

به نام خدا  
دانشگاه صنعتی امیرکبیر (پلی تکنیک تهران)  
دانشکده مهندسی کامپیوتر



دانشگاه صنعتی امیرکبیر  
( پلی تکنیک تهران )

## مبانی پردازش ابری

گزارش کار تمرین ۲  
داکر و مقدمات کوبرنتیز

استاد درس: دکتر جوادی

محمد رضا شهرستانی

۹۷۲۸۰۵۴

نیم سال دوم ۱۴۰۰-۱۴۰۱

## گام اول

(۱) ارسال ایمج ساخته شده بر روی داکرهاب و نتیجه آن:

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  1: docker
PS C:\Users\m shahr\Desktop\cc2\part1> docker image push 9728054/alpinewithcurl:1.0
The push refers to repository [docker.io/9728054/alpinewithcurl]
f9ae51cbf27a: Pushed
4fc242d58285: Mounted from library/alpine
1.0: digest: sha256:6fcf752202a70f3f0c29b37ebe663c9a50cf877b41d85565bebb11a91819ae80 size: 738
PS C:\Users\m shahr\Desktop\cc2\part1>

```

(۲) دریافت و اجرای ایمج ساخته شده از داکرهاب:

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  1: docker
PS C:\Users\m shahr\Desktop\cc2\part1> docker run -it 9728054/alpinewithcurl:1.0

```

(۳) اجرا و خروجی curl:

```

PROBLEMS  OUTPUT  DEBUG CONSOLE  TERMINAL  1: docker
PS C:\Users\m shahr\Desktop\cc2\part1> docker run -it 9728054/alpinewithcurl:1.0
/ # curl google.com
<HTML><HEAD><meta http-equiv="content-type" content="text/html; charset=utf-8">
<TITLE>301 Moved</TITLE></HEAD><BODY>
<H1>301 Moved</H1>
The document has moved
<A HREF="http://www.google.com/">here</A>.
</BODY></HTML>
/ #

```

به علت استفاده از `-it` بخش دوم و سوم به هم متصل شده اند.

## گام دوم

(۱) build کردن ایمج با استفاده از Dockerfile ساخته شده:

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 2: powershell +
PS C:\Users\m shahr\Desktop\cc2\part2> docker build -t image2 .
[+] Building 85.7s (10/10) FINISHED
=> [internal] load build definition from Dockerfile 0.1s
=> => transferring dockerfile: 169B 0.0s
=> [internal] load .dockerignore 0.1s
=> => transferring context: 2B 0.0s
=> [internal] load metadata for docker.io/library/python:3.7-alpine 5.6s
=> [auth] library/python:pull token for registry-1.docker.io 0.0s
=> [internal] load build context 0.1s
=> => transferring context: 230B 0.0s
=> [1/4] FROM docker.io/library/python:3.7-alpine@sha256:e578c3754c15062c810c0b45bf3425c81f70e79d4a52a2f0b20f420d3cd597d8 62.8s
=> => resolve docker.io/library/python:3.7-alpine@sha256:e578c3754c15062c810c0b45bf3425c81f70e79d4a52a2f0b20f420d3cd597d8 0.0s
=> => sha256:e578c3754c15062c810c0b45bf3425c81f70e79d4a52a2f0b20f420d3cd597d8 1.65kB / 1.65kB 0.0s
=> => sha256:026c4bc93d48679ebf4263bf2286479c6973b4826a33c4cb716fe3aeb74f6f15 1.37kB / 1.37kB 0.0s
=> => sha256:e3f6bf100ea40388de70a2436a9877196660d0add146cd43afc0bbbc9aaaa63f 7.56kB / 7.56kB 0.0s
=> => sha256:c86a7acf952c7faf3ea2f4ba61a552116295312cb031ba3f1dc6a7509a55e63a 667.03kB / 667.03kB 14.5s
=> => sha256:51acb131917ed5977f8285164e675412565ef6ed8988b90be5b226d2a8841444 11.19MB / 11.19MB 60.0s
=> => sha256:988db338fb470e9c4b4d80e23a79d1ee3edbb559c4d4ff802d4886e8c62309ed 229B / 229B 1.5s
=> => sha256:9e85ba3d63c3595ba4ac082fb66ad27a11c87bb215ca1f5befe45e77236098 2.87MB / 2.87MB 20.1s
=> => extracting sha256:c86a7acf952c7faf3ea2f4ba61a552116295312cb031ba3f1dc6a7509a55e63a 0.6s
=> => extracting sha256:51acb131917ed5977f8285164e675412565ef6ed8988b90be5b226d2a8841444 1.2s
=> => extracting sha256:988db338fb470e9c4b4d80e23a79d1ee3edbb559c4d4ff802d4886e8c62309ed 0.0s
=> => extracting sha256:9e85ba3d63c3595ba4ac082fb66ad27a11c87bb215ca1f5befe45e77236098 0.6s
=> [2/4] WORKDIR /directory 0.3s
=> [3/4] COPY . . 0.2s
=> [4/4] RUN python -m pip install -r requirement.txt 15.6s
=> exporting to image 0.7s
=> => exporting layers 0.7s
=> => writing image sha256:a71dca090251e09e4e5989cb6c203bf3e424d0b60d3afd665513c5fb0eff27c6 0.0s
=> => naming to docker.io/library/image2 0.0s

