CSE - 322 Cloud Computing

Gaurav Malave

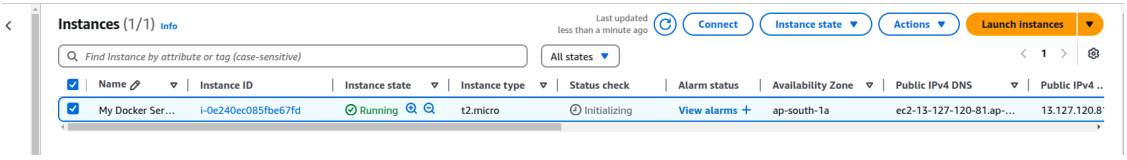
2022BCD0017

GoLang Service Experimentation and Deployment on AWS EC2

Initial Setup: EC2 Instance and GoLang Installation

Launch and Connect to EC2 Instance

1. **Launch EC2 Instance:**
   - Launched an EC2 instance using an Ubuntu AMI in the `ap-south-1` region.
   - Public IP: `13.127.120.81`.
   - Security group configured to allow:
     - SSH (port 22)
     - HTTP (port 8080)



2. **Connect to EC2 Instance:**
   - Attempted SSH connection using:

```
ssh -i "~/Downloads/Lab10.pem" ubuntu@ec2-13-127-120-81.ap-south-1.compute.amazonaws.com
```

   - Encountered error due to permissions on Lab10.pem being too open (0664).

3. **Fix Permissions Issue:**
   - Changed permissions to 600:

```
chmod 600 ~/Downloads/Lab10.pem
```

   - Retried SSH connection, which succeeded:

```
ubuntu@ip-172-31-42-132:~$
```

```
nightfury653@nightfury653-IdeaPad-Gaming-3-15ACH6:~/Downloads$ ssh -i "Lab10.pem" ubuntu@ec2-13-127-120-81.ap-south-1.compute.amazonaws.com
Welcome to Ubuntu 24.04.1 LTS (GNU/Linux 6.8.0-1021-aws x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/pro

 System information as of Tue Mar 18 07:06:31 UTC 2025

  System load:  0.08              Processes:             106
  Usage of /:   25.1% of 6.71GB   Users logged in:       1
  Memory usage: 20%               IPv4 address for enX0: 172.31.42.132
  Swap usage:   0%


Expanded Security Maintenance for Applications is not enabled.

0 updates can be applied immediately.

Enable ESM Apps to receive additional future security updates.
See https://ubuntu.com/esm or run: sudo pro status


The list of available updates is more than a week old.
To check for new updates run: sudo apt update

Last login: Tue Mar 18 07:02:04 2025 from 13.233.177.5
To run a command as administrator (user "root"), use "sudo <command>".
See "man sudo_root" for details.
```

Install GoLang

1. **Install GoLang on EC2:**
   - Updated the instance:

     ```
     sudo apt update && sudo apt upgrade -y
     ```

   - Downloaded and installed Go:

     ```
     wget https://golang.org/dl/go1.21.8.linux-amd64.tar.gz
     sudo tar -C /usr/local -xzf go1.21.8.linux-amd64.tar.gz
     ```

```
ubuntu@ip-172-31-42-132:~$ wget https://golang.org/dl/go1.21.8.linux-amd64.tar.gz
sudo tar -C /usr/local -xzf go1.21.8.linux-amd64.tar.gz
--2025-03-18 07:17:32--  https://golang.org/dl/go1.21.8.linux-amd64.tar.gz
Resolving golang.org (golang.org)... 142.250.199.145, 2404:6800:4009:803::2011
Connecting to golang.org (golang.org)|142.250.199.145|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://go.dev/dl/go1.21.8.linux-amd64.tar.gz [following]
--2025-03-18 07:17:33--  https://go.dev/dl/go1.21.8.linux-amd64.tar.gz
Resolving go.dev (go.dev)... 216.239.36.21, 216.239.38.21, 216.239.32.21, ...
Connecting to go.dev (go.dev)|216.239.36.21|:443... connected.
HTTP request sent, awaiting response... 302 Found
Location: https://dl.google.com/go/go1.21.8.linux-amd64.tar.gz [following]
--2025-03-18 07:17:33--  https://dl.google.com/go/go1.21.8.linux-amd64.tar.gz
Resolving dl.google.com (dl.google.com)... 142.250.77.46, 2404:6800:4009:82f::200e
Connecting to dl.google.com (dl.google.com)|142.250.77.46|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 66647874 (64M) [application/x-gzip]
Saving to: ‘go1.21.8.linux-amd64.tar.gz’

go1.21.8.linux-amd 100%[===============>]  63.56M  17.5MB/s    in 3.6s

2025-03-18 07:17:37 (17.4 MB/s) - ‘go1.21.8.linux-amd64.tar.gz’ saved [66647874/66647874]
```

