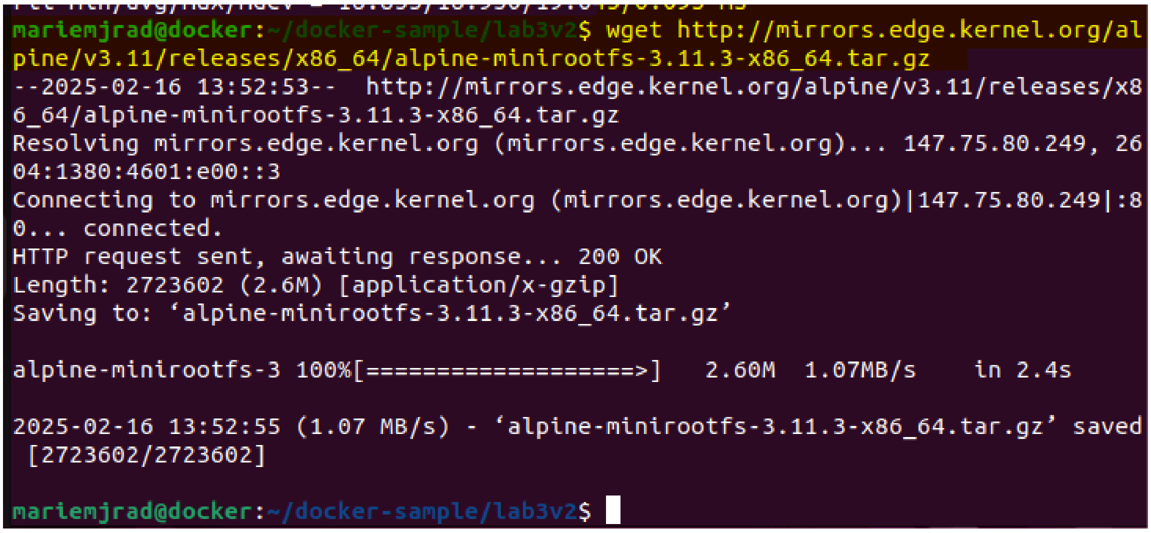
# Image From Scratch

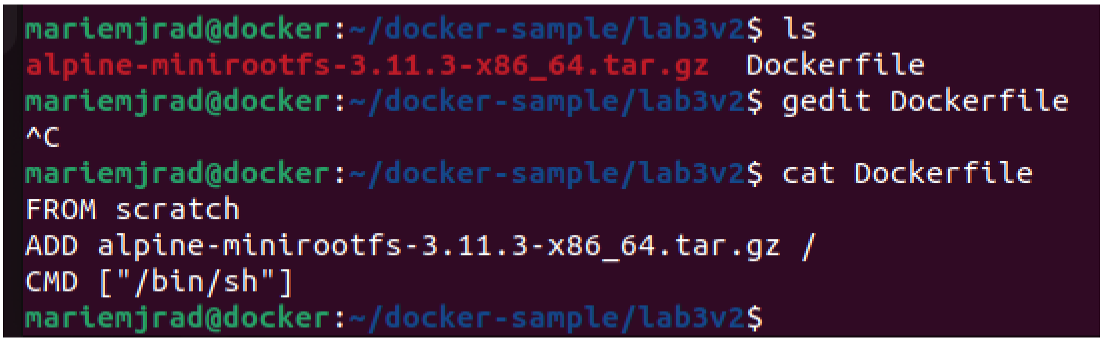
- downloading an alpine mirror tar to deploy it later on the container based on a **from scratch image**



```
mariemjrad@docker:~/docker-sample/lab3v2$ wget http://mirrors.edge.kernel.org/al
pine/v3.11/releases/x86_64/alpine-minirootfs-3.11.3-x86_64.tar.gz
--2025-02-16 13:52:53--  http://mirrors.edge.kernel.org/alpine/v3.11/releases/x8
6_64/alpine-minirootfs-3.11.3-x86_64.tar.gz
Resolving mirrors.edge.kernel.org (mirrors.edge.kernel.org)... 147.75.80.249, 26
04:1380:4601:e00::3
Connecting to mirrors.edge.kernel.org (mirrors.edge.kernel.org)|147.75.80.249|:8
0... connected.
HTTP request sent, awaiting response... 200 OK
Length: 2723602 (2.6M) [application/x-gzip]
Saving to: ‘alpine-minirootfs-3.11.3-x86_64.tar.gz’

alpine-minirootfs-3 100%[===================>]   2.60M  1.07MB/s    in 2.4s

2025-02-16 13:52:55 (1.07 MB/s) - ‘alpine-minirootfs-3.11.3-x86_64.tar.gz’ saved
 [2723602/2723602]

mariemjrad@docker:~/docker-sample/lab3v2$
```

