گزارش تمرین دوم درس رایانش ابری (داکر و مقدمات کوبرنتیز)

محمدجواد زندیه (۹۸۳۱۰۳۲)

گام اول:

۱. ساخت Dockerfile:

```
Dockerfile
part1 > Dockerfile > ...
1  FROM alpine:3.17
2
3  RUN apk update && apk add curl
```

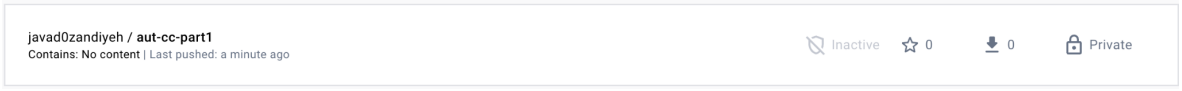
در این image از ورژن سبک alpine به عنوان base image استفاده شده است. سپس با استفاده از دستور apk ابزار curl را بر روی image نصب کرده ایم.

۲. ساخت image:

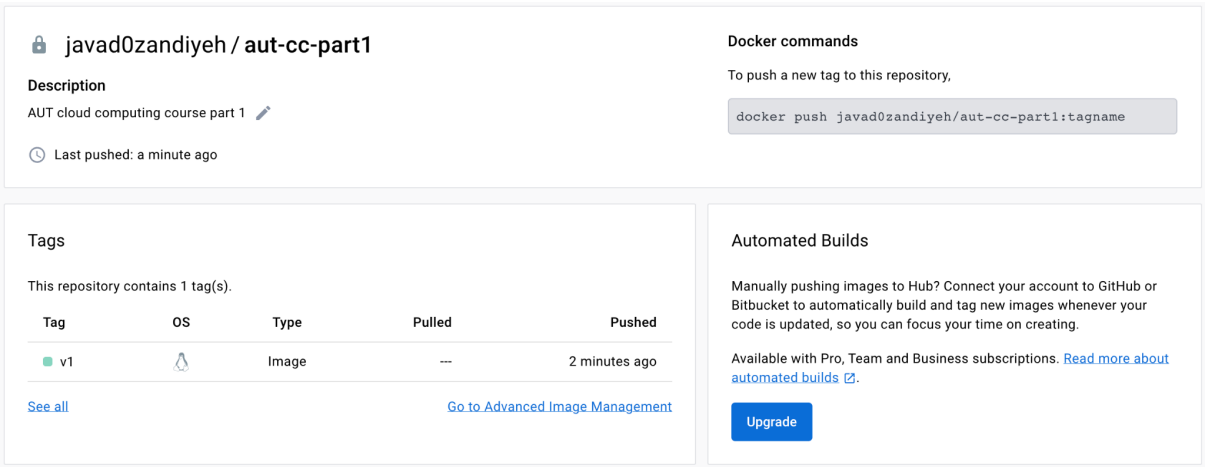
با استفاده از دستور زیر یک image با تگ curl-image:v1 میسازیم:

```
~/Documents/files/cloud-computing/home-works/hw2/9831032_HW2/part1                    01:14:09
docker build -t curl-image:v1 .
[+] Building 3.7s (7/7) FINISHED
 => [internal] load build definition from Dockerfile                                  0.0s
 => => transferring dockerfile: 36B                                                   0.0s
 => [internal] load .dockerignore                                                     0.0s
 => => transferring context: 2B                                                       0.0s
 => [internal] load metadata for docker.io/library/alpine:3.17                        3.6s
 => [auth] library/alpine:pull token for registry-1.docker.io                         0.0s
 => [1/2] FROM docker.io/library/alpine:3.17@sha256:124c7d2707904eea7431fffe91522a01e  0.0s
 => CACHED [2/2] RUN apk update && apk add curl                                       0.0s
 => exporting to image                                                                0.0s
 => => exporting layers                                                               0.0s
 => => writing image sha256:baa958abc29f0fd104e4cabe581e76944ed59e4242a354d25b8cec07  0.0s
 => => naming to docker.io/library/curl-image:v1                                      0.0s
```

۳. ساخت repository در dockerhub:



۴. ارسال image بر روی dockerhub:



```
~/Documents/files/cloud-computing/home-works/hw2/9831032_HW2/part1                    ✔ | 01:17:12
docker tag curl-image:v1 javad0zandiyeh/aut-cc-part1:v1

~/Documents/files/cloud-computing/home-works/hw2/9831032_HW2/part1                    ✔ | 01:24:51
docker push javad0zandiyeh/aut-cc-part1:v1
The push refers to repository [docker.io/javad0zandiyeh/aut-cc-part1]
5f1cde106be0: Pushed
26cbea5cba74: Pushed
v1: digest: sha256:0f3ab5fa97e40f69796f6cdb848ce551dba7b2a8570e605ac9a3da8b716775c4 size: 738
```

۵. نمایش لیست image ها:

```
~/Documents/files/cloud-computing/home-works/hw2/9831032_HW2/part1                    ✔ | 1m 17s ⧖ | 01:26:32
docker images
REPOSITORY                    TAG       IMAGE ID       CREATED         SIZE
curl-image                    v1        baa958abc29f   42 hours ago    12.5MB
javad0zandiyeh/aut-cc-part1   v1        baa958abc29f   42 hours ago    12.5MB
```

۶. دریافت image از dockerhub:

```
~/Documents/files/cloud-computing/home-works/hw2/9831032_HW2/part1                    ✔ | 1m 17s ⧖ | 01:26:32
docker images
REPOSITORY                    TAG       IMAGE ID       CREATED         SIZE
curl-image                    v1        baa958abc29f   42 hours ago    12.5MB
javad0zandiyeh/aut-cc-part1   v1        baa958abc29f   42 hours ago    12.5MB
kicbase/stable                v0.0.39   1465debfde6f   8 days ago      964MB
docker/disk-usage-extension   0.2.5     8f028a5e62df   8 months ago    2.79MB

~/Documents/files/cloud-computing/home-works/hw2/9831032_HW2/part1                    ✔ | 01:30:41
docker pull javad0zandiyeh/aut-cc-part1:v1
v1: Pulling from javad0zandiyeh/aut-cc-part1
Digest: sha256:0f3ab5fa97e40f69796f6cdb848ce551dba7b2a8570e605ac9a3da8b716775c4
Status: Image is up to date for javad0zandiyeh/aut-cc-part1:v1
docker.io/javad0zandiyeh/aut-cc-part1:v1
```

۷. ساخت container و ارسال درخواست:

docker run --name curl-container -it javad0zandiyeh/aut-cc-part1:v1 sh

```
~/Documents/files/cloud-computing/home-works/hw2/9831032_HW2/part1                    ✔ | 9s ⧖ | 01:35:36
docker run --name curl-container -it javad0zandiyeh/aut-cc-part1:v1 sh
/ # curl wttr.in/moon
                     .--------
                .--'  o     .
            .-'     .    o   .
         .-'@    @@@@@@@    .  @
       /@@@  @@@@@@@@@@    @@
     ./    o @@@@@@@@@@    @@@
    /@@  o    @@@@@@@@@@.    @@
   /@@@@    .    @@@@@@@o     @@@
   |@@@@@               . @@@@
  /@@@@@  O  `.-./  .       @@@     Full Moon +
  | @@@@     __`-'        o          6  4:03:04
  |@ @@@           `     o    .     Last Quarter -
  |         @@  @           .-.      1  0:32:30
  \  . @          @@@      `-'
   |       @@    @@@@@ .
   \      @@@@  @\@@    /  .  O
    \  o  @@      \ \  /
     `\       .    .\.-:___   .
        \            `-'
         `-.   o    / |     o
            `-.    /     .
               `--.        .
                   `--------

Follow @igor_chubin for wttr.in updates
```

