Zerodha Ops Task

Task1
Description

This is a sample `Go` application which connects to Redis. The app increments a Redis `counter` on an incoming request.

1. Use `make build` to compile the binary.

```
dikshant@ubuntu-22:~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master$ make build
go build -o demo.bin -ldflags="-X 'main.version='"
dikshant@ubuntu-22:~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master$ ls
demo.bin  go.mod  go.sum  main.go  Makefile  README.md
dikshant@ubuntu-22:~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master$
```

2. Set the environment variables:
 - `DEMO_APP_ADDR`: Address where the app should listen to
 - `DEMO_REDIS_ADDR`: Address where Redis is running

```
dikshant@ubuntu-22:~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master$ export DEMO_APP_ADDR=localhost:3000
dikshant@ubuntu-22:~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master$ export DEMO_REDIS_ADDR=localhost:6379
dikshant@ubuntu-22:~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master$
```

Testing the website on port 3000

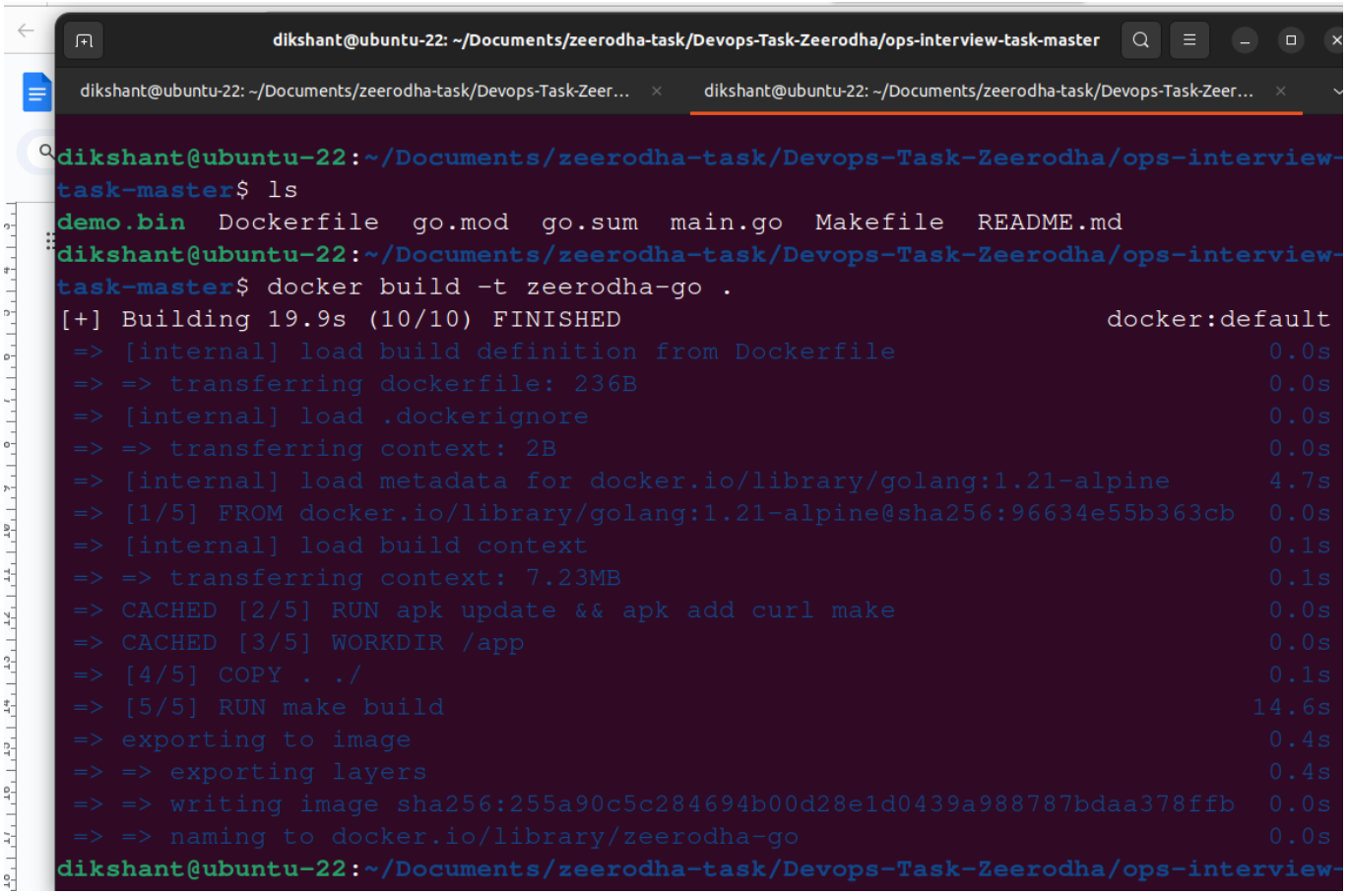
```
dikshant@ubuntu-22:~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master$ ./demo.bin
2023/09/23 15:19:55 Booting app on localhost:3000
```