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
PS C:\Users\m shahr\Desktop\cc2\part2>

```

(۲) ارسال ایمج ساخته شده بر روی داکرهاب و نتیجه آن:

```

PS C:\Users\m shahr\Desktop\cc2\part2> docker tag image2 9728054/part2:1.0
PS C:\Users\m shahr\Desktop\cc2\part2> docker image push 9728054/part2:1.0
The push refers to repository [docker.io/9728054/part2]
3c021faa9776: Pushed
f95b1de02bda: Pushed
20328744096b: Pushed
40a7faafac16: Mounted from library/python
f3bc2019fc26: Mounted from library/python
7fb86ec33c3d: Mounted from library/python
fbd7d5451c69: Mounted from library/python
4fc242d58285: Layer already exists
1.0: digest: sha256:a098ec5fff80858981b6a96f1e275f2de3c2df45829c2b4d9e846f2103014f55 size: 1993
PS C:\Users\m shahr\Desktop\cc2\part2>

```

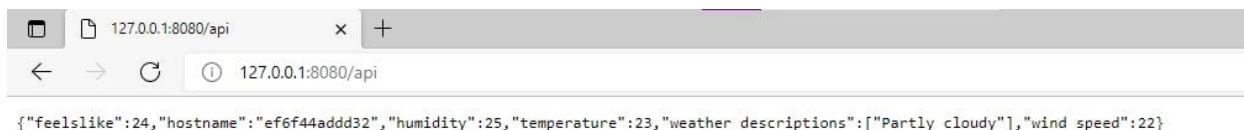
۳) در صورتی که پروژه خود را با استفاده از ایمپج ساخته شده بر روی سیستم شخصی خود تست کردید،

تصاویر مربوطه را قرار دهید:

اجرا کردن ایمپج روی سیستم:

```
PS C:\Users\m_shahr\Desktop\cc2> docker run -p 8080:8080 -it image2
* Serving Flask app 'p2' (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: off
* 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://127.0.0.1:8080
* Running on http://172.17.0.2:8080 (Press CTRL+C to quit)
172.17.0.1 - - [21/Apr/2022 14:24:37] "GET / HTTP/1.1" 404 -
172.17.0.1 - - [21/Apr/2022 14:24:37] "GET /favicon.ico HTTP/1.1" 404 -
172.17.0.1 - - [21/Apr/2022 14:24:42] "GET /شماره HTTP/1.1" 404 -
172.17.0.1 - - [21/Apr/2022 14:24:54] "GET /api HTTP/1.1" 200 -
```

تست لوکال هاست:



The screenshot shows a web browser window with the address bar set to `127.0.0.1:8080/api`. The page content displays a JSON response: `{"feelslike":24,"hostname":"ef6f44add32","humidity":25,"temperature":23,"weather_descriptions":["Partly cloudy"],"wind_speed":22}`.

## گام سوم

(۱) با استفاده از دستور `kubectl get` صحت ایجاد منابع بر روی کلاستر را نمایش دهید:

```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL 1: powershell
PS C:\Users\m shahr\Desktop\cc2\part3> kubectl get pods
NAME                                READY STATUS RESTARTS AGE
part3-5b74cbd9b6-p2gnw             1/1   Running 0       4m9s
part3-5b74cbd9b6-vpq6b             1/1   Running 0       4m9s
PS C:\Users\m shahr\Desktop\cc2\part3>

```