گام دوم:

۱. ساخت و یا دریافت image ها:

Dockerfile

```
FROM python:3.9.16-alpine

WORKDIR /app

COPY requirements.txt /app

RUN pip3 install --no-cache-dir -r requirements.txt

EXPOSE 8000

WORKDIR /app/api

CMD ["sh", "-c", "python3 manage.py makemigrations; python3 manage.py migrate; python3 manage.py runserver 0.0.0.0:8000"]
```

```
~/Desktop/files/cloud-computing/home-works/hw2/9831032_HW2/part2
docker build -t link-shortener:v1 .
[+] Building 125.0s (10/10) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 319B
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [internal] load metadata for docker.io/library/python:3.9.16-alpine
 => [internal] load build context
 => => transferring context: 37B
 => [1/5] FROM docker.io/library/python:3.9.16-alpine@sha256:a8d325f07e571fbadcffdf9fc6c06eda9cf3a501584d432b73b8e3db179b4e3c
 => => resolve docker.io/library/python:3.9.16-alpine@sha256:a8d325f07e571fbadcffdf9fc6c06eda9cf3a501584d432b73b8e3db179b4e3c
 => => sha256:888a919dc3cbf03f1ac16964f07f7d03a73f80f467e4d447443342a38e0ec5c1 1.37kB / 1.37kB
 => => sha256:533363799c6035e569523d650ce9d1b33064fb281a6d5237ce28272c162e7664 6.32kB / 6.32kB
 => => sha256:74b3a271aef90b25de0dfddcaf22c79ffa1fc32f1e9eac89f21fdd8162c30666 624.23kB / 624.23kB
 => => sha256:c2d35d851133b5fb80e1076cb1ceaabea90ad68de2b037ceee64a16427a43841 11.54MB / 11.54MB
 => => sha256:f64666cf9dbc061382d4a88659b56cd4a06b2af7fb7bf16f8c478c5db33b555d 242B / 242B
 => => sha256:a8d325f07e571fbadcffdf9fc6c06eda9cf3a501584d432b73b8e3db179b4e3c 1.65kB / 1.65kB
 => => extracting sha256:74b3a271aef90b25de0dfddcaf22c79ffa1fc32f1e9eac89f21fdd8162c30666
 => => sha256:6486ba11a934a564b4deba08876e60bd9c46766b45d7da632e38a499e72344d5 2.90MB / 2.90MB
 => => extracting sha256:c2d35d851133b5fb80e1076cb1ceaabea90ad68de2b037ceee64a16427a43841
 => => extracting sha256:f64666cf9dbc061382d4a88659b56cd4a06b2af7fb7bf16f8c478c5db33b555d
 => => extracting sha256:6486ba11a934a564b4deba08876e60bd9c46766b45d7da632e38a499e72344d5
 => [2/5] WORKDIR /app
 => [3/5] COPY requirements.txt /app
 => [4/5] RUN pip3 install --no-cache-dir -r requirements.txt
 => [5/5] WORKDIR /app/api
 => exporting to image
 => => exporting layers
 => => writing image sha256:520d7c7be88ab915873f71fe5d75f63b2e51573cd948998213fc3ad9eb842520
 => => naming to docker.io/library/link-shortener:v1
```

```
~/Desktop/files/cloud-computing/home-works/hw2/9831032_HW2/part2/api
docker pull redis:5.0-alpine3.15
5.0-alpine3.15: Pulling from library/redis
9981e73032c8: Pull complete
d958ce1c7c36: Pull complete
bc31afb0d4b7: Pull complete
93def9bef38d: Pull complete
5ad32ac8dcfd: Pull complete
5747fd610e80: Pull complete
Digest: sha256:ed670ae81585e6bc3d4847fde5de862ac711cdd9e4c3b5fc3acc9f72c4ce5030
Status: Downloaded newer image for redis:5.0-alpine3.15
docker.io/library/redis:5.0-alpine3.15
```

```
~/Desktop/files/cloud-computing/home-works/hw2/9831032_HW2/part2
docker images redis
REPOSITORY   TAG             IMAGE ID       CREATED         SIZE
redis        5.0-alpine3.15  3c03664688c9   12 months ago   29.4MB

~/Desktop/files/cloud-computing/home-works/hw2/9831032_HW2/part2
docker images link-shortener:v1
REPOSITORY       TAG       IMAGE ID       CREATED             SIZE
link-shortener   v1        92d0c424f49b   About an hour ago   97.6MB
```

۲. ساخت network برای ارتباط بین container ها:

```
~/Desktop/files/cloud-computing/home-works/hw2/9831032_HW2
docker network create link-shortener
5677dd316ea77d0c8f3f293d7d3966375c59d9c9c1219443b14b51600550464c
```

۳. ساخت container ها:

docker run -it --name link-shortener-1 -p 8000:8000 --network link-shortener -v $(pwd)/api:/app/api link-shortener:v1

```
~/Desktop/files/cloud-computing/home-works/hw2/9831032_HW2/part2
docker run -it --name link-shortener-1 -p 8000:8000 --network link-shortener -v $(pwd)/api:/app/api link-shortener:v1
No changes detected
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  No migrations to apply.
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
April 13, 2023 - 11:57:39
Django version 4.2, using settings 'api.settings'
Starting development server at http://0.0.0.0:8000/
Quit the server with CONTROL-C.
```

docker run -d --restart=always --name redis-1 --network link-shortener -p 6379:6379 -v $(pwd)/api/redis_data:/data redis:5.0-alpine3.15

```
~/Desktop/files/cloud-computing/home-works/hw2/9831032_HW2/part2
docker run -d --restart=always --name redis-1 --network link-shortener -p 6379:6379 -v $(pwd)/api/redis_data:/data redis:5.0-alpine3.15

837074f9a8217ae14a57b4534eb47e92f07d7c79021a13c3721de1a8e00923d6
```

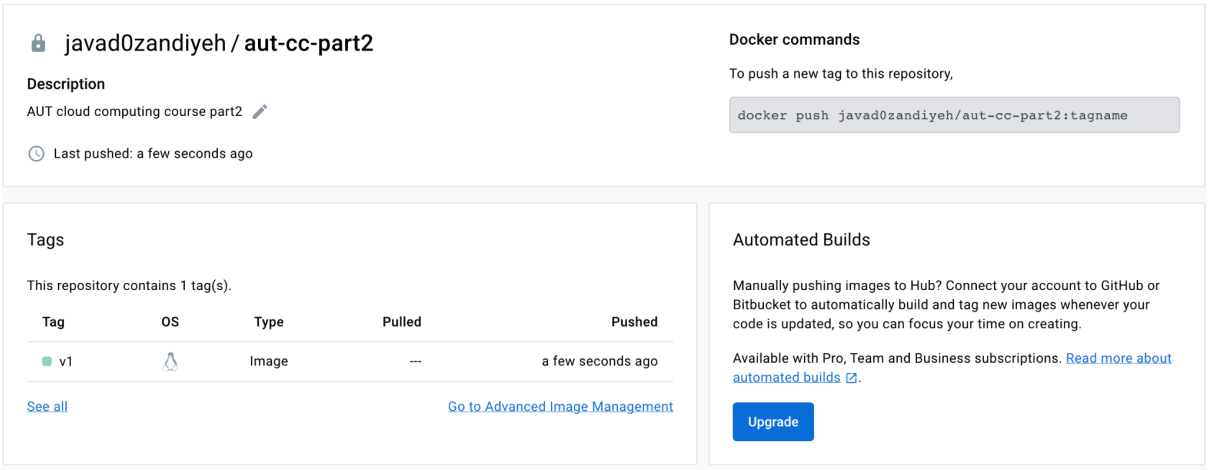
```
~/Desktop/files/cloud-computing/home-works/hw2/9831032_HW2/part2
docker ps -a
CONTAINER ID   IMAGE                  COMMAND                  CREATED          STATUS             PORTS                    NAMES
837074f9a821   redis:5.0-alpine3.15   "docker-entrypoint.s…"   33 seconds ago   Up 32 seconds      0.0.0.0:6379->6379/tcp   redis-1
7c38d8d17b47   link-shortener:v1      "sh -c 'python3 mana…"   2 hours ago      Up About an hour   0.0.0.0:8000->8000/tcp   link-shortener-1
```