Tasks

3 Create a `Dockerfile` for the app.

Dockerfile

```
FROM golang:1.21-alpine
RUN apk update && apk add curl make
ENV DEMO_APP_ADDR=0.0.0.0:3000
ENV DEMO_REDIS_ADDR=10.0.2.15:6379
WORKDIR /app
COPY . ./
RUN make build
EXPOSE 8080
CMD [ "./demo.bin" ]
```



```
dikshant@ubuntu-22: ~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master
dikshant@ubuntu-22: ~/Documents/zeerodha-task/Devops-Task-Zeer...   dikshant@ubuntu-22: ~/Documents/zeerodha-task/Devops-Task-Zeer...
dikshant@ubuntu-22:~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master$ docker run -dit -p 3000:3000 zeerodha-go
18fa9a12a7d11eff464603d025812657a7a33f96b841f85f391c364cd53d1016

dikshant@ubuntu-22:~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master$
dikshant@ubuntu-22:~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master$ docker ps
CONTAINER ID   IMAGE          COMMAND         CREATED          STATUS          PORTS
                                                   NAMES
18fa9a12a7d1   zeerodha-go    "./demo.bin"    3 seconds ago    Up 2 seconds    0.0.0
.0:3000->3000/tcp, :::3000->3000/tcp, 8080/tcp   lucid_easley
dikshant@ubuntu-22:~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master$ curl localhost:3000
welcome to api. key count is: 18dikshant@ubuntu-22:~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master$ curl localhost:3000
welcome to api. key count is: 19dikshant@ubuntu-22:~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master$
```

Done!!

Task2

Create a `docker-compose.yml` for the app which includes the following:
- `redis` service, with data directory mounted.
- `app` service, ensuring that it has a dependency on the Redis service starting correctly.
- `nginx` service acting as a reverse proxy for the app. Bonus: Implement SSL using self-signed certificates.

Let's generate a self signed ssl certificate for our app for domain **dikshant.zeerodha.com.**

1. Generate pvt key for making self signed certificate.
$ openssl genpkey -algorithm RSA -out dikshant.zeerodha.com

```
dikshant@ubuntu-22: ~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master
dikshant@ubuntu-22:~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master$ ls
demo.bin            Dockerfile  go.sum   Makefile    README.md
docker-compose.yml  go.mod      main.go  nginx.conf  ssl
dikshant@ubuntu-22:~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master$ #openssl genpkey -algorithm RSA -out example.com.key
dikshant@ubuntu-22:~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master$ openssl genpkey -algorithm RSA -out dikshant.zeerodha.com
..+.+..+....+...+...............+...+........+..........+.....+.+...+.....+...++
++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*.....+.+++++++++
+++++++++++++++++++++++++++++++++++++++++++++++++++++++++++*..+.+..............+...
.........+................+........+....+..++++++++++++++++++++++++++++++++++++++
++++++++++++++++++++++++++
..........+......+.........+..+....+........+...+..+...++++++++++++++++++++++++
+++++++++++++++++++++++++++++++++++++++++*...........+..+.......+...+...+......+
...+..+....+...............+.....++++++++++++++++++++++++++++++++++++++++++++++
+++++++++++++++++*..........+.....+.......+..+...+.......+.....+++++++++++++++
++++++++++++++++++++++++++++++++++++++++++++++++
dikshant@ubuntu-22:~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master$ ls
demo.bin               docker-compose.yml  go.mod  main.go   nginx.conf  ssl
dikshant.zeerodha.com  Dockerfile          go.sum  Makefile  README.md
dikshant@ubuntu-22:~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master$
```

2. Generate a Self-Signed Certificate:

$ openssl req -new -x509 -key dikshant.zeerodha.com -out dikshant.zeerodha.com.crt -days 365

