH=/usr/local/bin:/usr/local/sbin:/usr…   0B        buildkit.dockerfile.v0
<missing>      15 months ago   RUN /bin/sh -c set -ex;  apt-get update;  ap…   588MB     buildkit.dockerfile.v0
<missing>      15 months ago   RUN /bin/sh -c set -eux;  apt-get update;  a…   177MB     buildkit.dockerfile.v0
<missing>      23 months ago   RUN /bin/sh -c set -eux;  apt-get update;  a…   48.4MB    buildkit.dockerfile.v0
<missing>      23 months ago   # debian.sh --arch 'amd64' out/ 'bookworm' '…   117MB     debuerreotype 0.15
```

```
IMAGE          CREATED         CREATED BY                                      SIZE      COMMENT
07fc1ed22e72   25 hours ago    CMD ["python" "cat_app/app.py"]                 0B        buildkit.dockerfile.v0
<missing>      25 hours ago    RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDA…   5.02MB    buildkit.dockerfile.v0
<missing>      25 hours ago    RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDA…   113MB     buildkit.dockerfile.v0
<missing>      25 hours ago    COPY . . # buildkit                             10.1MB    buildkit.dockerfile.v0
<missing>      25 hours ago    WORKDIR /app                                    0B        buildkit.dockerfile.v0
<missing>      25 hours ago    RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDA…   173MB     buildkit.dockerfile.v0
<missing>      25 hours ago    RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDA…   254kB     buildkit.dockerfile.v0
<missing>      25 hours ago    RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDA…   125MB     buildkit.dockerfile.v0
<missing>      25 hours ago    RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDA…   19.6MB    buildkit.dockerfile.v0
<missing>      25 hours ago    RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDA…   0B        buildkit.dockerfile.v0
<missing>      25 hours ago    ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=D…   0B        buildkit.dockerfile.v0
<missing>      25 hours ago    ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=d…   0B        buildkit.dockerfile.v0
<missing>      25 hours ago    ENV POETRY_VIRTUALENVS_IN_PROJECT=0             0B        buildkit.dockerfile.v0
<missing>      25 hours ago    ENV POETRY_VIRTUALENVS_CREATE=0                 0B        buildkit.dockerfile.v0
<missing>      25 hours ago    ENV POETRY_NO_INTERACTION=1                     0B        buildkit.dockerfile.v0
<missing>      25 hours ago    ENV POETRY_VERSION=2.1.0                        0B        buildkit.dockerfile.v0
<missing>      25 hours ago    ENV DEBIAN_FRONTEND=noninteractive              0B        buildkit.dockerfile.v0
<missing>      25 hours ago    ARG TOKEN                                       0B        buildkit.dockerfile.v0
<missing>      10 days ago     CMD ["python3"]                                 0B        buildkit.dockerfile.v0
<missing>      10 days ago     RUN /bin/sh -c set -eux;  for src in idle3 p…   36B       buildkit.dockerfile.v0
<missing>      10 days ago     RUN /bin/sh -c set -eux;   wget -O python.ta…   68.5MB    buildkit.dockerfile.v0
<missing>      10 days ago     ENV PYTHON_SHA256=07ab697474595e06f06647417d…   0B        buildkit.dockerfile.v0
<missing>      10 days ago     ENV PYTHON_VERSION=3.12.10                      0B        buildkit.dockerfile.v0
<missing>      10 days ago     ENV GPG_KEY=7169605F62C751356D054A26A821E680…   0B        buildkit.dockerfile.v0
<missing>      10 days ago     RUN /bin/sh -c set -eux;  apt-get update;  a…   17.8MB    buildkit.dockerfile.v0
<missing>      10 days ago     ENV LANG=C.UTF-8                                0B        buildkit.dockerfile.v0
<missing>      10 days ago     ENV PATH=/usr/local/bin:/usr/local/sbin:/usr…   0B        buildkit.dockerfile.v0
<missing>      15 months ago   RUN /bin/sh -c set -ex;  apt-get update;  ap…   588MB     buildkit.dockerfile.v0
<missing>      15 months ago   RUN /bin/sh -c set -eux;  apt-get update;  a…   177MB     buildkit.dockerfile.v0
<missing>      23 months ago   RUN /bin/sh -c set -eux;  apt-get update;  a…   48.4MB    buildkit.dockerfile.v0
<missing>      23 months ago   # debian.sh --arch 'amd64' out/ 'bookworm' '…   117MB     debuerreotype 0.15
```

Instructions that have an impact on the image size

→ RUN

→ COPY

→ ADD

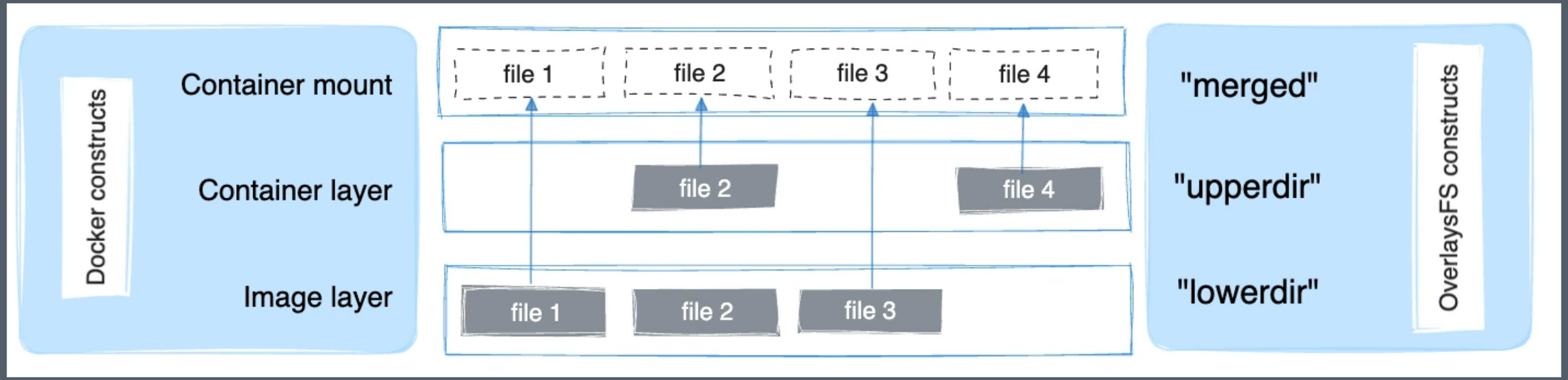
Inspecting the Docker image

3. OverlayFS union filesystem

What happens when you run a container?

- → **Image layers** form the **read-only** base. They are `lowerdir`.
- → A **writeable** container layer is created on top. This is `upperdir`.
- → Docker create a **merged** view of the container unified filesystem. This is `merged` directory.
- → Overlay2 is the **storage driver** that Docker uses to stack the image and container filesystems.

OverlayFS union filesystem[1]



[1] Image source

This means:

→ We can access even the *hidden* image layers by accessing the overlay2 filesystem on the host machine.

Linux: Accessing image and container layers on disk

```
ls -la /var/lib/docker/overlay2
```

MacOS: Accessing image and container layers on disk

Start a Debian privileged container with root access to the host system:

```
docker run -it --privileged --pid=host debian \
      nsenter -t 1 -m -u -n -i \
      sh
```

MacOS: Accessing image and container layers on disk

Run inside the Debian privileged container:

```
ls -la /var/lib/docker/overlay2
```

```
➜  ~ docker run -it --privileged --pid=host debian \
      nsenter -t 1 -m -u -n -i \
      sh
/ # ls -la /var/lib/docker/overlay2
total 3228
drwx--x---  785 root     root        49152 Apr 18 21:12 .
drwx--x---   12 root     root         4096 Apr 15 14:37 ..
drwx--x---    4 root     root         4096 Feb  5 16:04 0325cb9fa0755c5e4ad48ab54b03e188d940312dc35b736d2f1ac6b503da4574
drwx--x---    4 root     root         4096 Feb  5 16:04 0472fc03e8bb52d235e8b5ac6061fbcf5fef22c52375fa5c7bb227b447359c07
drwx--x---    4 root     root         4096 Feb  5 16:04 0511192ccb2703302d4eaf058f0d56fb9bde74474cb60b8bd167467b0efa12f7
drwx--x---    4 root     root         4096 Apr  7 14:30 0652nn03hydgq5ccy4ark770g
```

Inspecting the Docker image

4. Docker inspect

Inspect the image with `docker inspect`

```
docker inspect 0-original
```

```
[
    {
        "Id": "sha256:07fc1ed22e7296bf2bcd150810dc53e269d814233 4cdc6aba1dee42ea486 4ef1",
        "RepoTags": [
            "0-original:latest"
        ],
        "RepoDigests": [],
        "Parent": "",
        "Comment": "buildkit.dockerfile.v0",
        "Created": "2025-04-17T17:36:35.648678215Z",
        "Container": "",
        "ContainerConfig": {
            "Hostname": "",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": null,
            "Cmd": null,
            "Image": "",
            "Volumes": null,
            "WorkingDir": "",
            "Entrypoint": null,
            "OnBuild": null,
            "Labels": null
        },
        "DockerVersion": "",
        "Author": "",
        "Config": {
            "Hostname": "",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": [
                "PATH=/usr/local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin",
                "LANG=C.UTF-8",
                "GPG_KEY=7169605F62C751356D054A26A821E680E5FA6305",
                "PYTHON_VERSION=3.12.10",
                "PYTHON_SHA256=07ab69747595e06f06647417d3c7fa97ded07afc1a7e4454c5639919b46eaea",
                "DEBIAN_FRONTEND=noninteractive",
                "POETRY_VERSION=2.1.0",
                "POETRY_NO_INTERACTION=1",
                "POETRY_VIRTUALENVS_CREATE=0",
                "POETRY_VIRTUALENVS_IN_PROJECT=0",
                "POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=docker-user",
                "POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=DeFxwLAO0aKSnc9bMHHJec8d04MpgDAjqyTwivYQzWBysl kzDUShJQQJ99BCACAAAAAAAAAAAASAZDOgpe4"
            ],
            "Cmd": [
                "python",
                "cat_app/app.py"
            ],
            "ArgsEscaped": true,
            "Image": "",
            "Volumes": null,
            "WorkingDir": "/app",
            "Entrypoint": null,
            "OnBuild": null,
            "Labels": null
        },
        "Architecture": "amd64",
        "Os": "linux",
        "Size": 1462279615,
        "VirtualSize": 1462279615,
        "GraphDriver": {
            "Data": {
                "LowerDir": "/var/lib/docker/overlay2/mse04v1g84340iw5woqjabp2h/diff:/var/lib/docker/overlay2/fjic757fm1bw2nh613e9nyrqo/diff:
                /var/lib/docker/overlay2/iubt0thlvo881q3pifn4tj6pbb/diff:/var/lib/docker/overlay2/a30pkgydx8r8ud3oivkx4ea7j/diff:
                /var/lib/docker/overlay2/07hs502ow828cc68r328rf94p/diff:/var/lib/docker/overlay2/zc7lnlm6kzwxhc2fb1tnlvq3j/diff:
                /var/lib/docker/overlay2/tfjkee7mr3ng5i4x1wjw7mjve/diff:/var/lib/docker/overlay2/sbr9iduct38yzhynoroliojuS/diff:
                /var/lib/docker/overlay2/fd28c533a2d2b73f82ff52c2bfa5652be26f07961022fac547983316c5405068/diff:/var/lib/docker/overlay2/4f2cbdf3aee2772f7caa6470f2b0a413ab972620bb4f007af851fb682c0f0ded/diff:
                /var/lib/docker/overlay2/f5fa4xe8b00216186 9c5e5e4ae9a19fea6378d27f0b567debad68dd72ff127e5/diff:/var/lib/docker/overlay2/4eefa20a1e913502b915b9c65d475c6c3a09008f4f2146496cb18c7c8ff82482/diff:
                /var/lib/docker/overlay2/32fc57dd0917ad5acf6beb61b0ff8 4d4d118d6c38a81a216076e3d61ac8864b05/diff:/var/lib/docker/overlay2/97c8e152943113941023eb985d2be244 8b2895efGc8604 07106a89b33d5ae2bb/diff:
                /var/lib/docker/overlay2/dae590ffc404b98621ad88289038c5c51d2311ade8d1b63bec28522ee4d9b9fc/diff",
                "MergedDir": "/var/lib/docker/overlay2/inx6s5r51fwv90lmy74z6w678/merged",
                "UpperDir": "/var/lib/docker/overlay2/inx6s5r51fwv90lmy74z6w678/diff",
                "WorkDir": "/var/lib/docker/overlay2/inx6s5r51fwv90lmy74z6w678/work"
            },
            "Name": "overlay2"
        },
        "RootFS": {
            "Type": "layers",
            "Layers": [
                "sha256:f7f2b929d8a55112a2db1bc16fb87310 45c9572b84d6dfbbdbd5dc6dd2bd9fe4",
                "sha256:7f0053786e6e2411ce577e795ef6c278601de3faa0817c071801b11a3f83fd0d",
                "sha256:b2bcbd8ebb2be9869ca3198b4c521f18b612b4287ee4b705678d7b85383b71af",
                "sha256:6c7c1b88da61a2ad4efd3cb93b6e31b0937d3f4121050a9d1c4dd0b0485c7c68",
                "sha256:684b5a3b48fffb94f66fb5779e428a3c759 41c43e8f75231a03e37b61d6612d2f",
                "sha256:0941852 0a5f84d72e78e3d83ff80f42b1457285 6dddb806226e949b3f802cf66",
                "sha256:e9b3f3afe636ef33be417644e7696c3b6c8415c76d8932236dd2a35257 8ffec9",
                "sha256:a19a92503c801249c43e13d7f18a49a73a0cd0c7e0b8602fe76c2ab6ab0b6444",
                "sha256:571dd1be3f3b980d71254b8866bf6ee581c414ef6a23cc1ea3078704a56790",
                "sha256:7810c0c51b5ec8a9f98c60889dca0465dde1f3a56a8d6d4c2b6b43c68d1b9b46",
                "sha256:e0ac25e07f3fd116a6294e501c004 0eab3d9793c0f2d2dca9bc7ded098de9f7a",
                "sha256:86e3fb74a6e15dd2c3e22c38e9c95561d806d9d19d2a3311e1ba0d677a9c878a",
                "sha256:9caaf427cc29d2d27a32f6582f1746a2a4bda77b8fc8734 2baa83054133de",
                "sha256:8f53369f36e49fb8e0b1d283efb93c311cbb4d71581433a92436e376661b190",
                "sha256:0d918acd66efa3c4f5c94b7289bf5e83b5b300ea8f38f442cd75bab2000d707e",
                "sha256:7da5d094be9101ea5551 0572f0d118a2eef6cde20a3f17f4377891 0ab7b82cd"
            ]
        },
        "Metadata": {
            "LastTagTime": "2025-04-17T17:36:36.097356882Z"
        }
    }
]
```

`docker inspect`: Environment variables