```

PS C:\Users\m shahr\Desktop\cc2\part3> kubectl get configmap
NAME      DATA AGE
kube-root-ca.crt 1    31h
part3      1    19m
PS C:\Users\m shahr\Desktop\cc2\part3>

```

```

PS C:\Users\m shahr\Desktop\cc2\part3> kubectl get deployment
NAME READY UP-TO-DATE AVAILABLE AGE
part3 2/2    2           2      20m
PS C:\Users\m shahr\Desktop\cc2\part3>

```

```

PS C:\Users\m shahr\Desktop\cc2\part3> kubectl get svc
NAME      TYPE        CLUSTER-IP    EXTERNAL-IP  PORT(S)    AGE
kubernetes ClusterIP   10.96.0.1     <none>       443/TCP    31h
part3     ClusterIP   10.110.215.101 <none>       80/TCP     20m
PS C:\Users\m shahr\Desktop\cc2\part3>

```

(۲) آدرس IP پادها و نحوه برقراری ارتباط میان آن‌ها و سرویس ساخته شده:

```

PS C:\Users\m shahr\Desktop\cc2\part3> kubectl get ep
NAME      ENDPOINTS                                AGE
kubernetes 192.168.49.2:8443                      31h
part3     172.17.0.2:8080,172.17.0.3:8080        21m
PS C:\Users\m shahr\Desktop\cc2\part3>

```

علت استفاده از واحد سرویس: وقتی تنهایی Pod را بالا می آوریم، اگر به هر دلیل حذف بشوند و دوباره بالا بیاوریم‌شان، IP Address جدیدی داده می‌شود و کنترل آن سخت است و هر بار باید حواس‌مان باشد که چه IP ای اختصاص داده می‌شود. البته ممکن است IP یکسانی بگیرند ولی ممکن است IP شان عوض شود به جای اینکه هر دفعه چک کنیم که IP هاشان درست تنظیم شده و ارتباط بین‌شان درست است یک service می‌سازیم که اتوماتیک آن را کنترل کند.

## گام چهارم

(۱) با استفاده از دستور `kubectl get` صحت ایجاد پاد بر روی کلاستر را نمایش دهید:

```
PS C:\Users\m_shahr\Desktop\cc2\part3> kubectl get pods
NAME                READY   STATUS    RESTARTS   AGE
netutils            1/1     Running   1 (2m53s ago)  7m29s
part3-5b74cbd9b6-p2gnw  1/1     Running   0           21m
part3-5b74cbd9b6-vpq6b  1/1     Running   0           21m
PS C:\Users\m_shahr\Desktop\cc2\part3>
```

(۲) چند مورد استفاده از درخواست‌های ارسال شده به پروژه خود را همراه با توزیع بار میان پادها نشان دهید:

```
PS C:\Users\m_shahr\Desktop\cc2\part3> kubectl run -it netutils --image=9728054/alpinewithcurl:1.0
If you don't see a command prompt, try pressing enter.
/ # curl part3/api
{"feelslike":27,"hostname":"part3-5b74cbd9b6-p2gnw","humidity":14,"temperature":29,"weather_descriptions":["Partly cloudy"],"wind_speed":15}
/ # curl part3/api
{"feelslike":27,"hostname":"part3-5b74cbd9b6-vpq6b","humidity":14,"temperature":29,"weather_descriptions":["Partly cloudy"],"wind_speed":15}
/ # curl part3/api
{"feelslike":27,"hostname":"part3-5b74cbd9b6-p2gnw","humidity":14,"temperature":29,"weather_descriptions":["Partly cloudy"],"wind_speed":15}
/ # curl part3/api
{"feelslike":27,"hostname":"part3-5b74cbd9b6-vpq6b","humidity":14,"temperature":29,"weather_descriptions":["Partly cloudy"],"wind_speed":15}
/ # curl part3/api
{"feelslike":27,"hostname":"part3-5b74cbd9b6-vpq6b","humidity":14,"temperature":29,"weather_descriptions":["Partly cloudy"],"wind_speed":15}
/ #
```

(۳) دستور مورد استفاده برای اجرا کردن ایمج گام اول:

```
PS C:\Users\m_shahr\Desktop\cc2\part3> kubectl run -it netutils --image=9728054/alpinewithcurl:1.0
If you don't see a command prompt, try pressing enter.
```

به علت استفاده از `-it` بخش دوم و سوم به هم متصل شده اند.