System Administration: Final Project

•••

By Christopher Penner and Michael Gomez

Introduction

What is Docker?

- A way to package software to run on any hardware

It has 3 main parts to it:

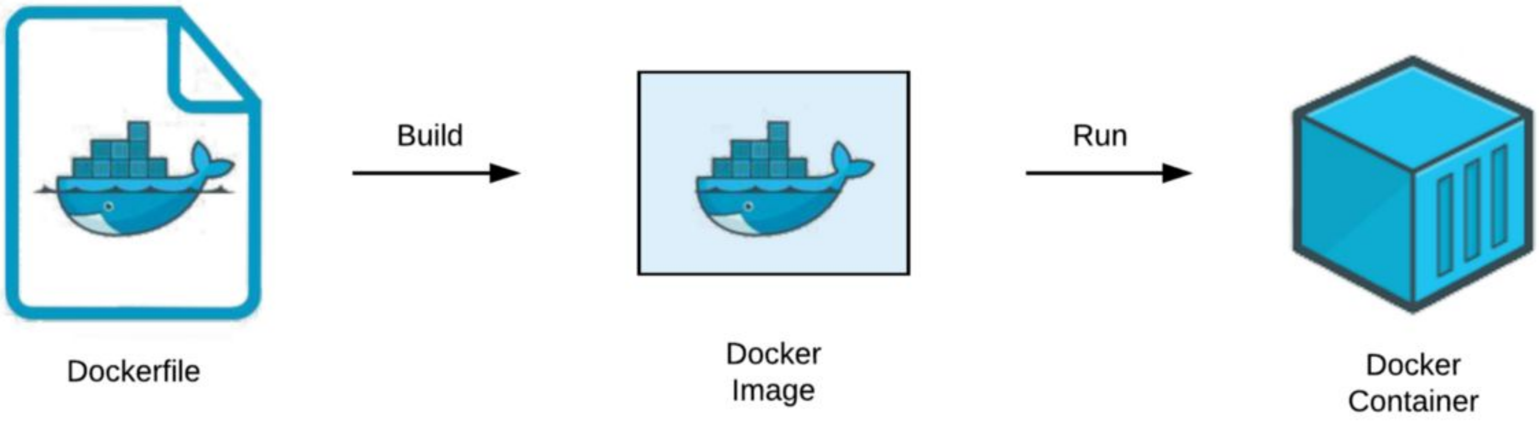
- Dockerfile
- Image
- Container

What is Docker?

Dockerfile - a blueprint for the Docker image

Image - template for running Docker containers

Container - template for running Docker containers

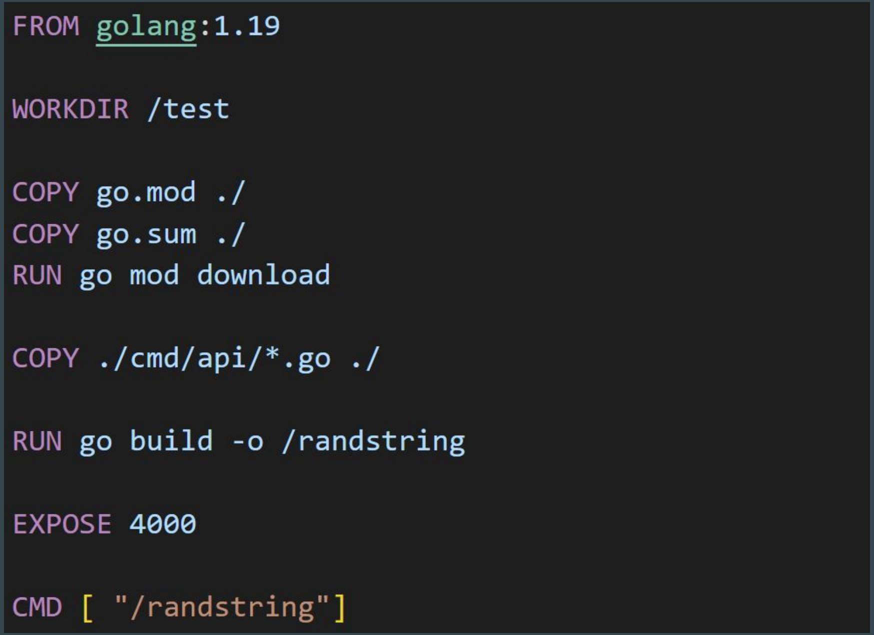


Dockerfile

Defines the Environment

It builds the docker image with all dependencies and the correct environment version.