۴. نمایش میزان منابع استفاده شده:

از دستور docker stats استفاده میکنیم

```
CONTAINER ID   NAME              CPU %     MEM USAGE / LIMIT     MEM %     NET I/O           BLOCK I/O     PIDS
4afd6415a09d   link-shortener-1  2.90%     68.05MiB / 3.841GiB   1.73%     50.5kB / 37.8kB   0B / 508kB    3
837074f9a821   redis-1           0.24%     1.52MiB / 3.841GiB    0.04%     1.89kB / 540B     0B / 0B       4
```

۵. ارسال image بر روی dockerhub:



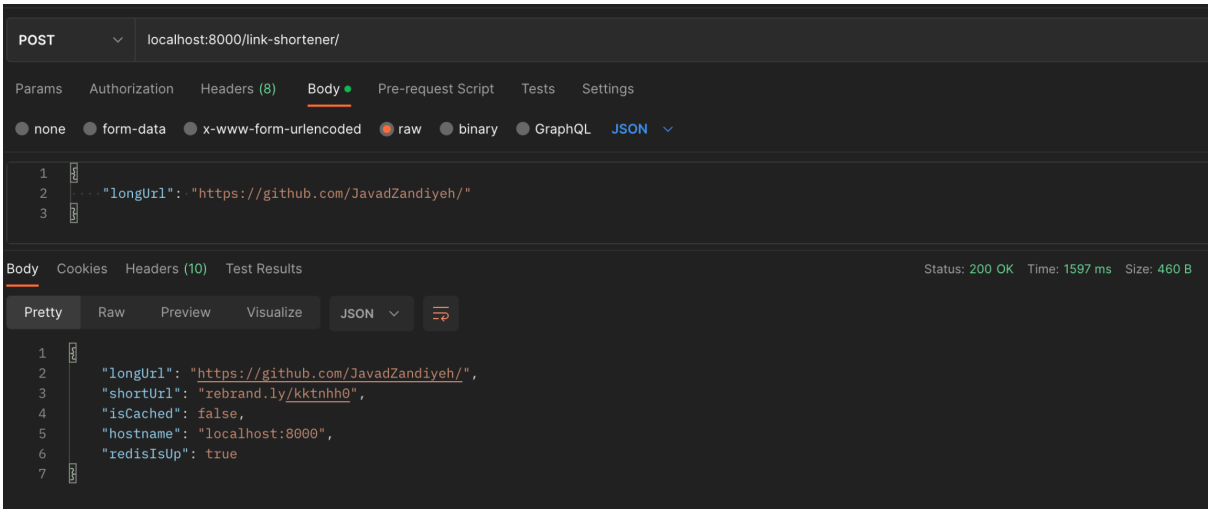

```
~/Desktop/files/cloud-computing/home-works/hw2/9831032_HW2/part2
docker tag link-shortener:v1 javad0zandiyeh/aut-cc-part2:v1
```

```
~/Desktop/files/cloud-computing/home-works/hw2/9831032_HW2/part2
docker push javad0zandiyeh/aut-cc-part2:v1
The push refers to repository [docker.io/javad0zandiyeh/aut-cc-part2]
68331e5cd86c: Pushed
a520399c575b: Pushed
cbba2c69983d: Pushed
dff24544ade2: Pushed
108a28221de8: Mounted from library/python
abf11f15b319: Mounted from library/python
dd4f1e4bf0c6: Mounted from library/python
2028de9f94ee: Mounted from library/python
26cbea5cba74: Mounted from javad0zandiyeh/aut-cc-part1
v1: digest: sha256:ad294c677eeb42194dafaf49b6944e55b16cdf16a8af91a155f9dec85476152c size: 2200
```

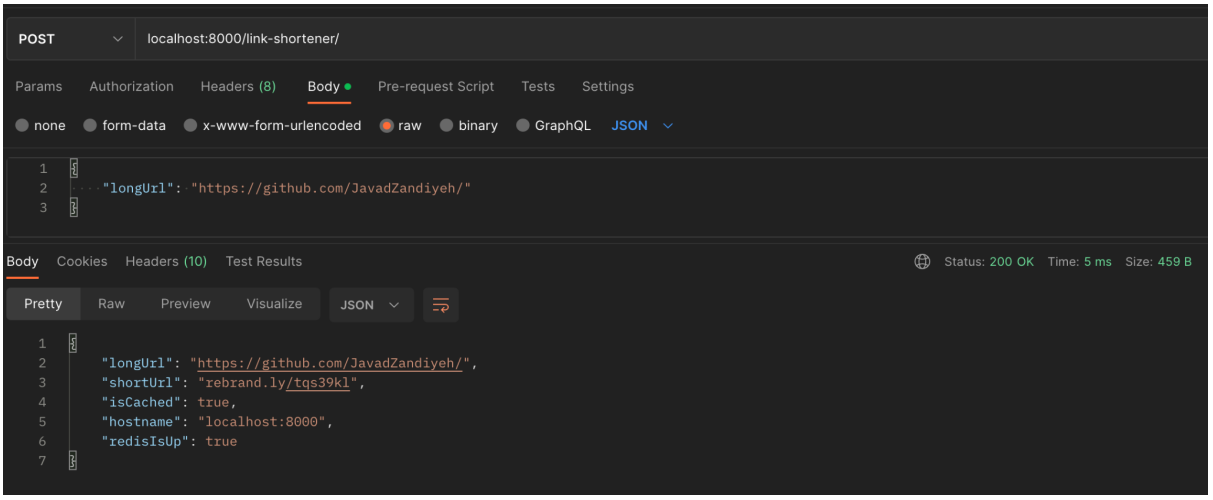
```
~/Desktop/files/cloud-computing/home-works/hw2/9831032_HW2/part2
docker pull javad0zandiyeh/aut-cc-part2:v1
v1: Pulling from javad0zandiyeh/aut-cc-part2
Digest: sha256:ad294c677eeb42194dafaf49b6944e55b16cdf16a8af91a155f9dec85476152c
Status: Image is up to date for javad0zandiyeh/aut-cc-part2:v1
docker.io/javad0zandiyeh/aut-cc-part2:v1
```

۶. نمایش خروجی:

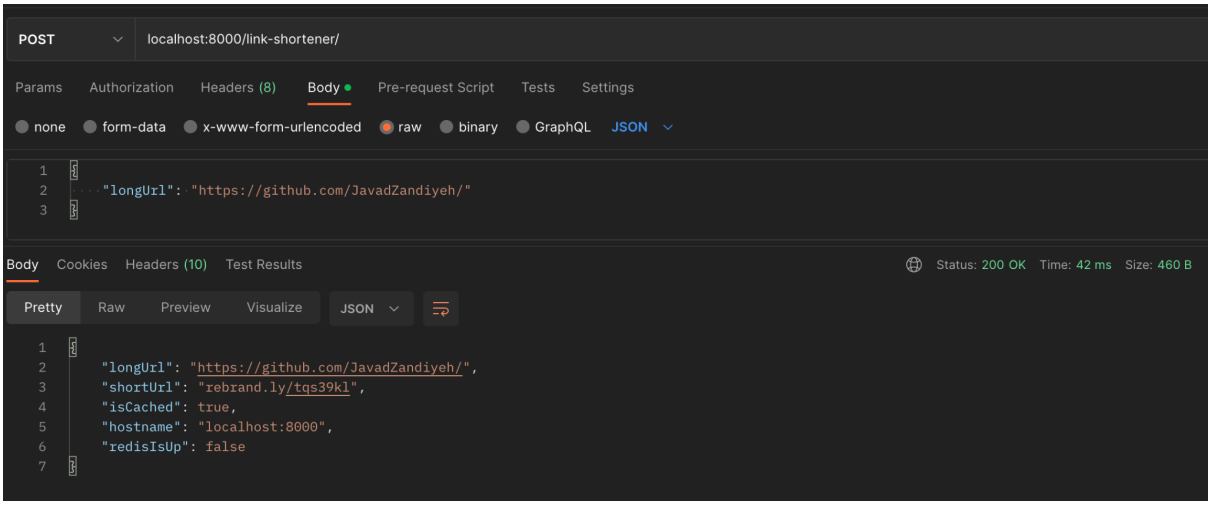
حالت اول (ردیس فعال است، اما دیتا برای اولین مرتبه است که درخواست میشود و در cache وجود ندارد):



حالت دوم (ردیس فعال است و دیتای در مدت ۵ دقیقه گذشته درخواست شده است و در cache دخیره شده است)



حالت سوم (ردیس فعال نیست، اما در هنوز دیتایی که خواسته شده است در cache وجود دارد و به صورت offline میتوان دیتا را خواند)



نمایش پنل Rebrandly:

۷. نمایش اطلاعات image:

```
~/Desktop/files/cloud-computing/home-works/hw2/9831032_HW2/part2
docker inspect javad0zandiyeh/aut-cc-part2:v1
[
    {
        "Id": "sha256:92d0c424f49b53b022b500e14814730e2c927e6d2afe6668a456647edfdfc1c9",
        "RepoTags": [
            "javad0zandiyeh/aut-cc-part2:v1",
            "link-shortener:v1"
        ],
        "RepoDigests": [
            "javad0zandiyeh/aut-cc-part2@sha256:ad294c677eeb42194dafaf49b6944e55b16cdf16a8af91a155f9dec85476152c"
        ],
        "Parent": "",
        "Comment": "buildkit.dockerfile.v0",
        "Created": "2023-04-13T15:27:18.6929861747",
        "Container": "",
        "ContainerConfig": {
            "Hostname": "",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": null,
            "Cmd": null,
            "Image": "",
            "Volumes": null,
            "WorkingDir": "",
            "Entrypoint": null,
            "OnBuild": null,
            "Labels": null
        },
        "DockerVersion": "",
        "Author": "",
        "Config": {
            "Hostname": "",
            "Domainname": "",
            "User": "",
            "AttachStdin": false,
            "AttachStdout": false,
            "AttachStderr": false,
            "ExposedPorts": {
                "8000/tcp": {}
            },
            "Tty": false,
            "OpenStdin": false,
            "StdinOnce": false,
            "Env": [
                "PATH=/usr/local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin",
                "LANG=C.UTF-8",
                "GPG_KEY=E3FF2839C048B25C084DEBE9B26995E310250568",
                "PYTHON_VERSION=3.9.16",
                "PYTHON_PIP_VERSION=22.0.4",
                "PYTHON_SETUPTOOLS_VERSION=58.1.0",
                "PYTHON_GET_PIP_URL=https://github.com/pypa/get-pip/raw/d5cb0afaf23b8520f1bbcfed521017b4a95f5c01/public/get-pip.py",
                "PYTHON_GET_PIP_SHA256=394be00f13fa1b9aaa47e911bdb59a09c3b2986472130f30aa0bfaf7f3980637"
            ],
            "Cmd": [
                "sh",
                "-c",
                "python3 manage.py makemigrations; python3 manage.py migrate; python3 manage.py runserver 0.0.0.0:8000"
            ],
            "ArgsEscaped": true,
            "Image": "",
            "Volumes": null,
```

```
            "Volumes": null,
            "WorkingDir": "/app/api",
            "Entrypoint": null,
            "OnBuild": null,
            "Labels": null
        },
        "Architecture": "arm64",
        "Variant": "v8",
        "Os": "linux",
        "Size": 97571838,
        "VirtualSize": 97571838,
        "GraphDriver": {
            "Data": {
                "LowerDir": "/var/lib/docker/overlay2/c4i2zcbbdfjicul1fug7yfi5v/diff:/var/lib/docker/overlay2/w6edmpxjzitkuvnanv9ef9nq6/diff:/va
ay2/nq6ji109x576prl1on72is4fm/diff:/var/lib/docker/overlay2/24ae9e65385ea702b295dfcfaa35d40e0f3ce6bbca40f499b67805f6cbb4e748/diff:/var/lib/docke
ded601b6619f340ff4f68836a5cf4e801f26beee19f184090d6675993/diff:/var/lib/docker/overlay2/bb70f718b5b8be31ca38472dc77adad90d737400b5b049c4fc2ff2e6
/lib/docker/overlay2/cec98b7291b29c09fa9ce64a388fe5e72d9151470d2fbe951df6df266c5413f2/diff:/var/lib/docker/overlay2/3a8244eae78a9d7d76a4a1ee4f77
9e4f725f9ada38f022/diff",
                "MergedDir": "/var/lib/docker/overlay2/q3kiv2kojh4kv2v7j1cvyl3pa/merged",
                "UpperDir": "/var/lib/docker/overlay2/q3kiv2kojh4kv2v7j1cvyl3pa/diff",
                "WorkDir": "/var/lib/docker/overlay2/q3kiv2kojh4kv2v7j1cvyl3pa/work"
            },
            "Name": "overlay2"
        },
        "RootFS": {
            "Type": "layers",
            "Layers": [
                "sha256:26cbea5cba74143fbe6f584f5fc5321543155aedc4a434fcaa63b643877b5a74",
                "sha256:2028de9f94ee2aec394312cb4a0623f85564ee0d6b8333f99399f3ffa78e966a",
                "sha256:dd4f1e4bf0c69e5a517d287134ba682739c7f0d991fe3ff219873b759c67d9a1",
                "sha256:abf11f15b319a665a0cafbfed3900baaeaa03c03379aa668fedc12f3ac599963",
                "sha256:108a28221de84e5f584fb80a880d77a61b7962c66ca3518bda3e49a977909f98",
                "sha256:dff24544ade2b812057d6044e7ffa40fc2b7b2f30e51efd42a998a622b18b8e7",
                "sha256:cbba2c69983d29f05ac1b1f8c35e66e6d6ec18294cec69c25277b5a43be3ddd0",
                "sha256:a520399c575bdac30a52c0220d8ab42a973a0eab111c13ad7e1649f1659109c1",
                "sha256:68331e5cd86c4fd204da42e6ba23497c5bc6c49959dd946c535683f01c753c3a"
            ]
        },
        "Metadata": {
            "LastTagTime": "2023-04-13T16:43:07.46192671Z"
        }
    }
]
```


گام سوم:

در این گام ابتدا manifect های لازم را پیاده سازی می‌کنیم و سپس لازم است که تغییراتی در کد اجرایی برنامه داده شود تا دیتای‌های ست شده در فایل‌های Secret , configMap در برنامه به عنوان متغیرهای محیطی خوانده و استفاده شود.