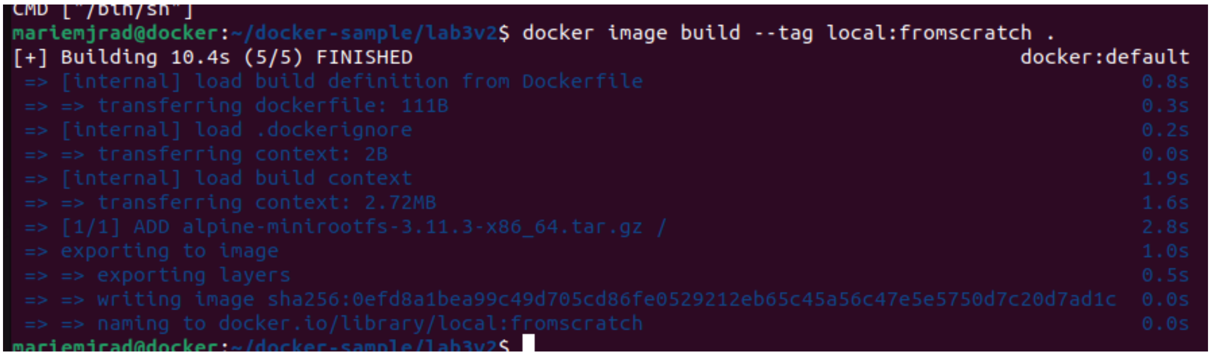
- ***Alpine MinirootFS** is a minimal root filesystem (rootfs) of **Alpine Linux**, designed for lightweight and containerized environments. It provides just enough of the Alpine Linux userland to boot into a working system, making it ideal for use in **containers, chroot environments, embedded systems, or custom minimal installation***

- Copying this minimal OS tar file to the root directory of the futurely created containers

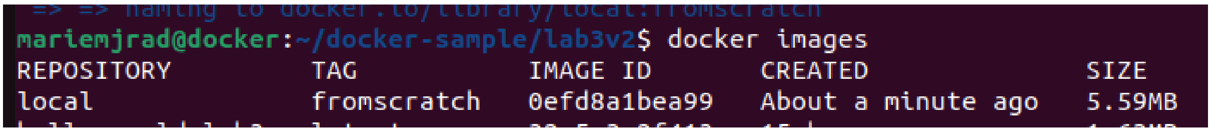


```
mariemjrad@docker:~/docker-sample/lab3v2$ ls
alpine-minirootfs-3.11.3-x86_64.tar.gz  Dockerfile
mariemjrad@docker:~/docker-sample/lab3v2$ gedit Dockerfile
^C
mariemjrad@docker:~/docker-sample/lab3v2$ cat Dockerfile
FROM scratch
ADD alpine-minirootfs-3.11.3-x86_64.tar.gz /
CMD ["/bin/sh"]
mariemjrad@docker:~/docker-sample/lab3v2$
```