Example:



```
FROM golang:1.19

WORKDIR /test

COPY go.mod ./
COPY go.sum ./
RUN go mod download

COPY ./cmd/api/*.go ./

RUN go build -o /randstring

EXPOSE 4000

CMD [ "/randstring"]
```

Docker Image

Build a Docker Image with:

docker build -t (name) (location)

-t (tag) puts a nametag on the image: (name)

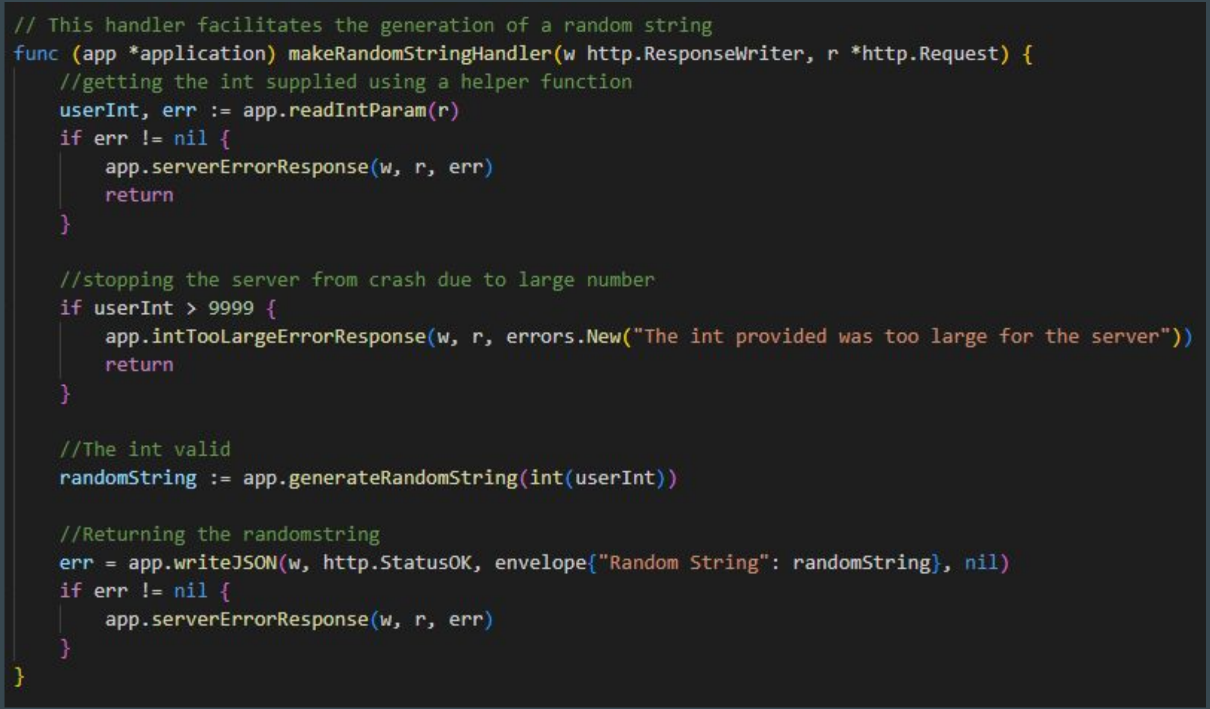
Followed by the path to the dockerfile: (location)

Web Application

Our web application is a simple backend web server that supports only one api call.

Where a random string is generated based on the integer we supply in the api call.

The web app using json to server the information back to the user.



```go
// This handler facilitates the generation of a random string
func (app *application) makeRandomStringHandler(w http.ResponseWriter, r *http.Request) {
    //getting the int supplied using a helper function
    userInt, err := app.readIntParam(r)
    if err != nil {
        app.serverErrorResponse(w, r, err)
        return
    }

    //stopping the server from crash due to large number
    if userInt > 9999 {
        app.intTooLargeErrorResponse(w, r, errors.New("The int provided was too large for the server"))
        return
    }

    //The int valid
    randomString := app.generateRandomString(int(userInt))

    //Returning the randomstring
    err = app.writeJSON(w, http.StatusOK, envelope{"Random String": randomString}, nil)
    if err != nil {
        app.serverErrorResponse(w, r, err)
    }
}
```

Container Demonstration

Thanks for watching