۱. پیاده سازی manifest های لازم برای برنامه بک‌اند

۱.۱ فایل ConfigMapAndSecret

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: link-shortener-configmap
data:
  app.ip: "0.0.0.0"
  app.port: "8000"
  redis.serviceName: "redis-service"
  redis.port: "6379"
  cache.timeOut: "5"
```

```yaml
apiVersion: v1
kind: Secret
metadata:
  name: link-shortener-secret
type: Opaque
stringData:
  rebrandly.key: YTVhZmM0OTliZTE1NDcxOGFjYzRiNjEwNWI4ZDM1MDY=
  rebrandly.api: aHR0cHM6Ly9hcGkucmVicmFuZGx5LmNvbS92MS9saW5rcw==
```

۱.۲ فایل Deployment

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: link-shortener-deployment
spec:
  replicas: 2
  selector:
    matchLabels:
      app: link-shortener-deployment
  template:
    metadata:
      labels:
        app: link-shortener-deployment
    spec:
      volumes:
        - name: api-volume
          hostPath:
            path: /app/api
      containers:
        - name: link-shortener-deployment
          image: docker.io/library/link-shortener:v2
          ports:
            - containerPort: 8000
          volumeMounts:
            - name: api-volume
              mountPath: /app/api
          resources:
            limits:
              memory: "128Mi"
              cpu: "500m"
          env:
            - name: APP_IP
              valueFrom:
                configMapKeyRef:
                  name: link-shortener-configmap
                  key: app.ip
            - name: APP_PORT
              valueFrom:
                configMapKeyRef:
                  name: link-shortener-configmap
```

```
            key: app.port
        - name: REDIS_HOST
          valueFrom:
            configMapKeyRef:
              name: link-shortener-configmap
              key: redis.serviceName
        - name: REDIS_PORT
          valueFrom:
            configMapKeyRef:
              name: link-shortener-configmap
              key: redis.port
        - name: CACHE_TIME_OUT
          valueFrom:
            configMapKeyRef:
              name: link-shortener-configmap
              key: cache.timeOut
        - name: REBRANDLY_API
          valueFrom:
            secretKeyRef:
              name: link-shortener-secret
              key: rebrandly.api
        - name: REBRANDLY_KEY
          valueFrom:
            secretKeyRef:
              name: link-shortener-secret
              key: rebrandly.key
```

۱.۳ فایل Service

```
apiVersion: v1
kind: Service
metadata:
  name: link-shortener-service
spec:
  selector:
    app: link-shortener-deployment
  type: LoadBalancer
  ports:
  - protocol: TCP
    port: 8000
    targetPort: 8000
    nodePort: 30000
```

توضیحات: در ابتدا در بخش secret , configmap متغیرها و کانفیگ‌هایی که نیاز داریم در طول برنامه استفاده کنیم را ست میکنیم. به عنوان مثال ip , key برای اتصال و درخواست به سرویس نگهداری و کوتاه کننده لینک را در آن ذخیره میکنیم.

در بخش deployment باید به مواردی از جمله اسم image که از روی آن قرار است pod ساخته شود، متغیرهای ست شده در configmap و secret و پورت داخلی کلاستر که قرار است کانتینر را روی آن پورت بالا بیاوریم و volume هایی که ست کرده ایم و ... دقت کرد.

در بخش service نیز برای آنکه بتوان از خارج از کلاستر به pod ریکوئست را رساند، پورت های داخلی کلاستر و پورتی که از خارج از کلاستر دیده میشود را قرار میدهیم. چون قرار است این سرویس از بیرون ریکوئست ها را هندل کند باید تایپ load balancer باشد.

۲. پیاده سازی manifest های لازم برای redis

۲.۱ فایل ConfigMap

```yaml
apiVersion: v1
kind: ConfigMap
metadata:
  name: redis-configmap
data:
  redis.port: "6379"
```

۲.۲ فایل Deployment

```yaml
apiVersion: apps/v1
kind: Deployment
metadata:
  name: redis-deployment
spec:
  replicas: 1
  selector:
    matchLabels:
      app: redis-deployment
  template:
    metadata:
      labels:
        app: redis-deployment
    spec:
      volumes:
        - name: redis-volume
          persistentVolumeClaim:
            claimName: redis-persist-volume-claim
      containers:
        - name: redis-deployment
          image: docker.io/library/redis:5.0-alpine3.15
          ports:
            - containerPort: 6379
          volumeMounts:
            - name: redis-volume
              mountPath: /data
          resources:
            limits:
              memory: "128Mi"
              cpu: "500m"
          env:
            - name: REDIS_PORT
              valueFrom:
                configMapKeyRef:
                  name: redis-configmap
                  key: redis.port
```

۲.۳ فایل Service

```yaml
apiVersion: v1
kind: Service
metadata:
  name: redis-service
spec:
  selector:
    app: redis-deployment
  type: ClusterIP
  ports:
  - protocol: TCP
    port: 6379
    targetPort: 6379
```

۲.۴ فایل PersistVolume

```yaml
apiVersion: v1
kind: PersistentVolume
metadata:
  name: redis-persist-volume
spec:
  capacity:
    storage: 512Mi
  accessModes:
    - ReadWriteMany
  hostPath:
    path: /data
  storageClassName: shared
```

۲.۵ فایل PersistVolumeClaim

```yaml
apiVersion: v1
kind: PersistentVolumeClaim
metadata:
  name: redis-persist-volume-claim
spec:
  accessModes:
    - ReadWriteMany
  storageClassName: shared
  resources:
    requests:
      storage: 256Mi
```

توضیحات: در این بخش همانند بخش قبل به service , deployment , configmap نیاز داریم. تفاوتی که در سرویس داریم این است که type آنرا به صورت ClusterIp گذاشته ایم چون قرار نیست از بیرون به آن دسترسی داشت. علاوه بر آنها نیاز است تا برای persist کردن دیتای داخل ردیس، یک volume را به آن اختصاص دهیم تا در صورت پایین آمدن یک پاد، پاد بعدی که بالا می‌آید همچنان به دیتای داخل redis دسترسی داشته باشد.