- Building the image and named it local

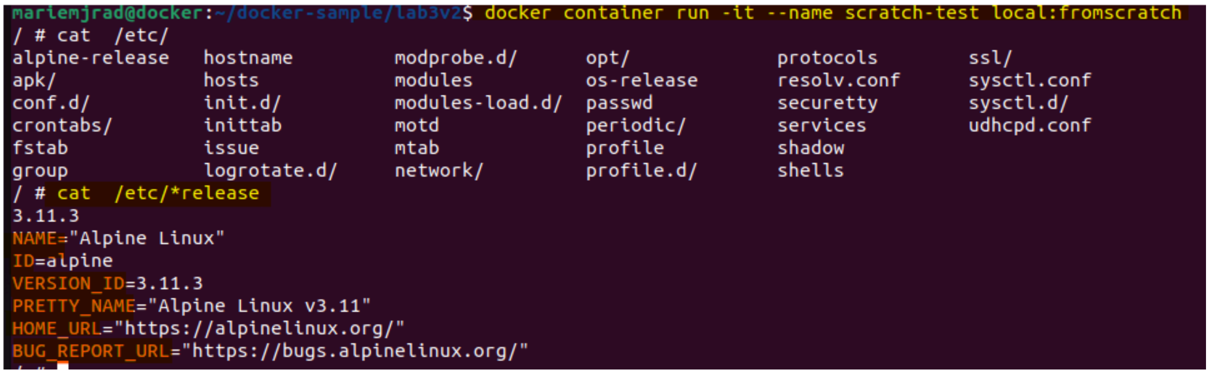


```
CMD ["/bin/sh"]
mariemjrad@docker:~/docker-sample/lab3v2$ docker image build --tag local:fromscratch .
[+] Building 10.4s (5/5) FINISHED                                          docker:default
 => [internal] load build definition from Dockerfile                                 0.8s
 => => transferring dockerfile: 111B                                                 0.3s
 => [internal] load .dockerignore                                                    0.2s
 => => transferring context: 2B                                                      0.0s
 => [internal] load build context                                                    1.9s
 => => transferring context: 2.72MB                                                  1.6s
 => [1/1] ADD alpine-minirootfs-3.11.3-x86_64.tar.gz /                               2.8s
 => exporting to image                                                               1.0s
 => => exporting layers                                                              0.5s
 => => writing image sha256:0efd8a1bea99c49d705cd86fe0529212eb65c45a56c47e5e5750d7c20d7ad1c  0.0s
 => => naming to docker.io/library/local:fromscratch                                 0.0s
mariemjrad@docker:~/docker-sample/lab3v2$
```



```
 => => naming to docker.io/library/local:fromscratch
mariemjrad@docker:~/docker-sample/lab3v2$ docker images
REPOSITORY           TAG           IMAGE ID       CREATED              SIZE
local                fromscratch   0efd8a1bea99   About a minute ago   5.59MB
```

- Display OS information in the container=local



```
mariemjrad@docker:~/docker-sample/lab3v2$ docker container run -it --name scratch-test local:fromscratch
/ # cat  /etc/
alpine-release   hostname         modprobe.d/      opt/             protocols        ssl/
apk/             hosts            modules          os-release       resolv.conf      sysctl.conf
conf.d/          init.d/          modules-load.d/  passwd           securetty        sysctl.d/
crontabs/        inittab          motd             periodic/        services         udhcpd.conf
fstab            issue            mtab             profile          shadow
group            logrotate.d/     network/         profile.d/       shells
/ # cat  /etc/*release
3.11.3
NAME="Alpine Linux"
ID=alpine
VERSION_ID=3.11.3
PRETTY_NAME="Alpine Linux v3.11"
HOME_URL="https://alpinelinux.org/"
BUG_REPORT_URL="https://bugs.alpinelinux.org/"
```