Use the private key to create a self-signed SSL certificate

```
dikshant@ubuntu-22: ~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master
dikshant@ubuntu-22:~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master$ openssl req -new -x509 -key dikshant.zeerodha.com -out dikshant.zeerodha.com.crt -days 365
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [AU]:IN
State or Province Name (full name) [Some-State]:Delhi
Locality Name (eg, city) []:Gurugram
Organization Name (eg, company) [Internet Widgits Pty Ltd]:
Organizational Unit Name (eg, section) []:
Common Name (e.g. server FQDN or YOUR name) []:dikshant.zeerodha.com
Email Address []:dikshantmali.dev@gmail.com
dikshant@ubuntu-22:~/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-master$
```

Updated docker file for docker compose.

```
FROM golang:1.21-alpine
RUN apk update && apk add curl make
ENV DEMO_APP_ADDR=0.0.0.0:3000
ENV DEMO_REDIS_ADDR=redis:6379
WORKDIR /app
COPY . ./
RUN make build
EXPOSE 8080
CMD [ "./demo.bin" ]
```

Nginx Configuration

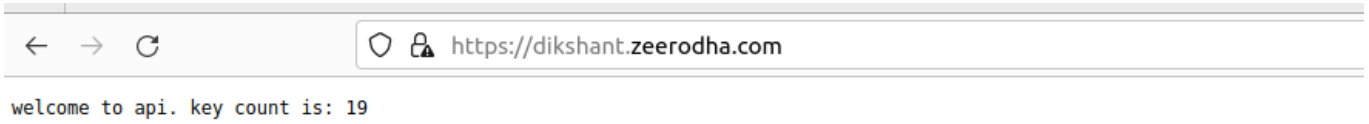
```
events {
}
http {
sendfile on;

        upstream appp {
        server app:3000;
        }
        server {
        listen 80;
   listen 443 ssl;
        server_name dikshant.zeerodha.com; # Replace with your domain name

        ssl_certificate /etc/nginx/ssl/dikshant.zeerodha.com.crt;
        ssl_certificate_key /etc/nginx/ssl/dikshant.zeerodha.com;

        # Additional SSL configurations
        ssl_protocols TLSv1.2 TLSv1.3;
        ssl_prefer_server_ciphers off;
        ssl_ciphers 'EECDH+AESGCM:EDH+AESGCM:AES256+EECDH:AES256+EDH';

        # Other Nginx configuration settings
        # ...

        location / {
        proxy_pass http://appp;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;

        # Your proxy or web server configuration here
        }
        }
}
```

Docker-compose file

```
version: '3'
services:
```

```
  redis:
        image: redis:latest
        volumes:
        - ./redis-data:/data
        ports:
        - "6379:6379"
        networks:
        - nginx-network
  app:
        image: zeerodha-go:latest
        build: .
        depends_on:
        - redis
        environment:
        - REDIS_HOST=redis
        ports:
        - "3000:3000"
        networks:
        - nginx-network

  nginx:
        image: nginx:latest
        ports:
        - "80:80"
        - "443:443"
        volumes:
        - ./nginx.conf:/etc/nginx/nginx.conf
        - ./ssl/:/etc/nginx/ssl/ # For SSL certificates
        depends_on:
        - app
        networks:
        - nginx-network

networks:
  nginx-network:
```

Accessing file on https with self signed ssl



Certificate details

Certificate

dikshant.zeerodha.com

Subject Name	
Country	IN
State/Province	Delhi
Locality	Gurugram
Organization	Internet Widgits Pty Ltd
Common Name	dikshant.zeerodha.com
Email Address	dikshantmali.dev@gmail.com
Issuer Name	
Country	IN
State/Province	Delhi
Locality	Gurugram
Organization	Internet Widgits Pty Ltd
Common Name	dikshant.zeerodha.com
Email Address	dikshantmali.dev@gmail.com
Validity	
Not Before	Sat, 23 Sep 2023 10:39:04 GMT
Not After	Sun, 22 Sep 2024 10:39:04 GMT
Public Key Info	

Done!!

