

## گزارش پروژه نهایی رایانش ابری

محمد متقی 9731057

رهام زنده دل نوبری 9731088

قسمت اول -

طبق دستورالعمل برنامه ای طراحی کردیم که از کاربر یادداشت می گیرد.

قسمت دوم -

DockerFile به شکل زیر است.

```
Dockerfile > ...
1  # Stage 1: Builder/Compiler
2  FROM python:3.8-alpine AS builder
3
4  WORKDIR /app
5  COPY requirements.txt ./
6  RUN pip install --target=/app/dependencies -r requirements.txt
7
8  COPY ./templates ./templates
9  COPY server.py ./
10 COPY config.json ./
11
12 # Stage 2: Runtime
13 FROM python:3.8-alpine
14 WORKDIR /app
15
16 #RUN apk update && apk add libusb-dev
17
18 COPY --from=builder /app .
19 ENV PYTHONPATH="${PYTHONPATH}:/app/dependencies"
20
21 CMD ["python", "./server.py"]
22
```

آن را با دستور `docker build -t note_reader:1.1` درست کردیم.

```
rohamzn@ubuntu:~$ docker images
REPOSITORY          TAG                 IMAGE ID            CREATED             SIZE
note_reader          1.1                b5a20824131d       14 minutes ago     56.5MB
rohamzn/note_reader  1.1                b5a20824131d       14 minutes ago     56.5MB
rohamzn/note_reader  1.0                6c8793384ee4       39 minutes ago     55MB
note_reader          1.0                6c8793384ee4       39 minutes ago     55MB
hw2_part2_test       latest             57a64e271ef1       2 months ago       60.9MB
rohamzn/hw2_part2_test 1                  57a64e271ef1       2 months ago       60.9MB
ardavan777/kapp       latest             192a003b1a1f       2 months ago       60.9MB
hw2_part2            latest             9bd2817d2463       2 months ago       60.9MB
```


می توانیم آن را در image های داکر ببینیم.

دستور بعدی `docker tag note_reader:1.1 rohamzn/note_reader:1.1` است.


در نهایت نیز آن را با دستور `docker push rohamzn/note_reader:1.1` به داکرهاب ارسال می کنیم.

rohamzn / **note\_reader**  
Last pushed: 14 minutes ago


Not Scanned
0
0
Public


**rohamzn / note\_reader**




**Description**  
*This repository does not have a description*


Last pushed: 15 minutes ago

## Tags and Scans

 VULNERABILITY SCANNING - DISABLED  
[Enable](#)

This repository contains 2 tag(s).

TAG	OS	PULLED	PUSHED
 1.1		---	15 minutes ago
 1.0		---	39 minutes ago

[See all](#)

قسمت سوم:

منابع را apply می کنیم:

```
rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl apply -f config-map.yaml
configmap/server-config created

rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl apply -f secret.yaml
secret/mongo-creds created

rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl apply -f PersistentVolume.yaml
persistentvolume/mongodb created

rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl apply -f PersistentVolumeClaim.yaml
persistentvolumeclaim/mongodb-pvc created

rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl apply -f mongo-deployment.yaml
error: the path "mongo-deployment.yaml" does not exist

rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl apply -f mongo_deployment.yaml
deployment.apps/mongo created

rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl apply -f mongo_service.yaml
service/mongo-svc created

rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl apply -f app_deployment.yaml
deployment.apps/app-server created

rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl apply -f app_service.yaml
service/app-svc created
```

برای مطمئن شدن از اجرا شدن سرویس ها، آنها را دونه به دونه get می کنیم:

```
rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl get configmap/server-config
NAME      DATA      AGE
server-config  1          76m

rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl get secret/mongo-creds
NAME      TYPE      DATA      AGE
mongo-creds  mongo-auth  2          75m

rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl get persistentvolume/mongodb
NAME      CAPACITY  ACCESS MODES  RECLAIM POLICY  STATUS  CLAIM                STORAGECLASS  REASON  AGE
mongodb  100Mi     RWO           Retain          Bound   default/mongodb-pvc  manual              74m

rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl get persistentvolumeclaim/mongodb-pvc
NAME      STATUS  VOLUME  CAPACITY  ACCESS MODES  STORAGECLASS  AGE
mongodb-pvc  Bound   mongodb  100Mi     RWO           manual         74m

rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl get deployment.apps/mongo
NAME      READY  UP-TO-DATE  AVAILABLE  AGE
mongo     1/1    1            1          90s

rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl get service/mongo-svc
NAME      TYPE      CLUSTER-IP  EXTERNAL-IP  PORT(S)      AGE
mongo-svc  ClusterIP  10.109.112.62  <none>       27017/TCP    74m

rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl get deployment.apps/app-server
NAME      READY  UP-TO-DATE  AVAILABLE  AGE
app-server  2/2    2            2          95s

rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl get service/app-svc
NAME      TYPE      CLUSTER-IP  EXTERNAL-IP  PORT(S)      AGE
app-svc   ClusterIP  10.98.46.186  <none>       8000/TCP     74m
```