۳. اعمال تغییرات در فایل‌های پروژه برای استفاده از دیتای درون configmap , secret

در هرجایی که نیاز به enviroment variable هایی که ست کرده ایم داشته باشیم میتوان از دستور os.getenv در پایتون استفاده کرد. در Dockerfile نیز میتوان با علامت ${اسم متغیر} به این دیتا دسترسی داشت. به عنوان نمونه:

```python
DEFAULT_AUTO_FIELD = 'django.db.models.BigAutoField'

# Redis settings
CACHES = {
    'default': {
        'BACKEND': 'django_redis.cache.RedisCache',
        'LOCATION': f"redis://{os.getenv('REDIS_HOST')}:{int(os.getenv('REDIS_PORT'))}/0",
        'OPTIONS': {
            'CLIENT_CLASS': 'django_redis.client.DefaultClient',
        },
        'TIMEOUT': int(os.getenv('CACHE_TIME_OUT')) * 60,
    }
}

# Use the cache for session storage
SESSION_ENGINE = 'django.contrib.sessions.backends.cache'
SESSION_CACHE_ALIAS = 'default'
```

Dockerfile

part3 > docker-part > Dockerfile

```
1  FROM python:3.9.16-alpine
2
3  ENV PYTHONDONTWRITEBYTECODE=1
4  ENV PYTHONUNBUFFERED=1
5
6  WORKDIR /app
7
8  COPY requirements.txt /app
9
10 RUN pip3 install --no-cache-dir -r requirements.txt
11
12 EXPOSE ${APP_PORT}
13
14 WORKDIR /app/api
15
16 CMD ["sh", "-c", "python3 manage.py makemigrations; python3 manage.py migrate; python3 manage.py runserver ${APP_IP}:${APP_PORT}"]
17
```

نکات:

الف. برای delete , apply کردن فایل‌هایی که گفته شد، دو عدد bash script نوشته شده اند که راحت‌تر عملیات انجام شود.

ب. دقت شود که در هنگام mount کردن volume آدرس host باید آدرس داخلی کلاستر باشد نه کامپیوتر ما. در نتیجه نیاز است که یک mount دیگر نیز بین کامپیوتر و minikube صورت گیرد. برای این تمرین به صورت‌های زیر انجام میشود:

```
~/Desktop/files/cloud-computing/home-works/hw2/9831032_HW2/part3/docker-part/api
17:49:08
minikube mount $(pwd):/app/api
Mounting host path /Users/javad/Desktop/files/cloud-computing/home-works/hw2/9831032_HW2/part3/docker-part/api into VM as /app/api ...
  ▪ Mount type:   9p
  ▪ User ID:      docker
  ▪ Group ID:     docker
  ▪ Version:      9p2000.L
  ▪ Message Size: 262144
  ▪ Options:      map[]
  ▪ Bind Address: 127.0.0.1:60329
Userspace file server: ufs starting
Successfully mounted /Users/javad/Desktop/files/cloud-computing/home-works/hw2/9831032_HW2/part3/docker-part/api to /app/api

NOTE: This process must stay alive for the mount to be accessible ...
```

پ. برای آنکه درخواست از خارج کلاستر به پاد برسد نیاز است که یک tunnel زده شود. به این صورت:

```
~/De/files/cloud-computing/home-works/hw2/9831032_HW2/part3/kubernetes-part/api
minikube tunnel
Tunnel successfully started

NOTE: Please do not close this terminal as this process must stay alive for the tunnel to be accessible ...
```

ت. برای آنکه بتوان درون minikube به image لوکالی که ساخته ایم دسترسی داشته باشیم باید از دستور minikube load به همراه اسم image استفاده کنیم:

```
~/Desktop/files/cloud-computing/home-works/hw2/9831032_HW2
minikube load link-shortener:v2
```

ث. برای استفاده از dashboard در داخل minikube از دستور minikube dashboard استفاده میکنیم تا بتوانیم وضیعت پادها و سرویس‌ها و ... را مشاهده کنیم.

۴. ساخت manifest ها و مشاهده نتایج

```
~/De/f/cloud-computing/home-works/hw2/9831032_HW2/part3/kubernetes-part/redis
11:23:46
./apply.sh
configmap/redis-configmap created
persistentvolume/redis-persist-volume created
persistentvolumeclaim/redis-persist-volume-claim created
deployment.apps/redis-deployment created
service/redis-service created

~/De/f/cloud-computing/home-works/hw2/9831032_HW2/part3/kubernetes-part/redis
11:23:59
cd ../api

~/De/files/cloud-computing/home-works/hw2/9831032_HW2/part3/kubernetes-part/api
11:24:04
./apply.sh
configmap/link-shortener-configmap created
secret/link-shortener-secret created
deployment.apps/link-shortener-deployment created
service/link-shortener-service created
```

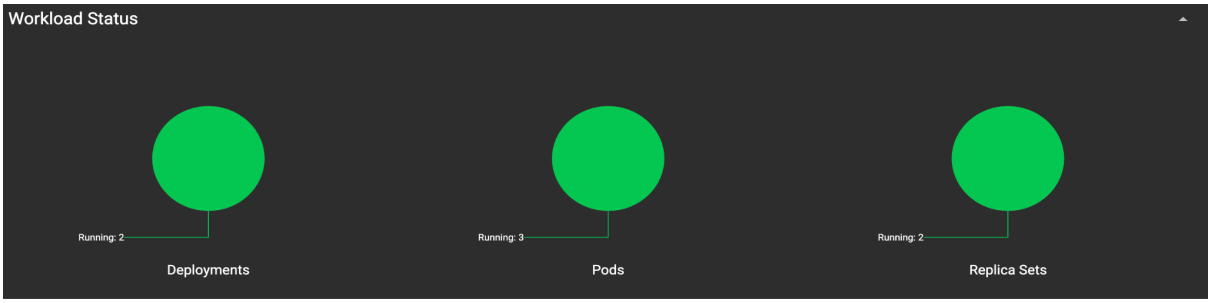
```
~/Desktop/files/cloud-computing/home-works/hw2/9831032_HW2/part3/docker-part/api
11
:24:56
minikube mount $(pwd):/app/api
Mounting host path /Users/javad/Desktop/files/cloud-computing/home-works/hw2/9831032_HW2/part3/docker-part/api into VM as /app/api ...
  ▪ Mount type:   9p
  ▪ User ID:      docker
  ▪ Group ID:     docker
  ▪ Version:      9p2000.L
  ▪ Message Size: 262144
  ▪ Options:      map[]
  ▪ Bind Address: 127.0.0.1:50072
Userspace file server: ufs starting
Successfully mounted /Users/javad/Desktop/files/cloud-computing/home-works/hw2/9831032_HW2/part3/docker-part/api to /app/api

NOTE: This process must stay alive for the mount to be accessible ...
```

```
~/Desktop/files/cloud-computing/home-works/hw2/9831032_HW2
minikube tunnel
Tunnel successfully started

NOTE: Please do not close this terminal as this process must stay alive for the tunnel to be accessible ...

Starting tunnel for service link-shortener-service.
```