- Set up environment variables:

```
echo "export PATH=$PATH:/usr/local/go/bin" >> ~/.bashrc
source ~/.bashrc
```

```
ubuntu@ip-172-31-42-132:~$ echo "export PATH=$PATH:/usr/local/go/bin" >> ~/.bashrc
source ~/.bashrc
```

- Verified installation:

```
go version
```

```
ubuntu@ip-172-31-42-132:~$ echo "export PATH=$PATH:/usr/local/go/bin" >> ~/.bashrc
source ~/.bashrc
ubuntu@ip-172-31-42-132:~$ go version
go version go1.21.8 linux/amd64
```

Task 1: Experiment with GoLang Services on AWS EC2 Server

Initial GoLang Service Setup

1. **Create a Simple Go Service:**

   - Created a directory and initial main.go file:

```
mkdir go-service && cd go-service
```

   - Initial main.go:

```
package main

import (
    "fmt"
    "net/http"
)

func handler(w http.ResponseWriter, r *http.Request) {
    fmt.Fprintf(w, "Hello from GoLang on EC2!")
}

func main() {
    http.HandleFunc("/", handler)
    fmt.Println("Server starting on port 8080...")
    http.ListenAndServe(":8080", nil)
}
```

2. **Build and Run (Initial Attempt):**
   - Attempted to build:

     ```
     go build -o app
     ```

   - Encountered error: `go: go.mod file not found in current directory or any parent directory.`

3. **Fix Go Module Issue:**
   - Initialized a Go module:

     ```
     go mod init go-service
     ```

   - Rebuilt the application:

     ```
     go build -o app
     ```

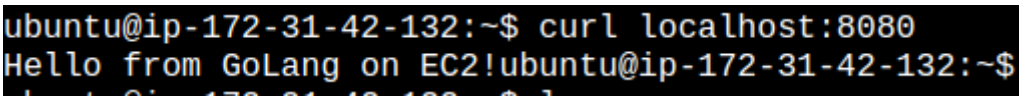
   - Ran the application:

     ```
     ./app
     ```

```
ubuntu@ip-172-31-42-132:~$ echo "export PATH=$PATH:/usr/local/go/bin" >> ~/.bashrc
source ~/.bashrc
ubuntu@ip-172-31-42-132:~$ go version
go version go1.21.8 linux/amd64
ubuntu@ip-172-31-42-132:~$ mkdir go-service && cd go-service
ubuntu@ip-172-31-42-132:~/go-service$ nano main.go
```

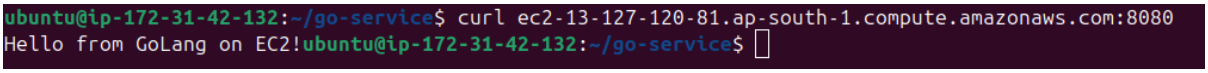
4. **Test the Service:**
   - Tested locally on EC2:

```
curl localhost:8080
```