```
"Env": [
    "PATH=/usr/local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin",
    "LANG=C.UTF-8",
    "GPG_KEY=7169605F62C751356D054A26A821E680E5FA6305",
    "PYTHON_VERSION=3.12.10",
    "PYTHON_SHA256=07ab697474595e06f06647417d3c7fa97ded07afc1a7e4454c5639919b46eaea",
    "DEBIAN_FRONTEND=noninteractive",
    "POETRY_VERSION=2.1.0",
    "POETRY_NO_INTERACTION=1",
    "POETRY_VIRTUALENVS_CREATE=0",
    "POETRY_VIRTUALENVS_IN_PROJECT=0",
    "POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=docker-user",
    "POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=DeFxwLAO0aKSnc9bMHHJec8d04MpgDAjqyTwivYQzW8yslkzDUShJQQJ99BCACAAAAAAAAAAAAASAZDOgpe4"
],
```

`docker inspect`: Layer's digest

```
"RootFS": {
    "Type": "layers",
    "Layers": [
        "sha256:f7f2b929d8a55112a2db1bc16fb8731045c9572b84d6dfbbdbd5dc6dd2bd9fe4",
        "sha256:7f0053786e6e2411ce577e795ef6c278601de3faa0817c071801b11a3f83fd0d",
        "sha256:b2bcbd8ebb2be9869ca3198b4c521f18b612b4287ee4b705678d7b85383b71af",
        "sha256:6c7c1b88da61a2ad4efd3cb93b6e31b0937d3f4121050a9d1c4ddd00485c7c68",
        "sha256:68405a3b48fffb94f66fb5779e420a3c75941c43e8f75231a03e37b61c661d2d",
        "sha256:09418520a5f84d72e78e3d83ff80f42b14572856dddb806226e949b3f802cf66",
        "sha256:e9b3f3afe636ef33be417644e7696c3b6c8415c76d8932236dd2a352578ffec9",
        "sha256:a19a92503c801249c43e13d7f18a49a73a0cd0c7e0b8602fe76c2ab6ab0b6444",
        "sha256:571dd1be3f3b980d712540d866bf6ee581c414efd4a23cc1ea30708704a56790",
        "sha256:7818c0c51b5ec8a9f98c60889dca0465dde1f3a56a8d6d4c2b6b43c68d1b9b46",
        "sha256:e0ac25e07f3fd116a6294e501c0040eab3d9793c0f2d2dca9bc7ded098de9f7a",
        "sha256:86e3fb74a6e15dd2c3e22c38e9c95561d806d9d19d2a3311e1ba0d677a9c878a",
        "sha256:9caaf427cc29d2d27a32f65827f1746a2a44ba477b0fc873442baa83054133de",
        "sha256:8f53369f36e49fb8e0b1d283efb93c311cbb4d171581433a92436e376661b190",
        "sha256:0d918acd66efa3c4f5c94b7289bf5e83b5b300ea8f38f442cd75bab2000d707e",
        "sha256:7da5d094be9101ea535108572f0d118a2eef6cde20a3f17f43778910ab7b82cd"
    ]
},
```

`docker inspect`: OverlayFS storage driver

```
"GraphDriver": {
    "Data": {
        "LowerDir": "/var/lib/docker/overlay2/mse04v1g8434oiw5woqjabp2h/diff:
                    /var/lib/docker/overlay2/fjic757fm1bw2nh613e9nyrqo/diff:
                    /var/lib/docker/overlay2/iubt0thlvo88i3pifn4tj6pbb/diff:
                    /var/lib/docker/overlay2/a30pkgydx8r8ud3oivkx4ea7j/diff:
                    /var/lib/docker/overlay2/07hs5m2ow828cc68r328rf94p/diff:
                    /var/lib/docker/overlay2/zc7ln1m6kzw4hc2fbltn1vq3j/diff:
                    /var/lib/docker/overlay2/tfjkee7mr3ng5i4×1wjw7mjve/diff:
                    /var/lib/docker/overlay2/sbr9iduct38yzhynorolioju5/diff:
                    /var/lib/docker/overlay2/fd28c533a2d2b73f82ff52c2bfa5652be26f07961022fac547983316c5405068/diff:
                    /var/lib/docker/overlay2/4f2cbdf3aee2772f7caa6470f2b0a413ab972620bb4f007af851fb682c0f0ded/diff:
                    /var/lib/docker/overlay2/f5fa44e8b002161869c5e5e4ae9a19fea6378d27f0b567debad68dd72ff127e5/diff:
                    /var/lib/docker/overlay2/4eefa20a1e913502b915b9c65d475c6c3a09008f4f2146496cb18c7c8ff82482/diff:
                    /var/lib/docker/overlay2/32fc57dd0917ad5acf6beb61b0ff046d118d6c38a01a216076e3d61ac08649b5/diff:
                    /var/lib/docker/overlay2/97c0e152943113941023eb985d2be2448b2895ef6c860407106a89b33d5ae2bb/diff:
                    /var/lib/docker/overlay2/dae590ffc404b986214d88289038c5c51d2311ade8d1b63bec28522ee4d9b9fc/diff",
        "MergedDir": "/var/lib/docker/overlay2/inx6s5r51fwv90lmy74z6w678/merged",
        "UpperDir": "/var/lib/docker/overlay2/inx6s5r51fwv90lmy74z6w678/diff",
        "WorkDir": "/var/lib/docker/overlay2/inx6s5r51fwv90lmy74z6w678/work"
    },
    "Name": "overlay2"
},
```

Inspecting the Docker image

5. Dive

Inspect the image with Dive

Dive: a command-line tool for exploring Docker images.