Task3

Write a bash script to set up a [Vagrant box](https://vagrant.io) with Ubuntu. Ensure the script has error checks and is idempotent.

```
# Define the number of master and worker nodes
# If this number is changed, remember to update setup-hosts.sh script with the new hosts IP details in /etc/hosts of each VM.
NUM_MASTER_NODE = 1
NUM_WORKER_NODE = 1
IP_NW = "192.168.56."
MASTER_IP_START = 1
NODE_IP_START = 2
Vagrant.configure("2") do |config|
  config.vm.box = "tknerr/baseimage-ubuntu-20.04"
  config.vm.box_check_update = false
  # Provision Master Nodes
  (1..NUM_MASTER_NODE).each do |i|
      config.vm.define "kubemaster" do |node|
        # Name shown in the GUI
        node.vm.provider "virtualbox" do |vb|
            vb.name = "kubemaster"
```

```
            #vb.memory = 2048
            #vb.cpus = 2
        end
        node.vm.hostname = "kubemaster"
        node.vm.network :private_network, ip: IP_NW + "#{MASTER_IP_START + i}"
        node.vm.network "forwarded_port", guest: 22, host: "#{2710 + i}"
      end
  end
  # Provision Worker Nodes
  (1..NUM_WORKER_NODE).each do |i|
    config.vm.define "kubenode0#{i}" do |node|
        node.vm.provider "virtualbox" do |vb|
            vb.name = "kubenode0#{i}"
            #vb.memory = 2048
            #vb.cpus = 2
        end
        node.vm.hostname = "kubenode0#{i}"
        node.vm.network :private_network, ip: IP_NW + "#{NODE_IP_START + i}"
                node.vm.network "forwarded_port", guest: 22, host: "#{2720 +
i}"
    end
  end
 end
```

The above script I have used to provision 3 ubuntu machines on virtualbox with vagrant to setup my own k8s cluster.

Task4:

- Using Ansible provision the VM to:
  - Setup hostname of VM as `demo-ops`.
  - Create a user `demo`.
  - Harden the security:
    - Disable root login.
    - Setup a basic firewall (e.g., UFW) allowing only specific ports.
  - Configure `sysctl` for sane defaults. (For eg: increasing open files limit)
  - Configure sysctl for sane defaults. For each sysctl parameter changed:
    - Document the change.
    - Provide a brief justification or explanation (2-3 lines) detailing why this specific change was made and its implications.
  - Set the system's timezone to "Asia/Kolkata".
  - Install Docker and Docker-Compose.
  - Configure Docker Daemon to have sane defaults. For eg: keep logs size in check.
  - Deploy the `docker-compose.yml` in `/etc/demo-ops` and start the services.

Below is the ansible playbook configuration which will do the above task on aws instance.

NOTE: for sysctl I have configured below 2 parameters

1. **fs.file-max** sets an upper limit on the total number of open files system-wide. When this limit is reached, processes may be unable to open additional files until existing ones are closed.