همانطور که مشاهده می شود تمام پاد ها به درستی اجرا شده اند.

```
rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl get pods
NAME                                READY  STATUS    RESTARTS  AGE
app-server-d9458cc77-8hrtg          1/1    Running   0          2m47s
app-server-d9458cc77-xvw4n          1/1    Running   0          2m47s
mongo-8447dc947f-jttn5              1/1    Running   0          3m11s
```

```

rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl get ep
NAME      ENDPOINTS          AGE
app-service <none>             73d
app-svc    172.17.0.4:8000,172.17.0.5:8000 86m
kubernetes 192.168.49.2:8443    73d
mongo-svc  172.17.0.3:27017     86m
rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl get svc
NAME      TYPE      CLUSTER-IP      EXTERNAL-IP      PORT(S)          AGE
app-service ClusterIP  10.105.83.55     <none>           8080/TCP         73d
app-svc    ClusterIP  10.98.46.186     <none>           8000/TCP         87m
kubernetes ClusterIP  10.96.0.1        <none>           443/TCP          73d
mongo-svc  ClusterIP  10.109.112.62    <none>           27017/TCP        87m
rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl get pods -o wide
NAME                                READY   STATUS    RESTARTS   AGE   IP              NODE      NOMINATED NODE   READINESS GATES
app-server-d9458cc77-8hrtg         1/1    Running   0          15m   172.17.0.4      minikube  <none>           <none>
app-server-d9458cc77-xvw4n         1/1    Running   0          15m   172.17.0.5      minikube  <none>           <none>
mongo-8447dc947f-jttn5             1/1    Running   0          15m   172.17.0.3      minikube  <none>           <none>

```

اندپوینت ها و سرویس ها نیز به درستی تخصیص داده شده اند.

```

rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl logs app-server-d9458cc77-8hrtg
{'port': 8000, 'minutes_to_expire': 5, 'hostname': 'mongo-svc'}
* Serving Flask app 'server' (lazy loading)
* Environment: production
  WARNING: This is a development server. Do not use it in a production deployment.
  Use a production WSGI server instead.
* Debug mode: on
* Running on all addresses (0.0.0.0)
  WARNING: This is a development server. Do not use it in a production deployment.
* Running on http://127.0.0.1:5000
* Running on http://172.17.0.4:5000 (Press CTRL+C to quit)
* Restarting with stat
* Debugger is active!
* Debugger PIN: 118-995-567

```

لاگ یکی از پاد های app را می گیریم و می بینیم که به درستی اجرا شده است.

برای database deployment از یک پاد استفاده شده است زیرا volume پاد ها مشترک است و اگر بیشتر از یکی باشد ممکن است miss input داشته باشیم و در نتیجه lossy data داشته باشیم.

قسمت امتیازی:

```

rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl autoscale deployment app-server --cpu-percent=50 --min=3 --max=5
horizontalpodautoscaler.autoscaling/app-server autoscaled
rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl get hpa
NAME      REFERENCE          TARGETS      MINPODS   MAXPODS   REPLICAS   AGE
app-server Deployment/app-server <unknown>/50% 3          5          0          11s
rohamzn@ubuntu:~/Cloud_Final_Project$ kubectl get hpa
NAME      REFERENCE          TARGETS      MINPODS   MAXPODS   REPLICAS   AGE
app-server Deployment/app-server <unknown>/50% 3          5          2          21s

```

HPA: پارامتر های موجود جهت مقیاس کردن خودکار به صورت کلی به دو دسته per-pod resource و per-pod custom تقسیم می شوند، پارامترهای cpu utilization,

memory utilization, ... از جمله دسته per-pod resource و پارامترهای per-pod custom نیز همانند per-pod resource هستند ولی اعداد جای درصد را می گیرد.

چون در پروژه از CPU بیشتر استفاده شده است (جهت generate کردن URL) پس ما از cpu-percent استفاده کردیم.

### :Stateful set

موقعی که قرار است یک اپلیکیشن stateless توسعه دهیم یا موقعی که چندین پاد از یک volume مشترک استفاده کنند، توصیه می شود که از deployment استفاده کنیم. اما در مواقعی که هر پاد state مستقل خودش را دارد و از volume مخصوص خودش استفاده کند، بهتر است از stateful set استفاده کنیم.

در فایل statefulset.yaml توصیف خود را آورده ایم.

### :Helm chart

ساختار آن شامل 5 قسمت می شود:

helmignore: این فایل همانند gitignore است و می توانیم فایل هایی که می خواهیم ignore شوند را در آن مشخص کنیم.

Chart.yaml: در این فایل تمام اطلاعاتی که قرار است بسته بندی شود، قرار می گیرد.

Values.yaml: در این فایل تمام داده هایی که قرار است در templates قرار بگیرد، تعریف می شود.

Charts: در اینجا تمام چارت های وابسته به چارت main ذخیره می شوند.

Templates: در این پوشه، تمام محتویات و داده هایی که برای توسعه چارت نیاز است قرار می گیرد.

## :Docker Compose

شامل دو سرویس app و دیتابیس mongo می باشد. داده های آن به شکل زیر است.

```
version: "3.3"
services:
  app:
    image: 'rohamzn/note_reader:1.2'
    container_name: app
    depends_on:
      - mongo
    links:
      - mongo
    networks:
      - mongonetwork
    ports:
      - "8080:8080"
  mongo:
    image: mongo:latest
    hostname: mongodb_test
    container_name: mongo
    environment:
      MONGO_INITDB_DATABASE: data
      MONGO_INITDB_ROOT_USERNAME: rohamzn
      MONGO_INITDB_ROOT_PASSWORD: "75321475"

    networks:
      - mongonetwork
    ports:
      - 27017:27017
    volumes:
      - /opt/data/post/:/data/db
networks:
  mongonetwork:
    driver: bridge
```