   - Tested externally:

```
curl 13.127.120.81:8080
```



Task 2: Develop Dockerfile and Deploy the Docker Image

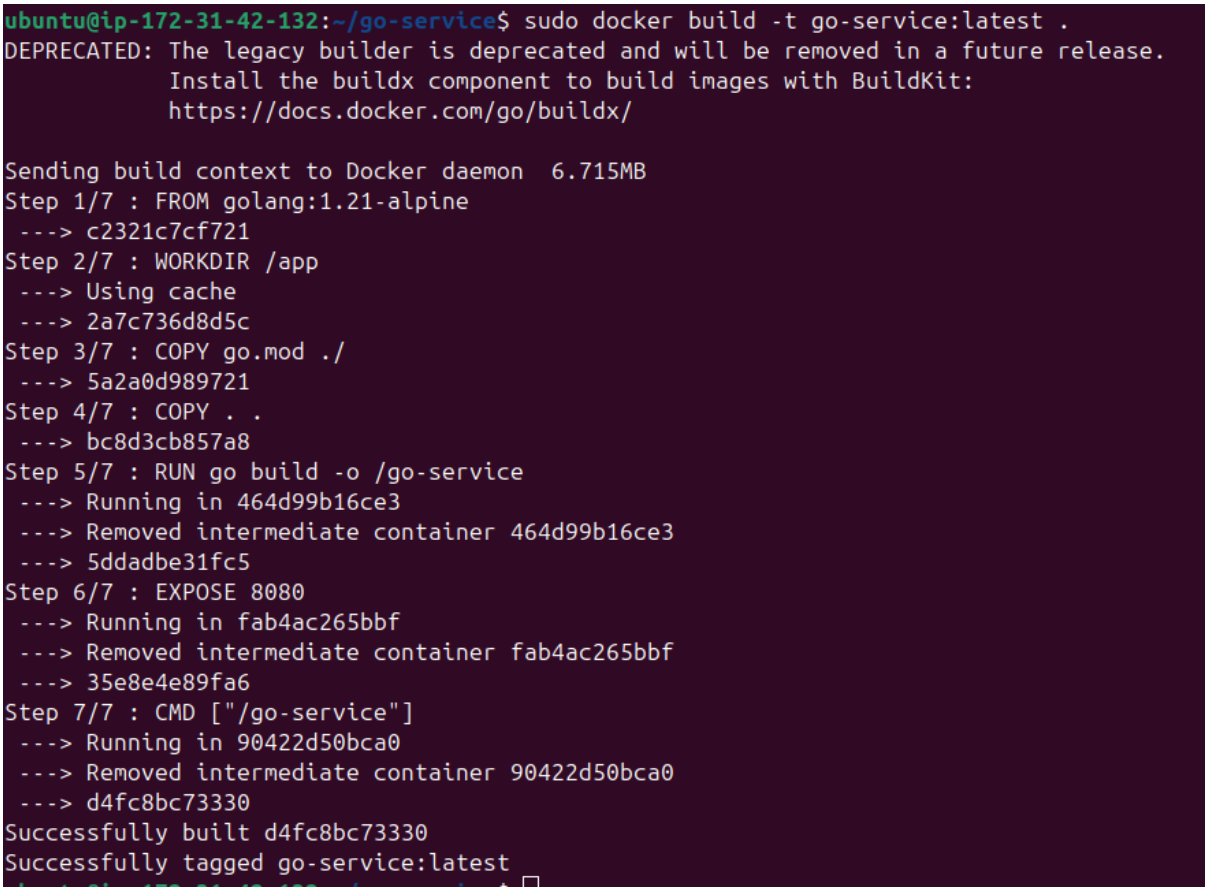
Create and Build the Dockerfile

1. **Initial Dockerfile Attempt:**
   - Created a Dockerfile in the go-service directory:

```
# Use official GoLang image as base
FROM golang:1.21-alpine

# Set working directory
WORKDIR /app

# Copy go.mod and go.sum (if you have them, otherwise skip)
COPY go.mod ./

# Copy the source code
COPY . .

# Build the Go app
RUN go build -o /go-service

# Expose port 8080
EXPOSE 8080

# Run the app
CMD ["/go-service"]
```

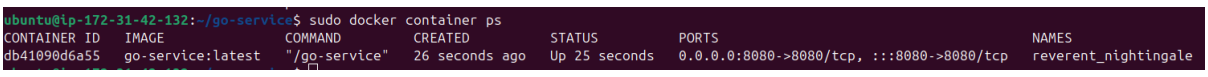
2. **Build the Docker Image:**
   - Build the image:

```
sudo docker build -t go-service:latest .
```