# Multi-Stage Builds

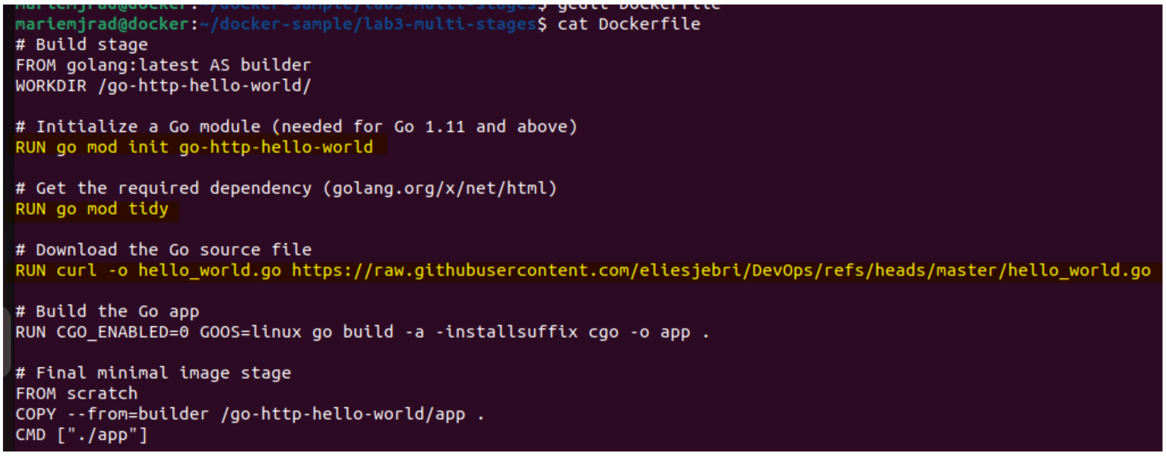
## 1. Understanding the Dockerfile

### Build Stage

- **Base Image:** Uses golang:latest as the builder.
- **Working Directory:** Sets to /go-http-hello-world.
- **Initialize Go Module:** Runs go mod init go-http-hello-world.
- **Download Dependencies:** Runs go mod tidy.
- **Fetch Source Code:** Uses curl to download hello_world.go from GitHub.
- **Compile Go Application:**
  - Uses CGO_ENABLED=0 for a statically compiled binary.
  - Targets **Linux OS** with **GOOS=linux.**
  - Produces an executable named app.

### Minimal Image Stage

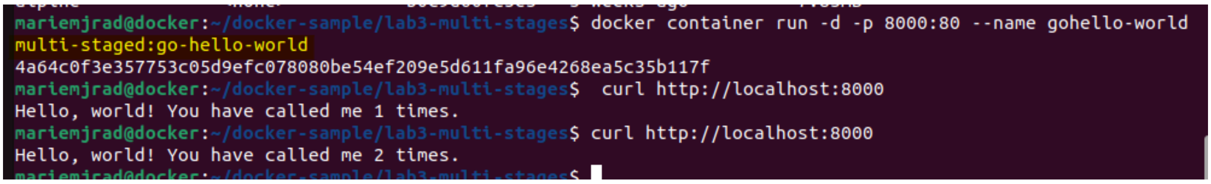
- Uses FROM scratch for a minimal image.
- Copies the compiled binary (app) from the previous stage.
- Defines the **entry command** as CMD ["./app"].



```
mariemjrad@docker:~/docker-sample/lab3-multi-stages$ gedit Dockerfile
mariemjrad@docker:~/docker-sample/lab3-multi-stages$ cat Dockerfile
# Build stage
FROM golang:latest AS builder
WORKDIR /go-http-hello-world/

# Initialize a Go module (needed for Go 1.11 and above)
RUN go mod init go-http-hello-world

# Get the required dependency (golang.org/x/net/html)
RUN go mod tidy

# Download the Go source file
RUN curl -o hello_world.go https://raw.githubusercontent.com/eliesjebri/DevOps/refs/heads/master/hello_world.go

# Build the Go app
RUN CGO_ENABLED=0 GOOS=linux go build -a -installsuffix cgo -o app .

# Final minimal image stage
FROM scratch
COPY --from=builder /go-http-hello-world/app .
CMD ["./app"]
```

## 2. Running the Container

- Runs the container in detached mode:
  docker container run -d -p 8000:80 --name gohello-world multi-staged:go-hello-world
- **Flags Explanation:**
  - -d: Runs in the background.
  - -p 8000:80: Maps port **8000 on the host** to **port 80 in the container**.
  - --name gohello-world: Assigns a container name.
  - multi-staged:go-hello-world: Specifies the built image.



```
mariemjrad@docker:~/docker-sample/lab3-multi-stages$ docker container run -d -p 8000:80 --name gohello-world
multi-staged:go-hello-world
4a64c0f3e357753c05d9efc078080be54ef209e5d611fa96e4268ea5c35b117f
mariemjrad@docker:~/docker-sample/lab3-multi-stages$  curl http://localhost:8000
Hello, world! You have called me 1 times.
mariemjrad@docker:~/docker-sample/lab3-multi-stages$ curl http://localhost:8000
Hello, world! You have called me 2 times.
mariemjrad@docker:~/docker-sample/lab3-multi-stages$
```