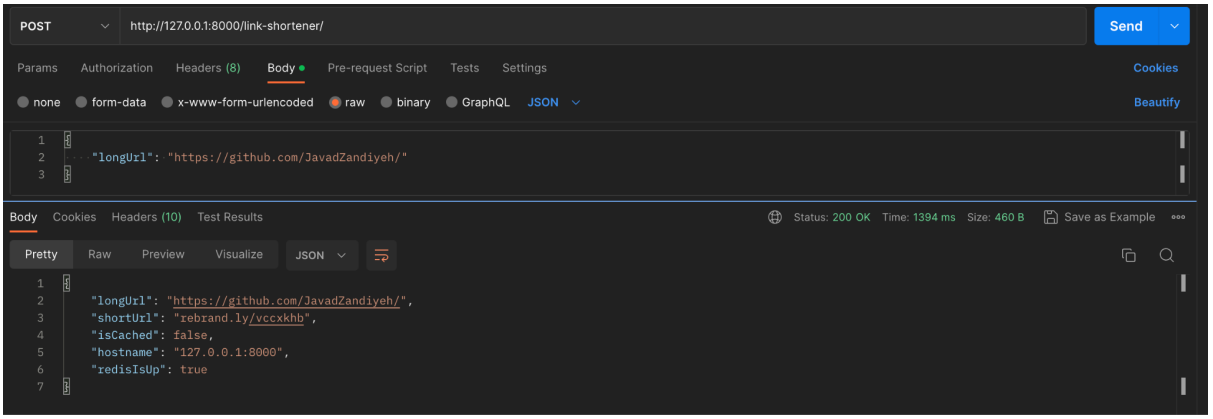
مشاهده منابع ایجاد شده:

```
~/De/files/cloud-computing/home-works/hw2/9831032_HW2/part3/kubernetes-part/api
kubectl get all
NAME                                          READY   STATUS    RESTARTS        AGE
pod/link-shortener-deployment-8765b6cdf-68nfv 1/1     Running   4 (5m20s ago)   6m9s
pod/link-shortener-deployment-8765b6cdf-v4g9v 1/1     Running   4 (5m23s ago)   6m9s
pod/redis-deployment-5dc86bc75f-bgtsv         1/1     Running   0               6m18s

NAME                             TYPE           CLUSTER-IP      EXTERNAL-IP   PORT(S)          AGE
service/kubernetes               ClusterIP      10.96.0.1       <none>        443/TCP          18h
service/link-shortener-service   LoadBalancer   10.102.51.233   127.0.0.1     8000:30000/TCP   6m9s
service/redis-service            ClusterIP      10.106.178.211  <none>        6379/TCP         6m17s

NAME                                     READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/link-shortener-deployment 2/2    2            2           6m9s
deployment.apps/redis-deployment          1/1    1            1           6m18s

NAME                                                DESIRED   CURRENT   READY   AGE
replicaset.apps/link-shortener-deployment-8765b6cdf 2         2         2       6m9s
replicaset.apps/redis-deployment-5dc86bc75f         1         1         1       6m18s
```