2. Kernel.pid_max this parameter defines the maximum ID that can be assigned to a process.

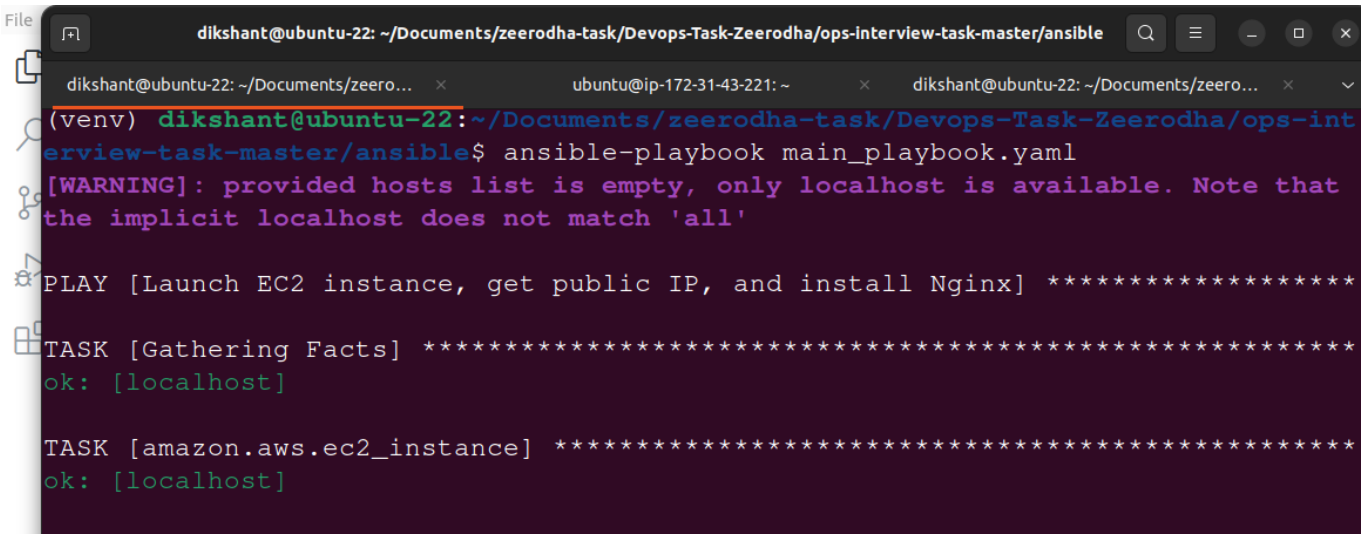
The above two parameters I have set through ansible and are working fine.

```
Ansible playbook
---
- name: Launch EC2 instance, get public IP, and install Nginx
 hosts: localhost
 tasks:
   - amazon.aws.ec2_instance:
       name: "public-compute-instance"
       access_key: Accesskey
       secret_key: secretkey
       key_name: "ansible"
       vpc_subnet_id: subnet-00009cc64bb842e77
       instance_type: t3.micro
       security_group: default
       network:
         assign_public_ip: true
       image_id: ami-053b0d53c279acc90
       tags:
         Environment: Testing
     register: ec2
   - name: Disable Root Login
     become: yes
     ansible.builtin.lineinfile:
       path: /etc/ssh/sshd_config
       regexp: '^PermitRootLogin'
       line: 'PermitRootLogin no'

   - name: Create SSH Group to login dynamically to EC2 Instance
     add_host:
       hostname: "34.235.131.96"
       ansible_ssh_private_key_file:
/home/dikshant/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-m
aster/ansible/ansible.pem
       groupname: ec2_server
     with_items: ec2.instances

   - name: Wait for SSH to come up
     wait_for:
```

```
      host: "34.235.131.96"
      port: 22
      state: started
    with_items: ec2.instances

  - name: Install Nginx
    become: yes
    ansible.builtin.shell: "sudo apt-get update && sudo apt-get install -y
nginx"
    delegate_to: "34.235.131.96"
    remote_user: "ubuntu"

  - name: Allow SSH and enable UFW
    become: yes
    ansible.builtin.shell: |
      sudo ufw allow OpenSSH
      sudo ufw --force enable
    delegate_to: "34.235.131.96"
    remote_user: "ubuntu"

  - name: Increase Open Files Limit
    become: yes
    ansible.builtin.sysctl:
      name: fs.file-max
      value: 65536
    delegate_to: "34.235.131.96"
    remote_user: "ubuntu"

  - name: Change Kernel PID Max
    become: yes
    ansible.builtin.sysctl:
      name: kernel.pid_max
      value: 65535
    delegate_to: "34.235.131.96"
    remote_user: "ubuntu"


  - name: Set Timezone to Asia/Kolkata
    ansible.builtin.shell: |
      sudo timedatectl set-timezone Asia/Kolkata
    delegate_to: "34.235.131.96"
    remote_user: "ubuntu"

  - name: Install docker and docker compose
    ansible.builtin.shell: |
      sudo apt-get update -y
```

```
      sudo apt-get install docker-ce docker-ce-cli containerd.io
docker-buildx-plugin docker-compose-plugin -y
      sudo chmod 777 /var/run/docker.sock
    delegate_to: "34.235.131.96"
    remote_user: "ubuntu"

  - name: Copy All data to ubuntu
    ansible.builtin.copy:
      src:
/home/dikshant/Documents/zeerodha-task/Devops-Task-Zeerodha/ops-interview-task-m
aster/
      dest: /home/ubuntu
      owner: ubuntu
      group: ubuntu
      mode: u+rw,g-wx,o-rwx
    delegate_to: "34.235.131.96"
    remote_user: "ubuntu"

  - name: Deploy Dockercompsoe
    ansible.builtin.shell: |
      cd /home/ubuntu
      docker compose up -d
    delegate_to: "34.235.131.96"
    remote_user: "ubuntu"
```

Playbook output



```
TASK [Create SSH Group to login dynamically to EC2 Instance] *******************
changed: [localhost] => (item=ec2.instances)

TASK [Wait for SSH to come up] *************************************************
ok: [localhost] => (item=ec2.instances)

TASK [Install Nginx] ***********************************************************
changed: [localhost -> 34.235.131.96]

TASK [Allow SSH and enable UFW] ************************************************
changed: [localhost -> 34.235.131.96]

TASK [Increase Open Files Limit] ***********************************************
ok: [localhost -> 34.235.131.96]

TASK [Change Kernel PID Max] ***************************************************
ok: [localhost -> 34.235.131.96]

TASK [Set Timezone to Asia/Kolkata] ********************************************
changed: [localhost -> 34.235.131.96]

TASK [Install docker and docker compose] ***************************************
```

Output

Done!!

The assignment is completed from my side