```
dive 0-original
```

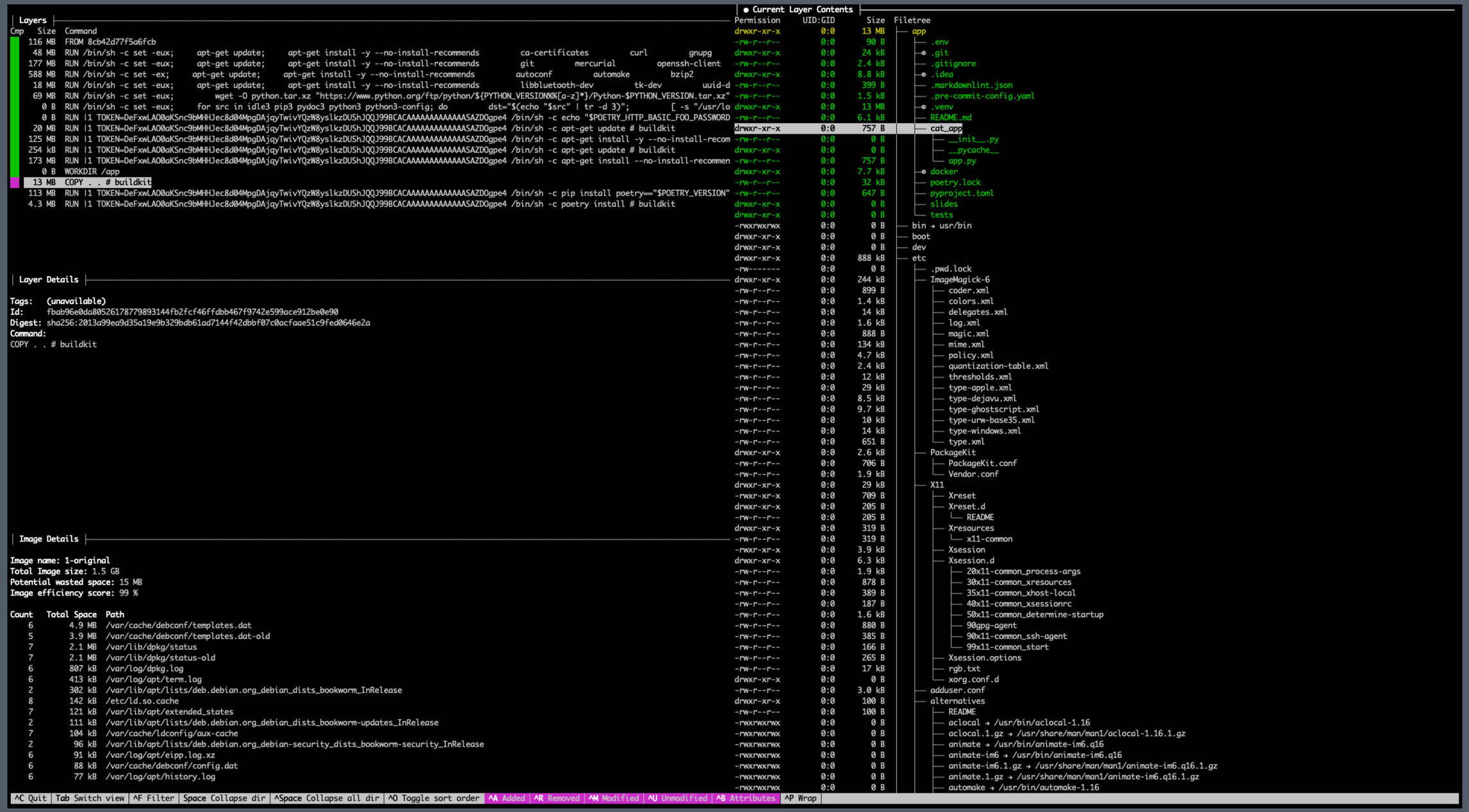

Layers
Cmp Size Command
116 MB FROM 8cb42d77f5a6fcb
48 MB RUN /bin/sh -c set -eux; apt-get update; apt-get install -y --no-install-recommends ca-certificates curl gnupg
177 MB RUN /bin/sh -c set -eux; apt-get update; apt-get install -y --no-install-recommends git mercurial openssh-client
588 MB RUN /bin/sh -c set -ex; apt-get update; apt-get install -y --no-install-recommends autoconf automake bzip2
18 MB RUN /bin/sh -c set -eux; apt-get update; apt-get install -y --no-install-recommends libbluetooth-dev tk-dev uuid-d
69 MB RUN /bin/sh -c set -eux; wget -O python.tar.xz "https://www.python.org/ftp/python/${PYTHON_VERSION%%[a-z]*}/Python-$PYTHON_VERSION.tar.xz"
0 B RUN /bin/sh -c set -eux; for src in idle3 pip3 pydoc3 python3 python3-config; do dst="$(echo "$src" | tr -d 3)"; [-s "/usr/lo
0 B RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDAjqyTwivYQzW8yslkzDUShJQQJ99BCACAAAAAAAAAAAAASAZDOgpe4 /bin/sh -c echo "$POETRY_HTTP_BASIC_FOO_PASSWORD
20 MB RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDAjqyTwivYQzW8yslkzDUShJQQJ99BCACAAAAAAAAAAAAASAZDOgpe4 /bin/sh -c apt-get update # buildkit
125 MB RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDAjqyTwivYQzW8yslkzDUShJQQJ99BCACAAAAAAAAAAAAASAZDOgpe4 /bin/sh -c apt-get install -y --no-install-recom
254 kB RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDAjqyTwivYQzW8yslkzDUShJQQJ99BCACAAAAAAAAAAAAASAZDOgpe4 /bin/sh -c apt-get update # buildkit
173 MB RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDAjqyTwivYQzW8yslkzDUShJQQJ99BCACAAAAAAAAAAAAASAZDOgpe4 /bin/sh -c apt-get install --no-install-recommen
0 B WORKDIR /app
13 MB COPY . . # buildkit
113 MB RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDAjqyTwivYQzW8yslkzDUShJQQJ99BCACAAAAAAAAAAAAASAZDOgpe4 /bin/sh -c pip install poetry=="$POETRY_VERSION"
4.3 MB RUN |1 TOKEN=DeFxwLAO0aKSnc9bMHHJec8d04MpgDAjqyTwivYQzW8yslkzDUShJQQJ99BCACAAAAAAAAAAAAASAZDOgpe4 /bin/sh -c poetry install # buildkit
Layer Details
Tags: (unavailable)
Id: fbab96e0da80526178779893144fb2fcf46ffdbb467f9742e599ace912be0e90
Digest: sha256:2013a99ea9d35a19e9b329bdb61ad7144f42dbbf07c0acfaae51c9fed0646e2a
Command:
COPY . . # buildkit
Image Details
Image name: 1-original
Total Image size: 1.5 GB
Potential wasted space: 15 MB
Image efficiency score: 99 %
Count Total Space Path
6 4.9 MB /var/cache/debconf/templates.dat
5 3.9 MB /var/cache/debconf/templates.dat-old
7 2.1 MB /var/lib/dpkg/status
7 2.1 MB /var/lib/dpkg/status-old
6 807 kB /var/log/dpkg.log
6 413 kB /var/log/apt/term.log
2 302 kB /var/lib/apt/lists/deb.debian.org_debian_dists_bookworm_InRelease
8 142 kB /etc/ld.so.cache
7 121 kB /var/lib/apt/extended_states
2 111 kB /var/lib/apt/lists/deb.debian.org_debian_dists_bookworm-updates_InRelease
7 104 kB /var/cache/ldconfig/aux-cache
2 96 kB /var/lib/apt/lists/deb.debian.org_debian-security_dists_bookworm-security_InRelease
6 91 kB /var/log/apt/eipp.log.xz
6 88 kB /var/cache/debconf/config.dat
6 77 kB /var/log/apt/history.log
Current Layer Contents
Permission UID:GID Size Filetree
drwxr-xr-x 0:0 13 MB app
-rw-r--r-- 0:0 90 B .env
drwxr-xr-x 0:0 24 kB .git
-rw-r--r-- 0:0 2.4 kB .gitignore
drwxr-xr-x 0:0 8.8 kB .idea
-rw-r--r-- 0:0 399 B .markdownlint.json
-rw-r--r-- 0:0 1.5 kB .pre-commit-config.yaml
drwxr-xr-x 0:0 13 MB .venv
-rw-r--r-- 0:0 6.1 kB README.md
drwxr-xr-x 0:0 757 B cat_app
-rw-r--r-- 0:0 0 B __init__.py
drwxr-xr-x 0:0 0 B __pycache__
-rw-r--r-- 0:0 757 B app.py
drwxr-xr-x 0:0 7.7 kB docker
-rw-r--r-- 0:0 32 kB poetry.lock
-rw-r--r-- 0:0 647 B pyproject.toml
drwxr-xr-x 0:0 0 B slides
drwxr-xr-x 0:0 0 B tests
-rwxrwxrwx 0:0 0 B bin → usr/bin
drwxr-xr-x 0:0 0 B boot
drwxr-xr-x 0:0 0 B dev
drwxr-xr-x 0:0 888 kB etc
-rw------- 0:0 0 B .pwd.lock
drwxr-xr-x 0:0 244 kB ImageMagick-6
-rw-r--r-- 0:0 899 B coder.xml
-rw-r--r-- 0:0 1.4 kB colors.xml
-rw-r--r-- 0:0 14 kB delegates.xml
-rw-r--r-- 0:0 1.6 kB log.xml
-rw-r--r-- 0:0 888 B magic.xml
-rw-r--r-- 0:0 134 kB mime.xml
-rw-r--r-- 0:0 4.7 kB policy.xml
-rw-r--r-- 0:0 2.4 kB quantization-table.xml
-rw-r--r-- 0:0 12 kB thresholds.xml
-rw-r--r-- 0:0 29 kB type-apple.xml
-rw-r--r-- 0:0 8.5 kB type-dejavu.xml
-rw-r--r-- 0:0 9.7 kB type-ghostscript.xml
-rw-r--r-- 0:0 10 kB type-urw-base35.xml
-rw-r--r-- 0:0 14 kB type-windows.xml
-rw-r--r-- 0:0 651 B type.xml
drwxr-xr-x 0:0 2.6 kB PackageKit
-rw-r--r-- 0:0 706 B PackageKit.conf
-rw-r--r-- 0:0 1.9 kB Vendor.conf
drwxr-xr-x 0:0 29 kB X11
-rwxr-xr-x 0:0 709 B Xreset
drwxr-xr-x 0:0 205 B Xreset.d
-rw-r--r-- 0:0 205 B README
drwxr-xr-x 0:0 319 B Xresources
-rw-r--r-- 0:0 319 B x11-common
-rwxr-xr-x 0:0 3.9 kB Xsession
drwxr-xr-x 0:0 6.3 kB Xsession.d
-rw-r--r-- 0:0 1.9 kB 20x11-common_process-args
-rw-r--r-- 0:0 878 B 30x11-common_xresources
-rw-r--r-- 0:0 389 B 35x11-common_xhost-local
-rw-r--r-- 0:0 187 B 40x11-common_xsessionrc
-rw-r--r-- 0:0 1.6 kB 50x11-common_determine-startup
-rw-r--r-- 0:0 880 B 90gpg-agent
-rw-r--r-- 0:0 385 B 90x11-common_ssh-agent
-rw-r--r-- 0:0 166 B 99x11-common_start
-rw-r--r-- 0:0 265 B Xsession.options
-rw-r--r-- 0:0 17 kB rgb.txt
drwxr-xr-x 0:0 0 B xorg.conf.d
-rw-r--r-- 0:0 3.0 kB adduser.conf
drwxr-xr-x 0:0 100 B alternatives
-rw-r--r-- 0:0 100 B README
-rwxrwxrwx 0:0 0 B aclocal → /usr/bin/aclocal-1.16
-rwxrwxrwx 0:0 0 B aclocal.1.gz → /usr/share/man/man1/aclocal-1.16.1.gz
-rwxrwxrwx 0:0 0 B animate → /usr/bin/animate-im6.q16
-rwxrwxrwx 0:0 0 B animate-im6 → /usr/bin/animate-im6.q16
-rwxrwxrwx 0:0 0 B animate-im6.1.gz → /usr/share/man/man1/animate-im6.q16.1.gz
-rwxrwxrwx 0:0 0 B animate.1.gz → /usr/share/man/man1/animate-im6.q16.1.gz
-rwxrwxrwx 0:0 0 B automake → /usr/bin/automake-1.16
^C Quit
Tab Switch view
^F Filter
Space Collapse dir
^Space Collapse all dir
^O Toggle sort order
^A Added
^R Removed
^M Modified
^U Unmodified
^B Attributes
^P Wrap

Image details

→ Size: 1.5 GB

→ Layers: 16

→ Build time: ~188 sec

Combining Dive with `docker inspect`

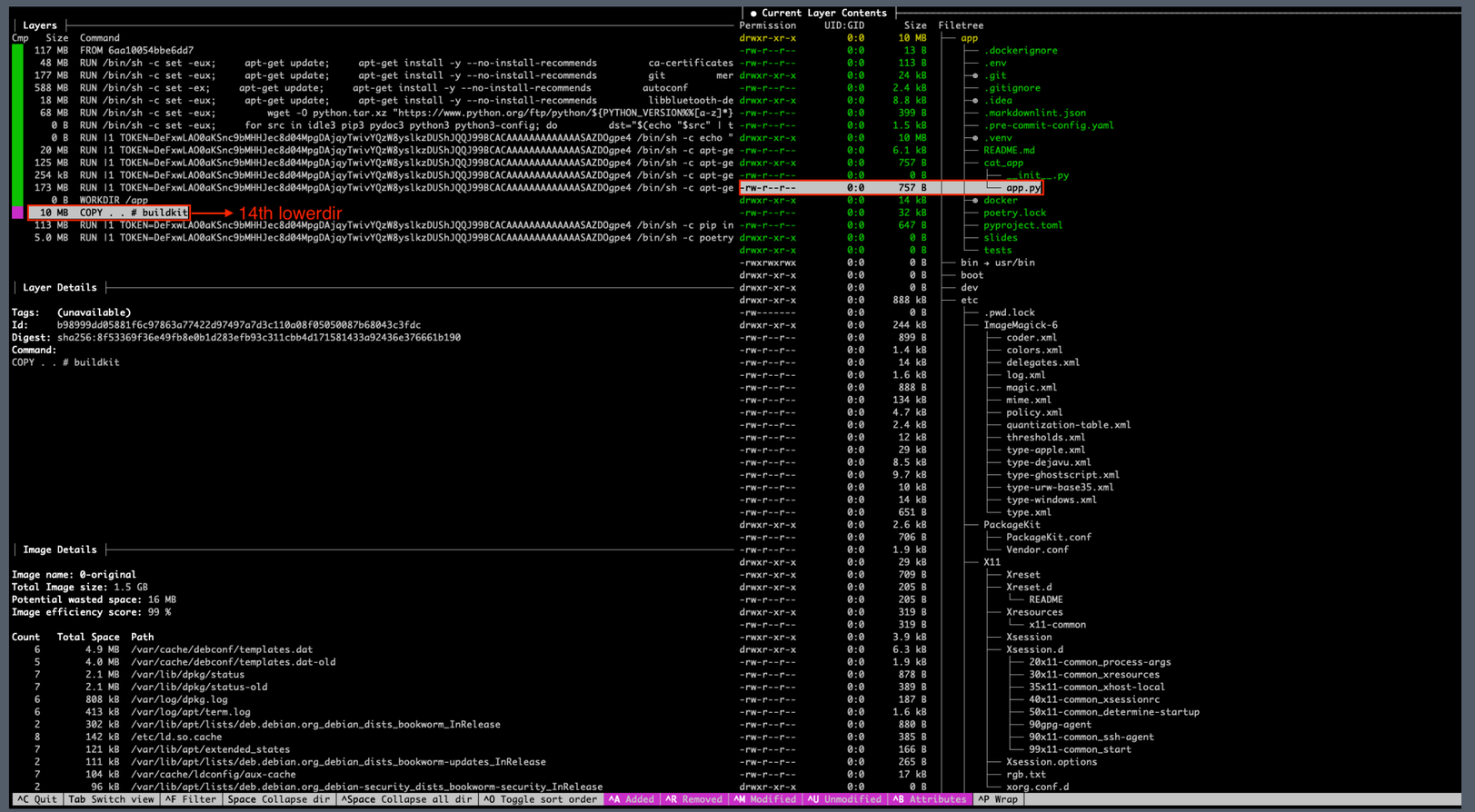
Combining Dive with `docker inspect`

→ Use Dive to **find the layer** whose files you want to inspect.

→ **Order the layers** in `docker inspect` to match Dive.

→ Use `overlay2` to **access the files.**

Use Dive to find the layer you want to inspect



Order the layers in `docker inspect` to match Dive

```
docker inspect 0-original | jq '
  .[]
  | .GraphDriver.Data.UpperDir + ":" + .GraphDriver.Data.LowerDir
  | split(":")
  | reverse
'
```

Order the layers in `docker inspect` to match Dive

```
[
  "/var/lib/docker/overlay2/dae590ffc404b986214d88289038c5c51d2311ade8d1b63bec28522ee4d9b9fc/diff",
  "/var/lib/docker/overlay2/97c0e152943113941023eb985d2be2448b2895ef6c860407106a89b33d5ae2bb/diff",
  "/var/lib/docker/overlay2/32fc57dd0917ad5acf6beb61b0ff046d118d6c38a01a216076e3d61ac08649b5/diff",
  "/var/lib/docker/overlay2/4eefa20a1e913502b915b9c65d475c6c3a09008f4f2146496cb18c7c8ff82482/diff",
  "/var/lib/docker/overlay2/f5fa44e8b002161869c5e5e4ae9a19fea6378d27f0b567debad68dd72ff127e5/diff",
  "/var/lib/docker/overlay2/4f2cbdf3aee2772f7caa6470f2b0a413ab972620bb4f007af851fb682c0f0ded/diff",
  "/var/lib/docker/overlay2/fd28c533a2d2b73f82ff52c2bfa5652be26f07961022fac547983316c5405068/diff",
  "/var/lib/docker/overlay2/sbr9iduct38yzhynorolioju5/diff",
  "/var/lib/docker/overlay2/tfjkee7mr3ng5i4×1wjw7mjve/diff",
  "/var/lib/docker/overlay2/zc7ln1m6kzw4hc2fbltn1vq3j/diff",
  "/var/lib/docker/overlay2/07hs5m2ow828cc68r328rf94p/diff",
  "/var/lib/docker/overlay2/a30pkgydx8r8ud3oivkx4ea7j/diff",
  "/var/lib/docker/overlay2/iubt0thlvo88i3pifn4tj6pbb/diff",
  "/var/lib/docker/overlay2/fjic757fm1bw2nh613e9nyrqo/diff",
  "/var/lib/docker/overlay2/mse04v1g8434oiw5woqjabp2h/diff",
  "/var/lib/docker/overlay2/inx6s5r51fwv90lmy74z6w678/diff"
]
```

Order the layers in `docker inspect` to match Dive