   - Checked the image:

```
sudo docker images
```



Option 1: Deploy Docker Image to EC2

1. **Ensure Docker is Running on EC2:**
   - Verified Docker status:

```
sudo systemctl status docker
```

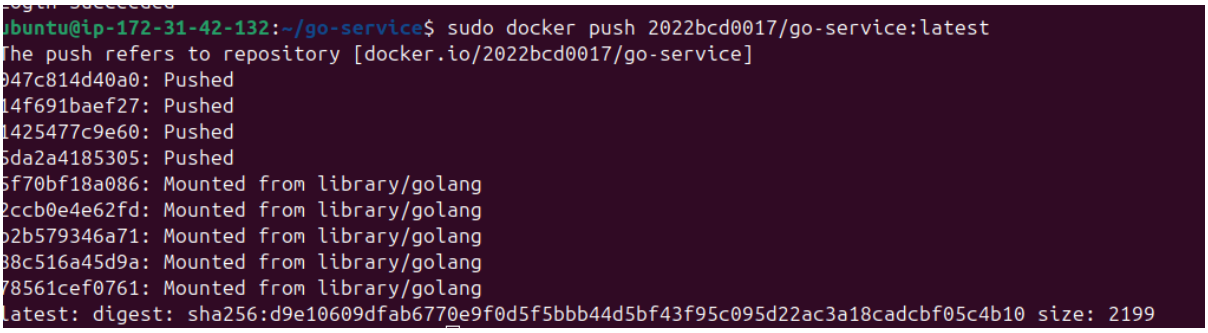
   - Started Docker if necessary:

```
sudo systemctl start docker
```

2. **Run the Docker Container:**

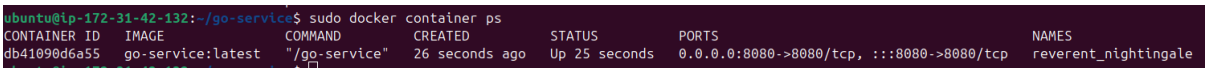
   - Deployed the image:

```
sudo docker run -d -p 8080:8080 go-service:latest
```



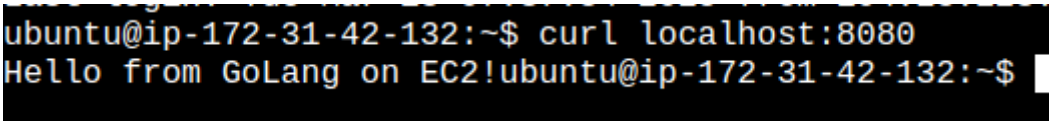
   - Checked running containers:

```
sudo docker ps
```

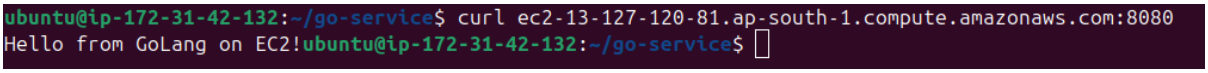


3. **Test the Deployed Service:**

   - Tested locally: `curl localhost:8080`



   - Tested externally: `curl 13.127.120.81:8080`



4. **Deployment to the Docker HUB:**

   - Tagging the image

```
sudo docker tag go-service:latest 2022bcd0017/go-service:latest
```

```
ubuntu@ip-172-31-42-132:~/go-service$ sudo docker tag go-service:latest 2022bcd0017/go-service:latest
ubuntu@ip-172-31-42-132:~/go-service$ sudo docker login
```