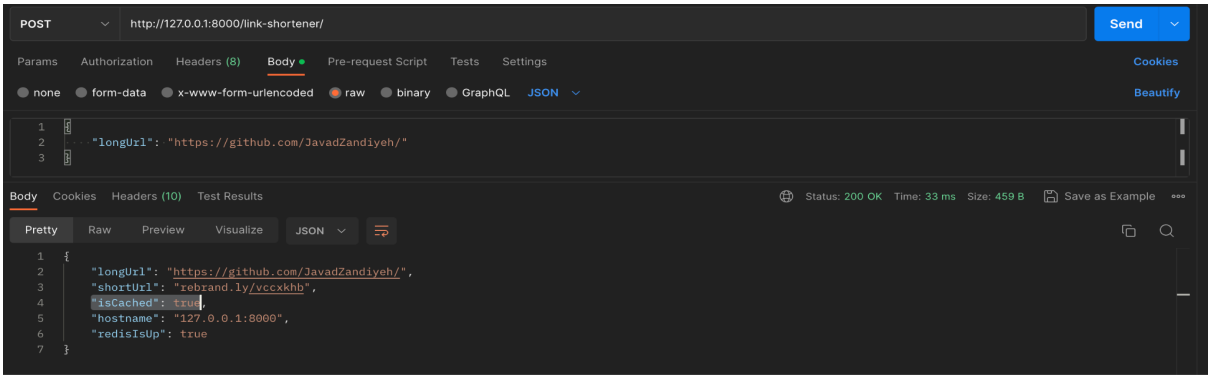
مشاهده منابع ایجاد شده از روی dashboard



ارسال درخواست به بک‌اند

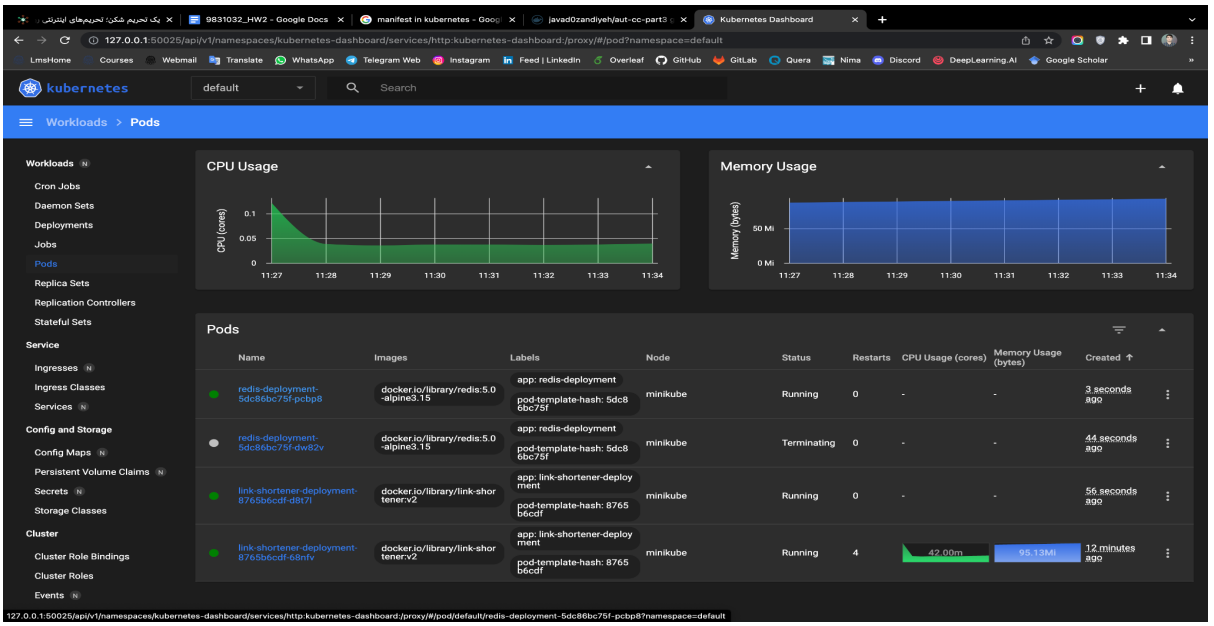


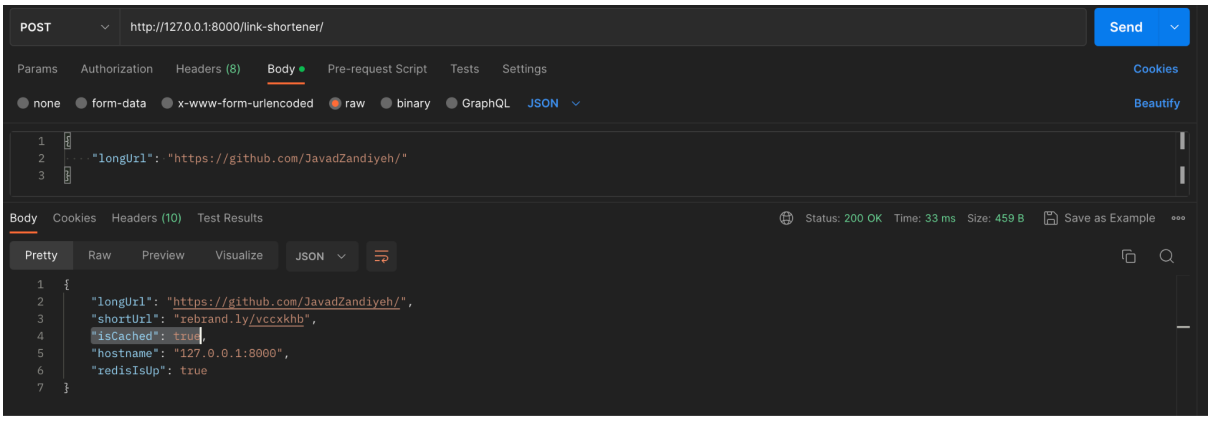
تست redis



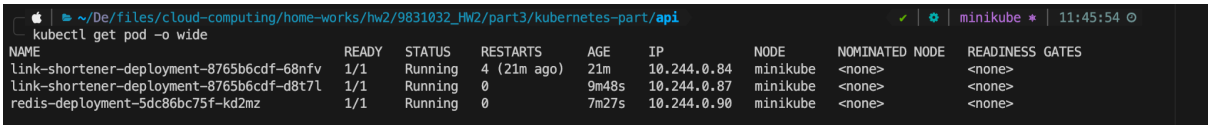
تست persist شدن دیتا

- ابتدا پاد مربوطه را دیلیت میکنیم. Deployment سعی میکند که پاد جدید را جایگزین کند. اگر پاد جدید به دیتای redis دسترسی داشته باشد، یعنی دیتا در جریان kill شدن pod قبلی از بین نرفته است:





مشاهده آدرس ip پادها با استفاده از endpoint ها



گام چهارم:

در این مرحله قصد داریم که image مرحله اول را روی cluster اجرا کنیم و بجای اینکه از بیرون ریکوئست بزنیم، از داخل cluster درخواست بزنیم. با این کار صحت کد را میخواهیم که تست کنیم.

اضافه کردن سرویس link-shortener-service به ALLOWED_HOSTS برای اینکه بتوان درخواست را از طریق آن به pod رساند.

```
ALLOWED_HOSTS = [
    'link-shortener-service'
]
```

در این مرحله image را روی کلاستر به صورت iterative اجرا میکنیم.

```
~/De/files/cloud-computing/home-works/hw2/9831032_HW2/part3/kubernetes-part/api   INT ✗ | minikube ⎈ | 17:02:40
k run curl-deployment -it --image=javad0zandiyeh/aut-cc-part1:v1 --port=2020
```

در گام آخر درخواست خود را به بکند ارسال میکنیم

```
/ # curl --location 'http://link-shortener-service:8000/link-shortener/' --header 'Content-Type: application/json' --data '{"longUrl": "https://github
.com/JavadZandiyeh"}'
{"longUrl":"https://github.com/JavadZandiyeh","shortUrl":"rebrand.ly/slv97as","isCached":false,"hostname":"link-shortener-service:8000","redisIsUp":tr
ue}/ #
```

در لاگ pod هم میتوان مشاهده کرد که درخواست به پاد رسیده و به درستی پاسخ را ارسال کرده است

```
[08/May/2023 13:34:08] "POST /link-shortener/ HTTP/1.1" 200 153
```

مشاهده منابع ایجاد شده: میتوان دید که curl-deployment ایجاد شده است.

```
kubectl get all
NAME                                          READY   STATUS    RESTARTS   AGE
pod/curl-deployment                           1/1     Running   0          30s
pod/link-shortener-deployment-6cd5ff588f-bkdr8   1/1     Running   0          39s
pod/link-shortener-deployment-6cd5ff588f-jjxgn   1/1     Running   0          39s
pod/redis-deployment-7b5445464c-jxw5f         1/1     Running   0          46s

NAME                              TYPE           CLUSTER-IP     EXTERNAL-IP   PORT(S)          AGE
service/kubernetes                ClusterIP      10.96.0.1      <none>        443/TCP          23h
service/link-shortener-service    LoadBalancer   10.96.170.94   127.0.0.1     8000:30000/TCP   38s
service/redis-service             ClusterIP      10.111.31.40   <none>        6379/TCP         46s

NAME                                     READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/link-shortener-deployment   2/2     2            2           39s
deployment.apps/redis-deployment            1/1     1            1           46s

NAME                                                DESIRED   CURRENT   READY   AGE
replicaset.apps/link-shortener-deployment-6cd5ff588f   2         2         2       39s
replicaset.apps/redis-deployment-7b5445464c            1         1         1       46s
```

همزمان log هر دو پادی که بالا آمده است پروژه بکند را به مشاهده میکنیم:

```
~/De/f/cl/h/hw2/9831032_HW2   ✓ | minikube ⎈ | 17:18:51
k logs -f link-shortener-deployment-6cd5ff588f-bkdr8
No changes detected
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  No migrations to apply.
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
May 08, 2023 - 13:45:03
Django version 4.2, using settings 'api.settings'
Starting development server at http://0.0.0.0:8000/
Quit the server with CONTROL-C.

[08/May/2023 13:49:42] "POST /link-shortener/ HTTP/1.1" 200 155
[08/May/2023 13:49:47] "POST /link-shortener/ HTTP/1.1" 200 154
```

```
~/De/f/cl/h/hw2/9831032_HW2   ✓ | minikube ⎈ | 17:19:12
k logs -f link-shortener-deployment-6cd5ff588f-jjxgn
No changes detected
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, sessions
Running migrations:
  No migrations to apply.
Watching for file changes with StatReloader
Performing system checks...

System check identified no issues (0 silenced).
May 08, 2023 - 13:45:03
Django version 4.2, using settings 'api.settings'
Starting development server at http://0.0.0.0:8000/
Quit the server with CONTROL-C.

[08/May/2023 13:49:54] "POST /link-shortener/ HTTP/1.1" 200 155
```

از ۳ درخواست زده شده، دوتای آن به پاد سمت چپ و یکی از آن به پاد سمت راست ارسال شده است.

این هم ۳ درخواستی که زده شده است:

```
k run curl-deployment -it --image=javad0zandiyeh/aut-cc-part1:v1 --port=2020
If you don't see a command prompt, try pressing enter.
/ # curl --location 'http://link-shortener-service:8000/link-shortener/' --header 'Content-Type: application/json' --data '{"longUrl": "https://github
.com/JavadZandiyeh1/"}'
{"longUrl":"https://github.com/JavadZandiyeh1/","shortUrl":"rebrand.ly/a686gfd","isCached":false,"hostname":"link-shortener-service:8000","redisIsUp":
/ # curl --location 'http://link-shortener-service:8000/link-shortener/' --header 'Content-Type: application/json' --data '{"longUrl": "https://github
.com/JavadZandiyeh1/"}'
{"longUrl":"https://github.com/JavadZandiyeh1/","shortUrl":"rebrand.ly/a686gfd","isCached":true,"hostname":"link-shortener-service:8000","redisIsUp":t
/ # curl --location 'http://link-shortener-service:8000/link-shortener/' --header 'Content-Type: application/json' --data '{"longUrl": "https://github
.com/JavadZandiyeh2/"}'
{"longUrl":"https://github.com/JavadZandiyeh2/","shortUrl":"rebrand.ly/iq884kg","isCached":false,"hostname":"link-shortener-service:8000","redisIsUp":
true}/ #
```