```
[
  "/var/lib/docker/overlay2/dae590ffc404b986214d88289038c5c51d2311ade8d1b63bec28522ee4d9b9fc/diff",
  "/var/lib/docker/overlay2/97c0e152943113941023eb985d2be2448b2895ef6c860407106a89b33d5ae2bb/diff",
  "/var/lib/docker/overlay2/32fc57dd0917ad5acf6beb61b0ff046d118d6c38a01a216076e3d61ac08649b5/diff",
  "/var/lib/docker/overlay2/4eefa20a1e913502b915b9c65d475c6c3a09008f4f2146496cb18c7c8ff82482/diff",
  "/var/lib/docker/overlay2/f5fa44e8b002161869c5e5e4ae9a19fea6378d27f0b567debad68dd72ff127e5/diff",
  "/var/lib/docker/overlay2/4f2cbdf3aee2772f7caa6470f2b0a413ab972620bb4f007af851fb682c0f0ded/diff",
  "/var/lib/docker/overlay2/fd28c533a2d2b73f82ff52c2bfa5652be26f07961022fac547983316c5405068/diff",
  "/var/lib/docker/overlay2/sbr9iduct38yzhynorolioju5/diff",
  "/var/lib/docker/overlay2/tfjkee7mr3ng5i4×1wjw7mjve/diff",
  "/var/lib/docker/overlay2/zc7ln1m6kzw4hc2fbltn1vq3j/diff",
  "/var/lib/docker/overlay2/07hs5m2ow828cc68r328rf94p/diff",
  "/var/lib/docker/overlay2/a30pkgydx8r8ud3oivkx4ea7j/diff",
  "/var/lib/docker/overlay2/iubt0thlvo88i3pifn4tj6pbb/diff",
  "/var/lib/docker/overlay2/fjic757fm1bw2nh613e9nyrqo/diff",
  "/var/lib/docker/overlay2/mse04v1g8434oiw5woqjabp2h/diff",
  "/var/lib/docker/overlay2/inx6s5r51fwv90lmy74z6w678/diff"
]
```

Use `overlay2` to access the files

```
docker run -it --privileged --pid=host debian nsenter -t 1 -m -u -n -i sh
cd /var/lib/docker/overlay2/fjic757fm1bw2nh613e9nyrqo/diff
ls -al
```

```
/var/lib/docker/overlay2/fjic757fm1bw2nh613e9nyrqo/diff # ls -al
total 12
drwxr-xr-x    3 root     root          4096 Apr 17 17:35 .
drwx--x---    4 root     root          4096 Apr 18 21:34 ..
drwxr-xr-x    9 root     root          4096 Apr 17 17:32 app
```

Use `overlay2` to access the files

```
cat app/cat_app/app.py
```

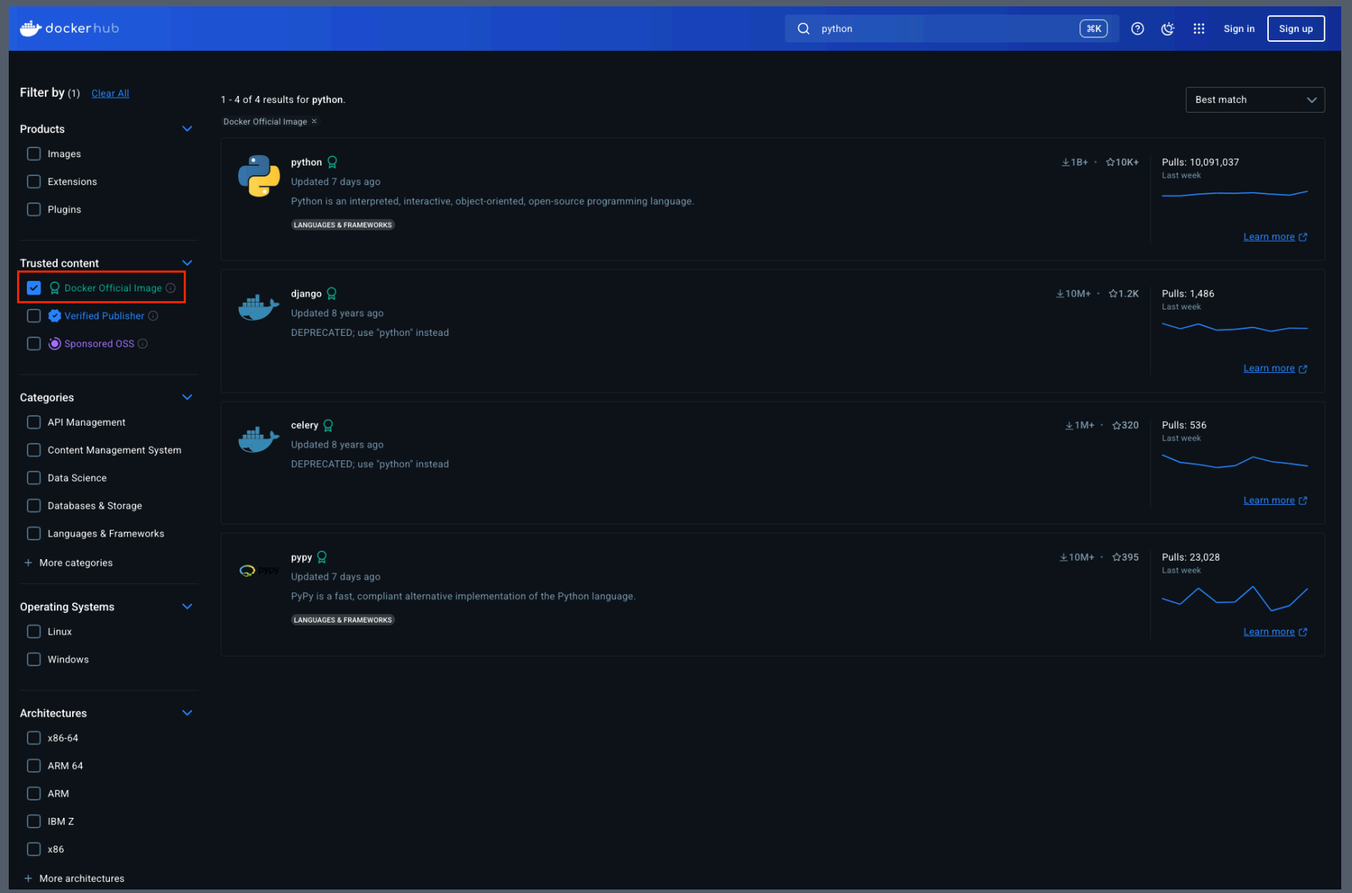
```
/var/lib/docker/overlay2/fjic757fm1bw2nh613e9nyrqo/diff # cat app/cat_app/app.py
from flask import Flask, render_template_string
from random_cat_generator.cat import get_cat_image_url

app = Flask(__name__)

HTML_TEMPLATE = """
<!DOCTYPE html>
<html>
<head>
    <title>Random Cat Photo</title>
</head>
<body>
    <h1>Random Cat Photo</h1>
    {% if image_url %}
      <img src="{{ image_url }}" alt="Cat Image" style="max-width: 100%; height: auto;">
    {% else %}
      <p>Could not load cat image. Please try again.</p>
    {% endif %}
    <br>
    <button onclick="window.location.reload();">Refresh</button>
</body>
</html>
"""


@app.route('/')
def index():
    image_url = get_cat_image_url()
    return render_template_string(HTML_TEMPLATE, image_url=image_url)


if __name__ == "__main__":
    app.run(host="0.0.0.0", port=3000)
/var/lib/docker/overlay2/fjic757fm1bw2nh613e9nyrqo/diff #
```

Improving the Docker image

Improving the Docker image

1. Choose the right base image

Choose your Python base image wisely



Choose your Python base image wisely

Image	Base OS	Size	Features	Best For
`python:3.12`	Debian Bookworm	~1 GB	Full-featured Python + Debian with build tools and utilities preinstalled.	Development and testing, builds with C deps.
`python:3.12-bullseye`	Debian Bullseye	~337 MB	Full Debian-based image with system libraries, compilers, and utilities.	Apps needing full Debian support.
`python:3.12-slim-bullseye`	Debian Bullseye	~42 MB	Minimal Debian-based image without extra utilities. No compilers, fewer dependencies.	Lightweight apps without native compilation.
`python:3.12-bookworm`	Debian Bookworm	~362 MB	Newer Debian-based image with updated packages, more libraries.	Apps needing newer system libraries.
`python:3.12-slim-bookworm`	Debian Bookworm	~43 MB	Minimal Debian Bookworm image without extra system tools.	Production builds with minimal dependencies.
`python:3.12-slim`	Debian (Default)	~43 MB	Currently defaults to Bookworm, identical to `3.12-slim-bookworm`.	General minimal Python image.
`python:3.12-alpine`	Alpine Linux	~17 MB	Extremely lightweight. Uses `musl` instead of `glibc`. Many libraries need recompilation.	Ultra-small images; Python apps that don't require `glibc`.

Image to improve: 0_original.Dockerfile

```
# 0_original.Dockerfile
FROM python:3.12
ARG TOKEN

ENV DEBIAN_FRONTEND=noninteractive
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"

RUN apt-get update
RUN apt-get install -y --no-install-recommends software-properties-common \
    imagemagick \
    build-essential \
    python3-distutils \
    python3-venv \
    python3-pip \
    python3-wheel \
    python3-dev \
    && curl -sS https://bootstrap.pypa.io/get-pip.py | python3

RUN apt-get update
RUN apt-get install --no-install-recommends -y libsm6 \
    libxext6 \
    libxrender-dev \
    libglib2.0-0 \
    libgl1

WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"
RUN poetry install

CMD ["python", "cat_app/app.py"]
```

Image to improve: 0_original.Dockerfile

```
# 0_original.Dockerfile
FROM python:3.12
ARG TOKEN

ENV DEBIAN_FRONTEND=noninteractive
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"

RUN apt-get update
RUN apt-get install -y --no-install-recommends software-properties-common \
    imagemagick \
    build-essential \
    python3-distutils \
    python3-venv \
    python3-pip \
    python3-wheel \
    python3-dev \
    && curl -sS https://bootstrap.pypa.io/get-pip.py | python3

RUN apt-get update
RUN apt-get install --no-install-recommends -y libsm6 \
    libxext6 \
    libxrender-dev \
    libglib2.0-0 \
    libgl1

WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"
RUN poetry install

CMD ["python", "cat_app/app.py"]
```

Choose your Python base image wisely

Instead of this:

```
FROM python:3.12
```

Do this:

```
FROM python:3.12-slim
```

Pin base image version to specific digest

Instead of this:

```
FROM python:3.12-slim
```

Do this:

```
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
```

The right base image: 1_base_image.Dockerfile

```
# 1_base_image.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN

ENV DEBIAN_FRONTEND=noninteractive
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"

RUN apt-get update
RUN apt-get install -y --no-install-recommends software-properties-common \
    imagemagick \
    build-essential \
    python3-distutils \
    python3-venv \
    python3-pip \
    python3-wheel \
    python3-dev \
    && curl -sS https://bootstrap.pypa.io/get-pip.py | python3

RUN apt-get update
RUN apt-get install --no-install-recommends -y libsm6 \
    libxext6 \
    libxrender-dev \
    libglib2.0-0 \
    libgl1

WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"

RUN poetry install

CMD ["python", "cat_app/app.py"]
```

The right base image: 1_base_image.Dockerfile

```
# 1_base_image.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN

ENV DEBIAN_FRONTEND=noninteractive
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"

RUN apt-get update
RUN apt-get install -y --no-install-recommends software-properties-common \
    imagemagick \
    build-essential \
    python3-distutils \
    python3-venv \
    python3-pip \
    python3-wheel \
    python3-dev \
    && curl -sS https://bootstrap.pypa.io/get-pip.py | python3

RUN apt-get update
RUN apt-get install --no-install-recommends -y libsm6 \
    libxext6 \
    libxrender-dev \
    libglib2.0-0 \
    libgl1

WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"

RUN poetry install

CMD ["python", "cat_app/app.py"]
```

Image details

→ Size: 1.5 GB ➡ 976 MB

→ Layers: 16 ➡ 13

→ Build time: ~188 sec ➡ ~370 sec

Improving the Docker image

2. Remove unnecessary packages

Image to improve: 1_base_image.Dockerfile

```
# 1_base_image.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN

ENV DEBIAN_FRONTEND=noninteractive
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"

RUN apt-get update
RUN apt-get install -y --no-install-recommends software-properties-common \
    imagemagick \
    build-essential \
    python3-distutils \
    python3-venv \
    python3-pip \
    python3-wheel \
    python3-dev \
    && curl -sS https://bootstrap.pypa.io/get-pip.py | python3

RUN apt-get update
RUN apt-get install --no-install-recommends -y libsm6 \
    libxext6 \
    libxrender-dev \
    libglib2.0-0 \
    libgl1

WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"