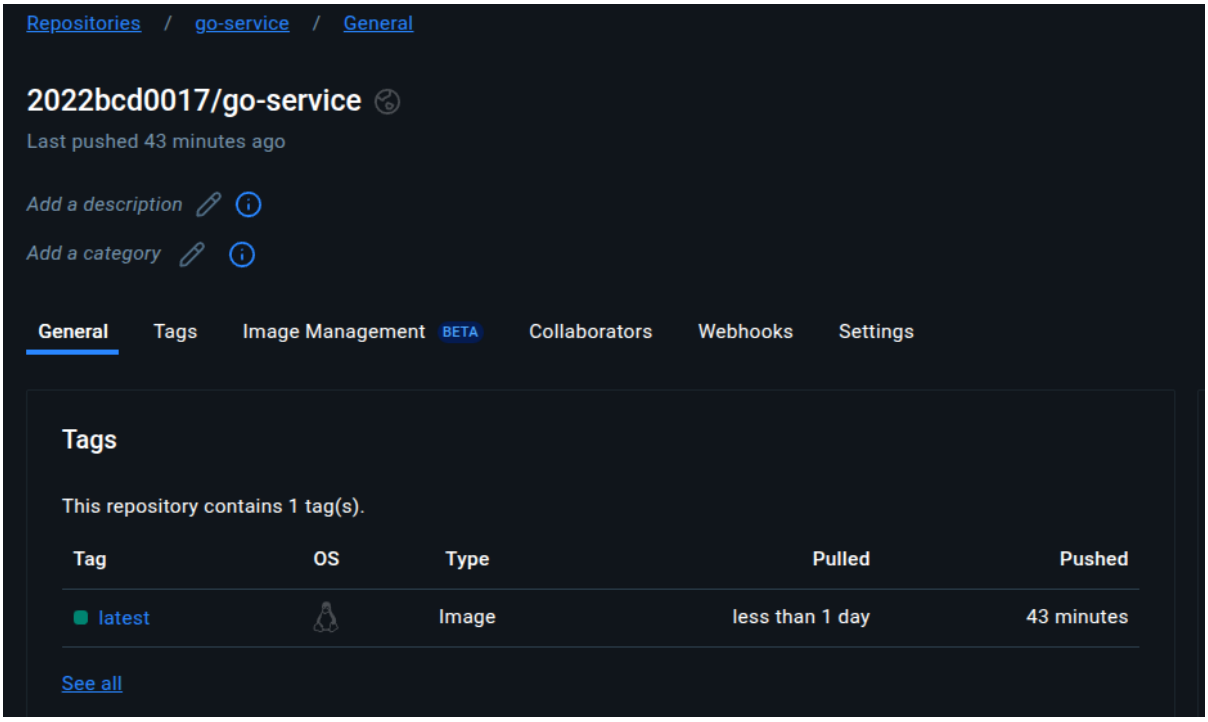
```
ubuntu@ip-172-31-42-132:~/go-service$ sudo docker login
Log in with your Docker ID or email address to push and pull images from Docker Hu
You can log in with your password or a Personal Access Token (PAT). Using a limite
ocker.com/go/access-tokens/

Username: 2022bcd0017
Password:
WARNING! Your password will be stored unencrypted in /root/.docker/config.json.
Configure a credential helper to remove this warning. See
https://docs.docker.com/engine/reference/commandline/login/#credentials-store

Login Succeeded
```

- Pushing the image

```
sudo docker push 2022bcd0017/go-service:latest
```

```
ubuntu@ip-172-31-42-132:~/go-service$ sudo docker push 2022bcd0017/go-service:latest
The push refers to repository [docker.io/2022bcd0017/go-service]
947c814d40a0: Pushed
14f691baef27: Pushed
1425477c9e60: Pushed
5da2a4185305: Pushed
5f70bf18a086: Mounted from library/golang
2ccb0e4e62fd: Mounted from library/golang
02b579346a71: Mounted from library/golang
38c516a45d9a: Mounted from library/golang
78561cef0761: Mounted from library/golang
latest: digest: sha256:d9e10609dfab6770e9f0d5f5bbb44d5bf43f95c095d22ac3a18cadcbf05c4b10 size: 2199
```

- DockerHUB



Conclusion

This Lab involved setting up a GoLang service on AWS EC2, and deploying it using Docker. Key takeaways:

- Go modules (go.mod) are essential for modern Go projects.
- Docker simplifies deployment but requires careful handling of files like go.sum.
- ECS provides a robust solution for container orchestration, though it requires more configuration than direct EC2 deployment.