RUN poetry install

CMD ["python", "cat_app/app.py"]
```

Image to improve: 1_base_image.Dockerfile

```
# 1_base_image.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN

ENV DEBIAN_FRONTEND=noninteractive
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"

RUN apt-get update
RUN apt-get install -y --no-install-recommends software-properties-common \
    imagemagick \
    build-essential \
    python3-distutils \
    python3-venv \
    python3-pip \
    python3-wheel \
    python3-dev \
    && curl -sS https://bootstrap.pypa.io/get-pip.py | python3

RUN apt-get update
RUN apt-get install --no-install-recommends -y libsm6 \
    libxext6 \
    libxrender-dev \
    libglib2.0-0 \
    libgl1

WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"

RUN poetry install

CMD ["python", "cat_app/app.py"]
```

Remove unnecessary packages and pin version

Instead of this:

```
RUN apt-get update
RUN apt-get install -y --no-install-recommends software-properties-common \
    imagemagick \
    build-essential \
    python3-distutils \
    python3-venv \
    python3-pip \
    python3-wheel \
    python3-dev \
    && curl -sS https://bootstrap.pypa.io/get-pip.py | python3

RUN apt-get update
RUN apt-get install --no-install-recommends -y libsm6 \
    libxext6 \
    libxrender-dev \
    libglib2.0-0 \
    libgl1
```

Do this:

```
RUN apt-get update
RUN apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2
```

Unnecessary packages removed: 2_remove_packages.Dockerfile

```
# 2_remove_packages.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN

ENV DEBIAN_FRONTEND=noninteractive
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"

RUN apt-get update
RUN apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2

WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"
RUN poetry install

CMD ["python", "cat_app/app.py"]
```

Unnecessary packages removed: 2_remove_packages.Dockerfile

```
# 2_remove_packages.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN

ENV DEBIAN_FRONTEND=noninteractive
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"

RUN apt-get update
RUN apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2

WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"
RUN poetry install

CMD ["python", "cat_app/app.py"]
```

Image details

→ Size: 976 MB ➡ 274 MB

→ Layers: 13 ➡ 11

→ Build time: ~370 sec ➡ ~60 sec

Improving the Docker image

3. Layering

Why you should care about the number of layers

More layers means:

→ **Bigger** image.

→ More **complexity.**

→ Slower **image pull.**

→ Slower **container start** time.

Docker build cache

→ Docker **caches layers.**

→ A layer only has to be rebuilt **if it changed.**

→ **All** other layers **after** the changed layer have to be rebuilt.

The order of instructions in Dockerfiles

→ **Least volatile** instructions **at the top.**

→ **Long running** instructions **at the top.**

Image to improve: 2_remove_packages.Dockerfile

```
# 2_remove_packages.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN

ENV DEBIAN_FRONTEND=noninteractive
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"

RUN apt-get update
RUN apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2

WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"
RUN poetry install

CMD ["python", "cat_app/app.py"]
```

Image to improve: 2_remove_packages.Dockerfile

```
# 2_remove_packages.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN

ENV DEBIAN_FRONTEND=noninteractive
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"

RUN apt-get update
RUN apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2

WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"
RUN poetry install

CMD ["python", "cat_app/app.py"]
```

Do not set `DEBIAN_FRONTEND=noninteractive` with `ENV`

Instead of this:

```
ENV DEBIAN_FRONTEND=noninteractive
```

Do this:

```
ARG DEBIAN_FRONTEND=noninteractive
```

Or this:

```
RUN DEBIAN_FRONTEND=noninteractive apt-get install -y \
    --no-install-recommends libpq5=15.12-0+deb12u2
```

Chain multiple `ENV` instructions into one

Instead of this:

```
ENV POETRY_VERSION="2.1.0"
ENV POETRY_NO_INTERACTION=1
ENV POETRY_VIRTUALENVS_CREATE=0
ENV POETRY_VIRTUALENVS_IN_PROJECT=0

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user"
ENV POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
```

Do this:

```
ENV POETRY_VERSION="2.1.0" \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=0 \
    POETRY_VIRTUALENVS_IN_PROJECT=0 \
    POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user" \
    POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
```

Do not `echo` **secrets**

Do not do this!

```
RUN echo "$POETRY_HTTP_BASIC_FOO_PASSWORD"
```

Chain `update` and `install` in one `RUN` instruction

Instead of this:

```
RUN apt-get update
RUN apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2
```

Do this:

```
RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*
```

How should we layer the `poetry` commands?

As two separate layers?

```
RUN pip install poetry=="$POETRY_VERSION"
RUN poetry install
```

Or as a single layer?

```
RUN pip install poetry=="$POETRY_VERSION" \
    && poetry install
```

How should we layer the `poetry` commands?

→ It depends on how you want to utilize **Docker Cache.**

→ Most of the time the **dependencies** will change **more often** than the `poetry` version. ➡ Keep as **separate** layers.

```
RUN pip install poetry=="$POETRY_VERSION"
RUN poetry install
```

Install Python packages, then copy application code

Instead of this:

```
WORKDIR /app
COPY . .

RUN pip install poetry=="$POETRY_VERSION"
RUN poetry install
```

Do this:

```
RUN pip install poetry=="$POETRY_VERSION"

WORKDIR /app
COPY pyproject.toml poetry.lock ./

RUN poetry install --no-root

COPY cat_app ./cat_app
```

Install Python packages, then copy application code

It is fine to use

```
COPY . .
```

if you maintain a `.dockerignore` file and keep it up to date.

Improved layering: 3_layering.Dockerfile

```
# 3_layering.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=0 \
    POETRY_VIRTUALENVS_IN_PROJECT=0 \
    POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user" \
    POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN pip install poetry=="$POETRY_VERSION"

WORKDIR /app
COPY pyproject.toml poetry.lock ./

RUN poetry install --no-root

COPY cat_app ./cat_app
CMD ["python", "cat_app/app.py"]
```

Improved layering: 3_layering.Dockerfile

```
# 3_layering.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=0 \
    POETRY_VIRTUALENVS_IN_PROJECT=0 \
    POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user" \
    POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN pip install poetry=="$POETRY_VERSION"

WORKDIR /app
COPY pyproject.toml poetry.lock ./

RUN poetry install --no-root

COPY cat_app ./cat_app
CMD ["python", "cat_app/app.py"]
```

Improved layering: 3_layering.Dockerfile

```
# 3_layering.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=0 \
    POETRY_VIRTUALENVS_IN_PROJECT=0 \
    POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user" \
    POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN pip install poetry=="$POETRY_VERSION"

WORKDIR /app
COPY pyproject.toml poetry.lock ./

RUN poetry install --no-root

COPY cat_app ./cat_app
CMD ["python", "cat_app/app.py"]
```

Improved layering: 3_layering.Dockerfile

```
# 3_layering.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=0 \
    POETRY_VIRTUALENVS_IN_PROJECT=0 \
    POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user" \
    POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN pip install poetry=="$POETRY_VERSION"

WORKDIR /app
COPY pyproject.toml poetry.lock ./

RUN poetry install --no-root

COPY cat_app ./cat_app
CMD ["python", "cat_app/app.py"]
```

Improved layering: 3_layering.Dockerfile

```
# 3_layering.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=0 \
    POETRY_VIRTUALENVS_IN_PROJECT=0 \
    POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user" \
    POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN pip install poetry=="$POETRY_VERSION"

WORKDIR /app
COPY pyproject.toml poetry.lock ./

RUN poetry install --no-root

COPY cat_app ./cat_app
CMD ["python", "cat_app/app.py"]
```

Improved layering: 3_layering.Dockerfile

```
# 3_layering.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=0 \
    POETRY_VIRTUALENVS_IN_PROJECT=0 \
    POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user" \
    POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN pip install poetry=="$POETRY_VERSION"

WORKDIR /app
COPY pyproject.toml poetry.lock ./

RUN poetry install --no-root

COPY cat_app ./cat_app
CMD ["python", "cat_app/app.py"]
```

Improved layering: 3_layering.Dockerfile

```
# 3_layering.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=0 \
    POETRY_VIRTUALENVS_IN_PROJECT=0 \
    POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user" \
    POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN pip install poetry=="$POETRY_VERSION"

WORKDIR /app
COPY pyproject.toml poetry.lock ./

RUN poetry install --no-root

COPY cat_app ./cat_app
CMD ["python", "cat_app/app.py"]
```

Improving the Docker image

4. Using Poetry in Docker images

Image to improve: 3_layering.Dockerfile

```
# 3_layering.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=0 \
    POETRY_VIRTUALENVS_IN_PROJECT=0 \
    POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user" \
    POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN pip install poetry=="$POETRY_VERSION"

WORKDIR /app
COPY pyproject.toml poetry.lock ./

RUN poetry install --no-root

COPY cat_app ./cat_app
CMD ["python", "cat_app/app.py"]
```

Image to improve: 3_layering.Dockerfile

```
# 3_layering.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=0 \
    POETRY_VIRTUALENVS_IN_PROJECT=0 \
    POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user" \
    POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN pip install poetry=="$POETRY_VERSION"

WORKDIR /app
COPY pyproject.toml poetry.lock ./

RUN poetry install --no-root

COPY cat_app ./cat_app
CMD ["python", "cat_app/app.py"]
```

Use `pip` to install Poetry[2], but disable `pip` cache

Instead of this:

```
RUN pip install poetry=="$POETRY_VERSION"
```

Do this:

```
RUN pip install --no-cache-dir poetry=="$POETRY_VERSION"
```

[2] Poetry is a Python packaging and dependency manager.

Install Poetry in a dedicated virtual environment

Instead of this:

```
RUN pip install --no-cache-dir poetry=="$POETRY_VERSION"
```

Do this:

```
ENV POETRY_HOME=/opt/poetry

RUN python -m venv $POETRY_HOME \
    && $POETRY_HOME/bin/pip install --no-cache-dir poetry=="$POETRY_VERSION"
```

Install only runtime dependencies

Instead of this:

```
RUN poetry install --no-root
```

Do this:

```
RUN poetry install --no-root --only main
```

Install dependencies in a dedicated virtual environment

Instead of this:

```
ENV POETRY_VIRTUALENVS_CREATE=0 \
    POETRY_VIRTUALENVS_IN_PROJECT=0

RUN poetry install --no-root --only main
```

Do this:

```
ENV POETRY_VIRTUALENVS_CREATE=1 \
    POETRY_VIRTUALENVS_IN_PROJECT=1 \
    PATH="/app/.venv/bin:$PATH"

RUN $POETRY_HOME/bin/poetry install --no-root --only main
```

Delete the Poetry cache directory[3]

Instead of this:

```
RUN $POETRY_HOME/bin/poetry install --no-root --only main
```

Do this:

```
ENV POETRY_CACHE_DIR=/tmp/poetry_cache

RUN $POETRY_HOME/bin/poetry install --no-root --only main \
    && rm -rf $POETRY_CACHE_DIR
```

[3] There is a way to mount the cache directory during the build, see this GitHub discussion.

Proper way of using poetry: 4_poetry.Dockerfile

```
# 4_poetry.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_HOME=/opt/poetry \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=1 \
    POETRY_VIRTUALENVS_IN_PROJECT=1 \
    POETRY_CACHE_DIR=/tmp/poetry_cache \
    POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user" \
    POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN" \
    PATH="/app/.venv/bin:$PATH"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN python -m venv $POETRY_HOME \
    && $POETRY_HOME/bin/pip install --no-cache-dir poetry=="$POETRY_VERSION"

WORKDIR /app
COPY pyproject.toml poetry.lock ./

RUN $POETRY_HOME/bin/poetry install --no-root --only main \
    && rm -rf $POETRY_CACHE_DIR

COPY cat_app ./cat_app
CMD ["python", "cat_app/app.py"]
```

Proper way of using poetry: 4_poetry.Dockerfile

```
# 4_poetry.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_HOME=/opt/poetry \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=1 \
    POETRY_VIRTUALENVS_IN_PROJECT=1 \
    POETRY_CACHE_DIR=/tmp/poetry_cache \
    POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user" \
    POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN" \
    PATH="/app/.venv/bin:$PATH"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN python -m venv $POETRY_HOME \
    && $POETRY_HOME/bin/pip install --no-cache-dir poetry=="$POETRY_VERSION"

WORKDIR /app
COPY pyproject.toml poetry.lock ./

RUN $POETRY_HOME/bin/poetry install --no-root --only main \
    && rm -rf $POETRY_CACHE_DIR

COPY cat_app ./cat_app
CMD ["python", "cat_app/app.py"]
```

Improving the Docker image

5. Secrets

Image to improve: 4_poetry.Dockerfile

```
# 4_poetry.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_HOME=/opt/poetry \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=1 \
    POETRY_VIRTUALENVS_IN_PROJECT=1 \
    POETRY_CACHE_DIR=/tmp/poetry_cache \
    POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user" \
    POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN" \
    PATH="/app/.venv/bin:$PATH"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN python -m venv $POETRY_HOME \
    && $POETRY_HOME/bin/pip install --no-cache-dir poetry=="$POETRY_VERSION"

WORKDIR /app
COPY pyproject.toml poetry.lock ./

RUN $POETRY_HOME/bin/poetry install --no-root --only main \
    && rm -rf $POETRY_CACHE_DIR

COPY cat_app ./cat_app
CMD ["python", "cat_app/app.py"]
```

Image to improve: 4_poetry.Dockerfile

```
# 4_poetry.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_HOME=/opt/poetry \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=1 \
    POETRY_VIRTUALENVS_IN_PROJECT=1 \
    POETRY_CACHE_DIR=/tmp/poetry_cache \
    POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user" \
    POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN" \
    PATH="/app/.venv/bin:$PATH"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN python -m venv $POETRY_HOME \
    && $POETRY_HOME/bin/pip install --no-cache-dir poetry=="$POETRY_VERSION"

WORKDIR /app
COPY pyproject.toml poetry.lock ./

RUN $POETRY_HOME/bin/poetry install --no-root --only main \
    && rm -rf $POETRY_CACHE_DIR

COPY cat_app ./cat_app
CMD ["python", "cat_app/app.py"]
```

Do not set build secrets as ENV or ARG[4]

```
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG TOKEN

ENV POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME="docker-user" \
    POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD="$TOKEN"
```

[4] Also, don't save secrets in a file, it can be extracted from OverlayFS.

Do not set secrets as `ENV`

Even a simple `docker inspect` command **exposes the environment** and the secrets within:

```
"Env": [
    "PATH=/app/.venv/bin:/usr/local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin",
    "LANG=C.UTF-8",
    "GPG_KEY=7169605F62C751356D054A26A821E680E5FA6305",
    "PYTHON_VERSION=3.12.10",
    "PYTHON_SHA256=07ab697474595e06f06647417d3c7fa97ded07afc1a7e4454c5639919b46eaea",
    "POETRY_VERSION=2.1.0",
    "POETRY_HOME=/opt/poetry",
    "POETRY_NO_INTERACTION=1",
    "POETRY_VIRTUALENVS_CREATE=1",
    "POETRY_VIRTUALENVS_IN_PROJECT=1",
    "POETRY_CACHE_DIR=/tmp/poetry_cache",
    "POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=docker-user",
    "POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=DeFxwLAO0aKSnc9bMHHJec8d04MpgDAjqyTwivYQzW8yslkzDUShJQQJ99BCACAAAAAAAAAAAAASAZDOgpe4"
],
```

Use Docker build secrets instead

Mount the required secret(s) to the `RUN` instruction:

```
RUN --mount=type=secret,id=user,target=/root/artifacts/user \
    --mount=type=secret,id=token,target=/root/artifacts/token \
    sh -c 'POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=$(cat /root/artifacts/user) \
           POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=$(cat /root/artifacts/token) \
           $POETRY_HOME/bin/poetry install --no-root --only main && \
           rm -rf $POETRY_CACHE_DIR'
```

Use Docker build secrets instead

Mount the required secret(s) to the `RUN` instruction:

```
RUN --mount=type=secret,id=user,target=/root/artifacts/user \
    --mount=type=secret,id=token,target=/root/artifacts/token \
    sh -c 'POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=$(cat /root/artifacts/user) \
           POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=$(cat /root/artifacts/token) \
           $POETRY_HOME/bin/poetry install --no-root --only main && \
           rm -rf $POETRY_CACHE_DIR'
```

Use Docker build secrets instead

Mount the required secret(s) to the `RUN` instruction:

```
RUN --mount=type=secret,id=user,target=/root/artifacts/user \
    --mount=type=secret,id=token,target=/root/artifacts/token \
    sh -c 'POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=$(cat /root/artifacts/user) \
           POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=$(cat /root/artifacts/token) \
           $POETRY_HOME/bin/poetry install --no-root --only main && \
           rm -rf $POETRY_CACHE_DIR'
```

Proper way of using build secrets: 5_secrets.Dockerfile

```
# 5_secrets.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_HOME=/opt/poetry \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=1 \
    POETRY_VIRTUALENVS_IN_PROJECT=1 \
    POETRY_CACHE_DIR=/tmp/poetry_cache \
    PATH="/app/.venv/bin:$PATH"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN python -m venv $POETRY_HOME \
    && $POETRY_HOME/bin/pip install --no-cache-dir poetry=="$POETRY_VERSION"

WORKDIR /app
COPY pyproject.toml poetry.lock ./

RUN --mount=type=secret,id=user,target=/root/artifacts/user \
    --mount=type=secret,id=token,target=/root/artifacts/token \
    sh -c 'POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=$(cat /root/artifacts/user) \
           POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=$(cat /root/artifacts/token) \
           $POETRY_HOME/bin/poetry install --no-root --only main && \
           rm -rf $POETRY_CACHE_DIR'

COPY cat_app ./cat_app
CMD ["python", "cat_app/app.py"]
```

Proper way of using build secrets: 5_secrets.Dockerfile

```
# 5_secrets.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_HOME=/opt/poetry \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=1 \
    POETRY_VIRTUALENVS_IN_PROJECT=1 \
    POETRY_CACHE_DIR=/tmp/poetry_cache \
    PATH="/app/.venv/bin:$PATH"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN python -m venv $POETRY_HOME \
    && $POETRY_HOME/bin/pip install --no-cache-dir poetry=="$POETRY_VERSION"

WORKDIR /app
COPY pyproject.toml poetry.lock ./

RUN --mount=type=secret,id=user,target=/root/artifacts/user \
    --mount=type=secret,id=token,target=/root/artifacts/token \
    sh -c 'POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=$(cat /root/artifacts/user) \
           POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=$(cat /root/artifacts/token) \
           $POETRY_HOME/bin/poetry install --no-root --only main && \
           rm -rf $POETRY_CACHE_DIR'

COPY cat_app ./cat_app
CMD ["python", "cat_app/app.py"]
```

To build the image use `--secret` option

```
docker build --platform linux/amd64 \
             --secret id=user,env=ADO_USER \
             --secret id=token,env=PAT \
             -f docker/5_secrets.Dockerfile \
             -t 5-secrets .
```

To build the image use `--secret` option

```
docker build --platform linux/amd64 \
             --secret id=user,env=ADO_USER \
             --secret id=token,env=PAT \
             -f docker/5_secrets.Dockerfile \
             -t 5-secrets .
```

Improving the Docker image

6. Non-root user

Do not run your container as ROOT user

Following the Principle of Least Privilege, your container **should not** run as ROOT user.

```
docker run -it --rm --platform linux/amd64 4-poetry bash
```

```
➜  ~ docker run -it --rm --platform linux/amd64 4-poetry bash
root@0d6c1a611b02:/app# id
uid=0(root) gid=0(root) groups=0(root)
root@0d6c1a611b02:/app#
```

Do not run your container as ROOT user

→ Create **a non-root user**[5].

→ Switch to it by using the USER instruction.

```
ENV DOCKER_USER=appuser \
    DOCKER_GROUP=appuser \
    UID=1001 \
    GID=1001

RUN groupadd -g "$GID" "$DOCKER_GROUP" && \
    useradd -l -u "$UID" -g "$DOCKER_GROUP" -s /bin/sh -m "$DOCKER_USER"

# ROOT-level instructions should come here

USER "$DOCKER_USER"
```

[5] To learn more about the USER instruction, read here.

Setting non-root user: 6_non_root_user.Dockerfile

```
# 6_non_root_user.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_HOME=/opt/poetry \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=1 \
    POETRY_VIRTUALENVS_IN_PROJECT=1 \
    POETRY_CACHE_DIR=/tmp/poetry_cache \
    PATH="/app/.venv/bin:$PATH" \
    DOCKER_USER=appuser \
    DOCKER_GROUP=appuser \
    UID=1001 \
    GID=1001

RUN groupadd -g "$GID" "$DOCKER_GROUP" && \
    useradd -l -u "$UID" -g "$DOCKER_GROUP" -s /bin/sh -m "$DOCKER_USER"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN python -m venv $POETRY_HOME \
    && $POETRY_HOME/bin/pip install --no-cache-dir poetry=="$POETRY_VERSION"

WORKDIR /app
COPY pyproject.toml poetry.lock ./

RUN --mount=type=secret,id=user,target=/root/artifacts/user \
    --mount=type=secret,id=token,target=/root/artifacts/token \
    sh -c 'POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=$(cat /root/artifacts/user) \
           POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=$(cat /root/artifacts/token) \
           $POETRY_HOME/bin/poetry install --no-root --only main && \
           rm -rf $POETRY_CACHE_DIR'

COPY cat_app ./cat_app

USER "$DOCKER_USER"
CMD ["python", "cat_app/app.py"]
```

Setting non-root user: 6_non_root_user.Dockerfile

```
# 6_non_root_user.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_HOME=/opt/poetry \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=1 \
    POETRY_VIRTUALENVS_IN_PROJECT=1 \
    POETRY_CACHE_DIR=/tmp/poetry_cache \
    PATH="/app/.venv/bin:$PATH" \
    DOCKER_USER=appuser \
    DOCKER_GROUP=appuser \
    UID=1001 \
    GID=1001

RUN groupadd -g "$GID" "$DOCKER_GROUP" && \
    useradd -l -u "$UID" -g "$DOCKER_GROUP" -s /bin/sh -m "$DOCKER_USER"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN python -m venv $POETRY_HOME \
    && $POETRY_HOME/bin/pip install --no-cache-dir poetry=="$POETRY_VERSION"

WORKDIR /app
COPY pyproject.toml poetry.lock ./

RUN --mount=type=secret,id=user,target=/root/artifacts/user \
    --mount=type=secret,id=token,target=/root/artifacts/token \
    sh -c 'POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=$(cat /root/artifacts/user) \
           POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=$(cat /root/artifacts/token) \
           $POETRY_HOME/bin/poetry install --no-root --only main && \
           rm -rf $POETRY_CACHE_DIR'

COPY cat_app ./cat_app

USER "$DOCKER_USER"
CMD ["python", "cat_app/app.py"]
```

Setting non-root user: 6_non_root_user.Dockerfile

Your container will now start as `appuser`.

```
docker run -it --rm --platform linux/amd64 6-non-root-user bash
```

```
➜  ~ docker run -it --rm --platform linux/amd64 6-non-root-user bash
appuser@4a28d5a9677a:/app$ id
uid=1001(appuser) gid=1001(appuser) groups=1001(appuser)
appuser@4a28d5a9677a:/app$
```

Improving the Docker image

7. Bind mounts

Image to improve: 6_non_root_user.Dockerfile

```
# 6_non_root_user.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_HOME=/opt/poetry \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=1 \
    POETRY_VIRTUALENVS_IN_PROJECT=1 \
    POETRY_CACHE_DIR=/tmp/poetry_cache \
    PATH="/app/.venv/bin:$PATH" \
    DOCKER_USER=appuser \
    DOCKER_GROUP=appuser \
    UID=1001 \
    GID=1001

RUN groupadd -g "$GID" "$DOCKER_GROUP" && \
    useradd -l -u "$UID" -g "$DOCKER_GROUP" -s /bin/sh -m "$DOCKER_USER"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN python -m venv $POETRY_HOME \
    && $POETRY_HOME/bin/pip install --no-cache-dir poetry=="$POETRY_VERSION"

WORKDIR /app
COPY pyproject.toml poetry.lock ./

RUN --mount=type=secret,id=user,target=/root/artifacts/user \
    --mount=type=secret,id=token,target=/root/artifacts/token \
    sh -c 'POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=$(cat /root/artifacts/user) \
           POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=$(cat /root/artifacts/token) \
           $POETRY_HOME/bin/poetry install --no-root --only main && \
           rm -rf $POETRY_CACHE_DIR'

COPY cat_app ./cat_app

USER "$DOCKER_USER"
CMD ["python", "cat_app/app.py"]
```

Image to improve: 6_non_root_user.Dockerfile

```
# 6_non_root_user.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_HOME=/opt/poetry \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=1 \
    POETRY_VIRTUALENVS_IN_PROJECT=1 \
    POETRY_CACHE_DIR=/tmp/poetry_cache \
    PATH="/app/.venv/bin:$PATH" \
    DOCKER_USER=appuser \
    DOCKER_GROUP=appuser \
    UID=1001 \
    GID=1001

RUN groupadd -g "$GID" "$DOCKER_GROUP" && \
    useradd -l -u "$UID" -g "$DOCKER_GROUP" -s /bin/sh -m "$DOCKER_USER"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN python -m venv $POETRY_HOME \
    && $POETRY_HOME/bin/pip install --no-cache-dir poetry=="$POETRY_VERSION"

WORKDIR /app
COPY pyproject.toml poetry.lock ./

RUN --mount=type=secret,id=user,target=/root/artifacts/user \
    --mount=type=secret,id=token,target=/root/artifacts/token \
    sh -c 'POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=$(cat /root/artifacts/user) \
           POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=$(cat /root/artifacts/token) \
           $POETRY_HOME/bin/poetry install --no-root --only main && \
           rm -rf $POETRY_CACHE_DIR'

COPY cat_app ./cat_app

USER "$DOCKER_USER"
CMD ["python", "cat_app/app.py"]
```

Use bind mounts instead of `COPY`

→ For files that are only needed to **execute a** `RUN` instruction.

→ Bind-mounted files **do not persist** in the final image.

Use bind mounts instead of COPY

Instead of this:

```
COPY pyproject.toml poetry.lock ./

RUN --mount=type=secret,id=user,target=/root/artifacts/user \
    --mount=type=secret,id=token,target=/root/artifacts/token \
    sh -c 'POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=$(cat /root/artifacts/user) \
           POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=$(cat /root/artifacts/token) \
           $POETRY_HOME/bin/poetry install --no-root --only main && \
           rm -rf $POETRY_CACHE_DIR'
```

Do this:

```
RUN --mount=type=secret,id=user,target=/root/artifacts/user \
    --mount=type=secret,id=token,target=/root/artifacts/token \
    --mount=type=bind,source=pyproject.toml,target=pyproject.toml \
    --mount=type=bind,source=poetry.lock,target=poetry.lock \
    sh -c 'POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=$(cat /root/artifacts/user) \
           POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=$(cat /root/artifacts/token) \
           $POETRY_HOME/bin/poetry install --no-root --only main && \
           rm -rf $POETRY_CACHE_DIR'
```

Use bind mounts instead of COPY

```
RUN --mount=type=secret,id=user,target=/root/artifacts/user \
    --mount=type=secret,id=token,target=/root/artifacts/token \
    --mount=type=bind,source=pyproject.toml,target=pyproject.toml \
    --mount=type=bind,source=poetry.lock,target=poetry.lock \
    sh -c 'POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=$(cat /root/artifacts/user) \
           POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=$(cat /root/artifacts/token) \
           $POETRY_HOME/bin/poetry install --no-root --only main && \
           rm -rf $POETRY_CACHE_DIR'
```

Use bind mounts instead of COPY

```
RUN --mount=type=secret,id=user,target=/root/artifacts/user \
    --mount=type=secret,id=token,target=/root/artifacts/token \
    --mount=type=bind,source=pyproject.toml,target=pyproject.toml \
    --mount=type=bind,source=poetry.lock,target=poetry.lock \
    sh -c 'POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=$(cat /root/artifacts/user) \
           POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=$(cat /root/artifacts/token) \
           $POETRY_HOME/bin/poetry install --no-root --only main && \
           rm -rf $POETRY_CACHE_DIR'
```

With bind mount: 7_bind_mount.Dockerfile

```
# 7_bind_mount.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_HOME=/opt/poetry \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=1 \
    POETRY_VIRTUALENVS_IN_PROJECT=1 \
    POETRY_CACHE_DIR=/tmp/poetry_cache \
    PATH="/app/.venv/bin:$PATH" \
    DOCKER_USER=appuser \
    DOCKER_GROUP=appuser \
    UID=1001 \
    GID=1001

RUN groupadd -g "$GID" "$DOCKER_GROUP" && \
    useradd -l -u "$UID" -g "$DOCKER_GROUP" -s /bin/sh -m "$DOCKER_USER"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN python -m venv $POETRY_HOME \
    && $POETRY_HOME/bin/pip install --no-cache-dir poetry=="$POETRY_VERSION"

WORKDIR /app

RUN --mount=type=secret,id=user,target=/root/artifacts/user \
    --mount=type=secret,id=token,target=/root/artifacts/token \
    --mount=type=bind,source=pyproject.toml,target=pyproject.toml \
    --mount=type=bind,source=poetry.lock,target=poetry.lock \
    sh -c 'POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=$(cat /root/artifacts/user) \
           POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=$(cat /root/artifacts/token) \
           $POETRY_HOME/bin/poetry install --no-root --only main && \
           rm -rf $POETRY_CACHE_DIR'

COPY cat_app ./cat_app

USER "$DOCKER_USER"
CMD ["python", "cat_app/app.py"]
```

With bind mount: 7_bind_mount.Dockerfile

```
# 7_bind_mount.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_HOME=/opt/poetry \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=1 \
    POETRY_VIRTUALENVS_IN_PROJECT=1 \
    POETRY_CACHE_DIR=/tmp/poetry_cache \
    PATH="/app/.venv/bin:$PATH" \
    DOCKER_USER=appuser \
    DOCKER_GROUP=appuser \
    UID=1001 \
    GID=1001

RUN groupadd -g "$GID" "$DOCKER_GROUP" && \
    useradd -l -u "$UID" -g "$DOCKER_GROUP" -s /bin/sh -m "$DOCKER_USER"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN python -m venv $POETRY_HOME \
    && $POETRY_HOME/bin/pip install --no-cache-dir poetry=="$POETRY_VERSION"

WORKDIR /app

RUN --mount=type=secret,id=user,target=/root/artifacts/user \
    --mount=type=secret,id=token,target=/root/artifacts/token \
    --mount=type=bind,source=pyproject.toml,target=pyproject.toml \
    --mount=type=bind,source=poetry.lock,target=poetry.lock \
    sh -c 'POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=$(cat /root/artifacts/user) \
           POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=$(cat /root/artifacts/token) \
           $POETRY_HOME/bin/poetry install --no-root --only main && \
           rm -rf $POETRY_CACHE_DIR'

COPY cat_app ./cat_app

USER "$DOCKER_USER"
CMD ["python", "cat_app/app.py"]
```

Improving the Docker image

8. Multi-stage build

Benefits of multi-stage builds

- → **Smaller** image size.
- → **Cleaner separation** between the building of your image and the final output.
- → **Reusable stages** with shared components for multiple images.
- → **Improved security** due to a smaller attack surface.
- → **Faster** builds.

Image to improve: 7_bind_mount.Dockerfile

```
# 7_bind_mount.Dockerfile
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57
ARG DEBIAN_FRONTEND=noninteractive

ENV POETRY_VERSION="2.1.0" \
    POETRY_HOME=/opt/poetry \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=1 \
    POETRY_VIRTUALENVS_IN_PROJECT=1 \
    POETRY_CACHE_DIR=/tmp/poetry_cache \
    PATH="/app/.venv/bin:$PATH" \
    DOCKER_USER=appuser \
    DOCKER_GROUP=appuser \
    UID=1001 \
    GID=1001

RUN groupadd -g "$GID" "$DOCKER_GROUP" && \
    useradd -l -u "$UID" -g "$DOCKER_GROUP" -s /bin/sh -m "$DOCKER_USER"

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

RUN python -m venv $POETRY_HOME \
    && $POETRY_HOME/bin/pip install --no-cache-dir poetry=="$POETRY_VERSION"

WORKDIR /app

RUN --mount=type=secret,id=user,target=/root/artifacts/user \
    --mount=type=secret,id=token,target=/root/artifacts/token \
    --mount=type=bind,source=pyproject.toml,target=pyproject.toml \
    --mount=type=bind,source=poetry.lock,target=poetry.lock \
    sh -c 'POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=$(cat /root/artifacts/user) \
           POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=$(cat /root/artifacts/token) \
           $POETRY_HOME/bin/poetry install --no-root --only main && \
           rm -rf $POETRY_CACHE_DIR'

COPY cat_app ./cat_app

USER "$DOCKER_USER"
CMD ["python", "cat_app/app.py"]
```

The multi-stage Dockerfile: Base stage

→ Reusable base stage.

→ Install **Linux packages** needed for **runtime**.

The multi-stage Dockerfile: Base stage

```
ARG INSTALLER_VENV_PATH=/build/.venv

FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57 as base
ARG DEBIAN_FRONTEND=noninteractive

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*
```

The multi-stage Dockerfile: Installer stage

→ Install packages that are not needed for runtime but are **needed for build** (e.g. Poetry).

→ Install **Python packages** that are needed for runtime.

The multi-stage Dockerfile: Installer stage

```
FROM base as installer
ARG INSTALLER_VENV_PATH

ENV POETRY_VERSION="2.1.0" \
    POETRY_HOME=/opt/poetry \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=1 \
    POETRY_VIRTUALENVS_IN_PROJECT=1 \
    POETRY_CACHE_DIR=/tmp/poetry_cache \
    PATH="${INSTALLER_VENV_PATH}/bin:$PATH"

RUN python -m venv $POETRY_HOME \
    && $POETRY_HOME/bin/pip install --no-cache-dir poetry=="$POETRY_VERSION"

WORKDIR /build

RUN --mount=type=secret,id=user,target=/root/artifacts/user \
    --mount=type=secret,id=token,target=/root/artifacts/token \
    --mount=type=bind,source=pyproject.toml,target=pyproject.toml \
    --mount=type=bind,source=poetry.lock,target=poetry.lock \
    sh -c 'POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=$(cat /root/artifacts/user) \
           POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=$(cat /root/artifacts/token) \
           $POETRY_HOME/bin/poetry install --no-root --only main && \
           rm -rf $POETRY_CACHE_DIR'
```

The multi-stage Dockerfile: Runner stage

→ Start from the **base stage**.

→ **Copy Python packages** needed for runtime from installer stage.

→ Set **non-root user**.

The multi-stage Dockerfile: Runner stage

```
FROM base as runner
ARG INSTALLER_VENV_PATH
ARG WORKDIR=/app

ENV PATH="${WORKDIR}/.venv/bin:$PATH" \
    DOCKER_USER=appuser \
    DOCKER_GROUP=appuser \
    UID=1001 \
    GID=1001

RUN groupadd -g "$GID" "$DOCKER_GROUP" && \
    useradd -l -u "$UID" -g "$DOCKER_GROUP" -s /bin/sh -m "$DOCKER_USER"

COPY --from=installer $INSTALLER_VENV_PATH $WORKDIR/.venv

WORKDIR $WORKDIR
COPY cat_app ./cat_app

USER "$DOCKER_USER"
CMD ["python", "cat_app/app.py"]
```

The multi-stage Dockerfile: Runner stage

```
FROM base as runner
ARG INSTALLER_VENV_PATH
ARG WORKDIR=/app

ENV PATH="${WORKDIR}/.venv/bin:$PATH" \
    DOCKER_USER=appuser \
    DOCKER_GROUP=appuser \
    UID=1001 \
    GID=1001

RUN groupadd -g "$GID" "$DOCKER_GROUP" && \
    useradd -l -u "$UID" -g "$DOCKER_GROUP" -s /bin/sh -m "$DOCKER_USER"

COPY --from=installer $INSTALLER_VENV_PATH $WORKDIR/.venv

WORKDIR $WORKDIR
COPY cat_app ./cat_app

USER "$DOCKER_USER"
CMD ["python", "cat_app/app.py"]
```

The multi-stage Dockerfile: Runner stage

```
FROM base as runner
ARG INSTALLER_VENV_PATH
ARG WORKDIR=/app

ENV PATH="${WORKDIR}/.venv/bin:$PATH" \
    DOCKER_USER=appuser \
    DOCKER_GROUP=appuser \
    UID=1001 \
    GID=1001

RUN groupadd -g "$GID" "$DOCKER_GROUP" && \
    useradd -l -u "$UID" -g "$DOCKER_GROUP" -s /bin/sh -m "$DOCKER_USER"

COPY --from=installer $INSTALLER_VENV_PATH $WORKDIR/.venv

WORKDIR $WORKDIR
COPY cat_app ./cat_app

USER "$DOCKER_USER"
CMD ["python", "cat_app/app.py"]
```

The multi-stage Dockerfile: Runner stage

```
FROM base as runner
ARG INSTALLER_VENV_PATH
ARG WORKDIR=/app

ENV PATH="${WORKDIR}/.venv/bin:$PATH" \
    DOCKER_USER=appuser \
    DOCKER_GROUP=appuser \
    UID=1001 \
    GID=1001

RUN groupadd -g "$GID" "$DOCKER_GROUP" && \
    useradd -l -u "$UID" -g "$DOCKER_GROUP" -s /bin/sh -m "$DOCKER_USER"

COPY --from=installer $INSTALLER_VENV_PATH $WORKDIR/.venv

WORKDIR $WORKDIR
COPY cat_app ./cat_app

USER "$DOCKER_USER"
CMD ["python", "cat_app/app.py"]
```

The multi-stage Dockerfile: Runner stage

```
FROM base as runner
ARG INSTALLER_VENV_PATH
ARG WORKDIR=/app

ENV PATH="${WORKDIR}/.venv/bin:$PATH" \
    DOCKER_USER=appuser \
    DOCKER_GROUP=appuser \
    UID=1001 \
    GID=1001

RUN groupadd -g "$GID" "$DOCKER_GROUP" && \
    useradd -l -u "$UID" -g "$DOCKER_GROUP" -s /bin/sh -m "$DOCKER_USER"

COPY --from=installer $INSTALLER_VENV_PATH $WORKDIR/.venv

WORKDIR $WORKDIR
COPY cat_app ./cat_app

USER "$DOCKER_USER"
CMD ["python", "cat_app/app.py"]
```

The multi-stage Dockerfile: Runner stage

```
FROM base as runner
ARG INSTALLER_VENV_PATH
ARG WORKDIR=/app

ENV PATH="${WORKDIR}/.venv/bin:$PATH" \
    DOCKER_USER=appuser \
    DOCKER_GROUP=appuser \
    UID=1001 \
    GID=1001

RUN groupadd -g "$GID" "$DOCKER_GROUP" && \
    useradd -l -u "$UID" -g "$DOCKER_GROUP" -s /bin/sh -m "$DOCKER_USER"

COPY --from=installer $INSTALLER_VENV_PATH $WORKDIR/.venv

WORKDIR $WORKDIR
COPY cat_app ./cat_app

USER "$DOCKER_USER"
CMD ["python", "cat_app/app.py"]
```

The multi-stage Dockerfile: 8_multi_stage.Dockerfile

```
# 8_multi_stage.Dockerfile
ARG INSTALLER_VENV_PATH=/build/.venv

# ──────────────────────────────── Base stage ─────────────────────────────────
FROM python:3.12-slim@sha256:85824326bc4ae27a1abb5bc0dd9e08847aa5fe73d8afb593b1b45b7cb4180f57 as base
ARG DEBIAN_FRONTEND=noninteractive

RUN apt-get update \
    && apt-get install -y --no-install-recommends libpq5=15.12-0+deb12u2 \
    && rm -rf /var/lib/apt/lists/*

# ──────────────────────────────── Installer stage ────────────────────────────
FROM base as installer
ARG INSTALLER_VENV_PATH

ENV POETRY_VERSION="2.1.0" \
    POETRY_HOME=/opt/poetry \
    POETRY_NO_INTERACTION=1 \
    POETRY_VIRTUALENVS_CREATE=1 \
    POETRY_VIRTUALENVS_IN_PROJECT=1 \
    POETRY_CACHE_DIR=/tmp/poetry_cache \
    PATH="${INSTALLER_VENV_PATH}/bin:$PATH"

RUN python -m venv $POETRY_HOME \
    && $POETRY_HOME/bin/pip install --no-cache-dir poetry=="$POETRY_VERSION"

WORKDIR /build

RUN --mount=type=secret,id=user,target=/root/artifacts/user \
    --mount=type=secret,id=token,target=/root/artifacts/token \
    --mount=type=bind,source=pyproject.toml,target=pyproject.toml \
    --mount=type=bind,source=poetry.lock,target=poetry.lock \
    sh -c 'POETRY_HTTP_BASIC_PYTHONMONTY_USERNAME=$(cat /root/artifacts/user) \
           POETRY_HTTP_BASIC_PYTHONMONTY_PASSWORD=$(cat /root/artifacts/token) \
           $POETRY_HOME/bin/poetry install --no-root --only main && \
           rm -rf $POETRY_CACHE_DIR'

# ──────────────────────────────── Runner stage ───────────────────────────────
FROM base as runner
ARG INSTALLER_VENV_PATH
ARG WORKDIR=/app

ENV PATH="${WORKDIR}/.venv/bin:$PATH" \
    DOCKER_USER=appuser \
    DOCKER_GROUP=appuser \
    UID=1001 \
    GID=1001

RUN groupadd -g "$GID" "$DOCKER_GROUP" && \
    useradd -l -u "$UID" -g "$DOCKER_GROUP" -s /bin/sh -m "$DOCKER_USER"

COPY --from=installer $INSTALLER_VENV_PATH $WORKDIR/.venv

WORKDIR $WORKDIR
COPY cat_app ./cat_app

USER "$DOCKER_USER"
CMD ["python", "cat_app/app.py"]
```

Image details

→ Size: 263 MB ➡ 136 MB

→ Layers: 10 ➡ 9

→ Build time: ~75 sec ➡ without cache ~70 sec ➡ with cache ~2 sec

Security tools

Security tools

1. Hadolint

Lint your Dockerfile with Hadolint

→ Hadolint: a linter that helps you **check building best practices** in your Dockerfiles.

→ It can be easily integrated with most **CI pipelines** (i.e. GitHub Actions, GitLab CI).

→ Available as a **pre-commit hook.**

Use Hadolint on 0_original.Dockerfile

```
hadolint docker/0_original.Dockerfile
```

```
➜  inspect-docker-images git:(main) ✗ hadolint docker/0_original.Dockerfile
docker/0_original.Dockerfile:14 DL3009 info: Delete the apt-get lists after installing something
docker/0_original.Dockerfile:14 DL3059 info: Multiple consecutive `RUN` instructions. Consider consolidation.
docker/0_original.Dockerfile:15 DL3008 warning: Pin versions in apt get install. Instead of `apt-get install <package>` use `apt-get install <package>=<version
>`
docker/0_original.Dockerfile:15 DL4006 warning: Set the SHELL option -o pipefail before RUN with a pipe in it. If you are using /bin/sh in an alpine image or i
f your shell is symlinked to busybox then consider explicitly setting your SHELL to /bin/ash, or disable this check
docker/0_original.Dockerfile:25 DL3009 info: Delete the apt-get lists after installing something
docker/0_original.Dockerfile:26 DL3059 info: Multiple consecutive `RUN` instructions. Consider consolidation.
docker/0_original.Dockerfile:26 DL3008 warning: Pin versions in apt get install. Instead of `apt-get install <package>` use `apt-get install <package>=<version
>`
docker/0_original.Dockerfile:35 DL3042 warning: Avoid use of cache directory with pip. Use `pip install --no-cache-dir <package>`
docker/0_original.Dockerfile:36 DL3059 info: Multiple consecutive `RUN` instructions. Consider consolidation.
➜  inspect-docker-images git:(main) ✗
```

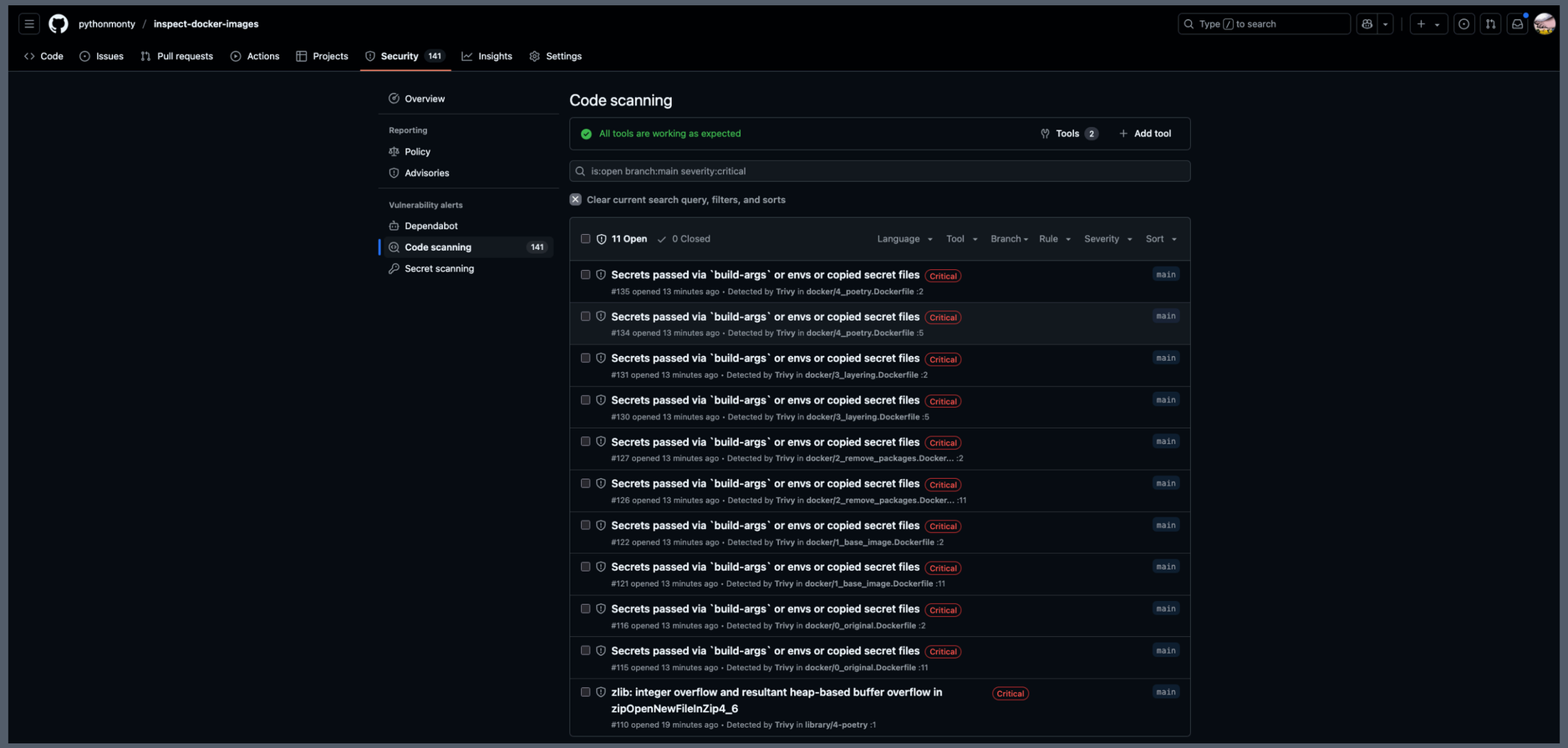
Security tools

2. Trivy

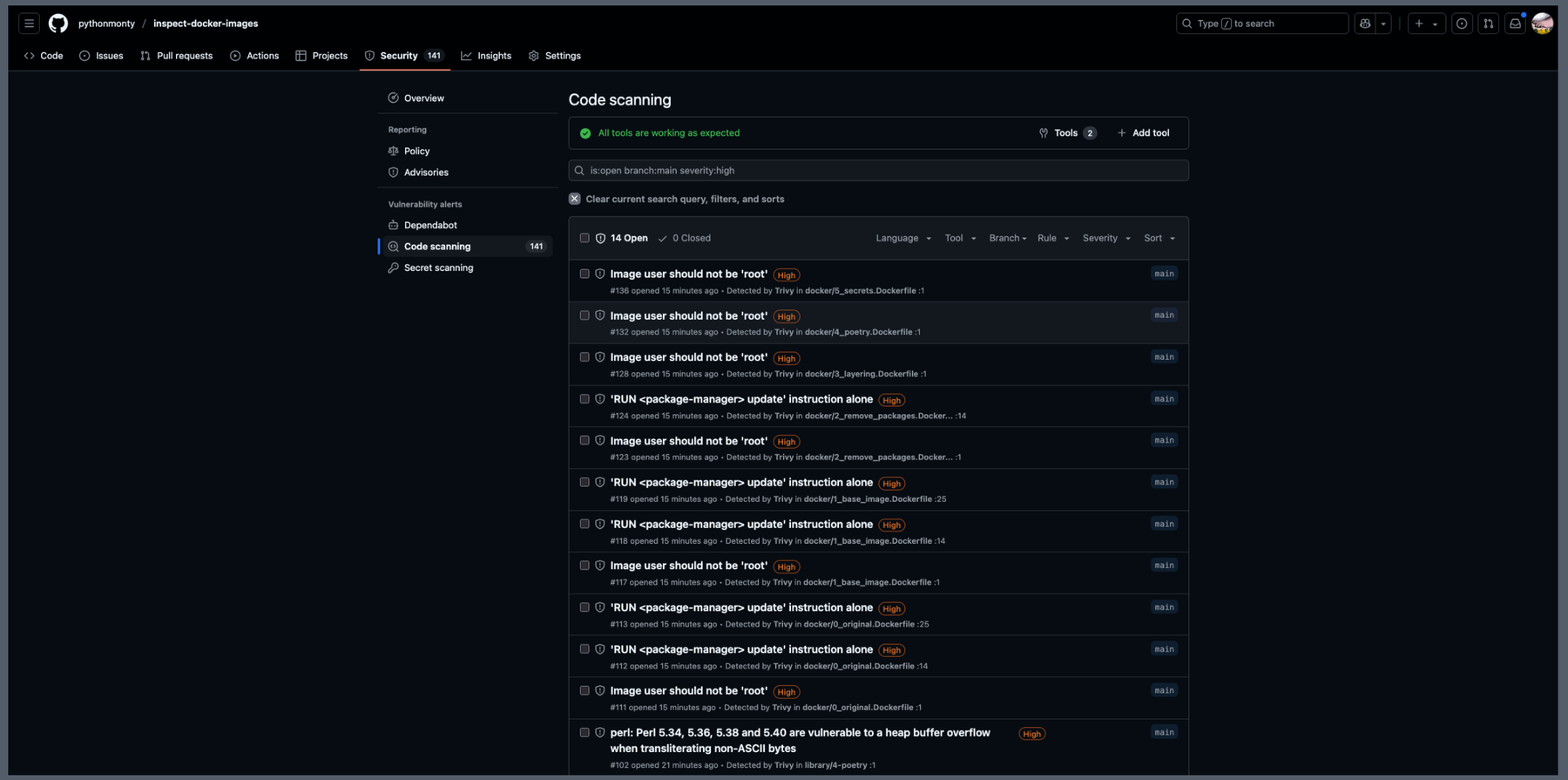
Trivy

- → Trivy is an open-source security scanner.
- → Scans for **vulnerabilities** in OS **packages** and application **dependencies.**
- → Scans for **misconfigurations** in images.
- → It can be easily integrated with most **CI pipelines** (i.e. GitHub Actions, GitLab CI).

Scanning images with Trivy - Severity: CRITICAL



Scanning images with Trivy - Severity: HIGH



Q & A

Clone the repository

https://github.com/pythonmonty/inspect-docker-images