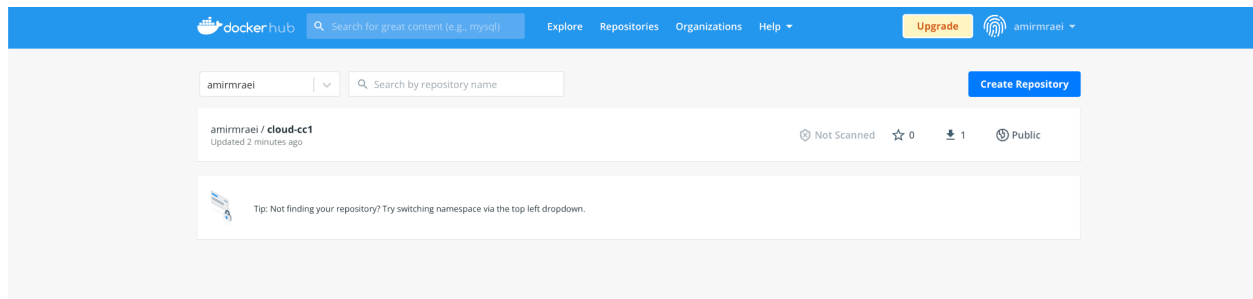


به نام خدا  
امیرمحمد راعی - ۹۷۳۱۰۸۳  
تمرین دوم: داکر و مقدمات کوبرنتیز

گام اول -

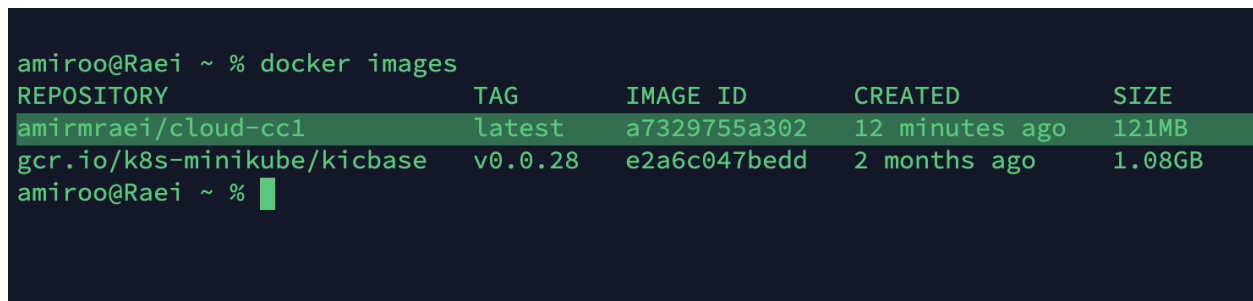
۱-

برای ارسال image بر روی داکرهاب در مرحله اول نیاز است یک account در این سایت بسازیم. سپس باید اسم image خود را به گونه username/repo\_name تغییر دهیم که username نام کاربری ما در این سایت است و repo\_name نام دایرکتوری ما است. سپس با دستور docker push amirmraei/cloud-cc1 ایمیج را به داکرهاب ارسال می‌کنیم.



۲-

اکنون باید image خود را از داکرهاب بگیریم. دستور docker push amirmraei/cloud-cc1 ایمیج را از داکر پوش کرده و به لیست image های ما اضافه می‌کند.



۳-

برای اجرای ایمیج ubuntu ای که داریم دستور amirmraei/cloud-cc1 را وارد می‌کنیم. و به صفحه ترمینال ubuntu می‌رویم. سپس دستور curl را برای google اجرا می‌کنیم.

```

amiroo@Raei ~ % docker run -it amirmraei/cloud-cc1
root@ec29bdf532430:/# ls
bin boot dev etc home lib lib32 lib64 libx32 media mnt opt proc root run sbin srv sys tmp usr var
root@ec29bdf532430:/# 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>
root@ec29bdf532430:/# █

```

گام دوم -

۱-

در این بخش با استفاده از دستور `docker build --tag amirmraei/cloud-cc2` ایمپج خود را build می‌کنیم.

```

amiroo@Raei step2 % docker build --tag amirmraei/cloud-cc2 .
[+] Building 15.2s (11/11) FINISHED
=> [internal] load build definition from Dockerfile                                0.0s
=> => transferring dockerfile: 276B                                              0.0s
=> [internal] load .dockerignore                                                  0.0s
=> => transferring context: 2B                                                    0.0s
=> [internal] load metadata for docker.io/library/python:3.8-slim-buster        3.3s
=> [auth] library/python:pull token for registry-1.docker.io                   0.0s
=> [internal] load build context                                                 0.4s
=> => transferring context: 17.83MB                                              0.4s
=> [1/5] FROM docker.io/library/python:3.8-slim-buster@sha256:95db858dd3cb0587820edd001cb5fc46715529ed609e56344715f9adfc6d42fe 0.0s
=> => resolve docker.io/library/python:3.8-slim-buster@sha256:95db858dd3cb0587820edd001cb5fc46715529ed609e56344715f9adfc6d42fe 0.0s
=> CACHED [2/5] WORKDIR /app                                                    0.0s
=> [3/5] COPY requirements.txt requirements.txt                                  0.1s
=> [4/5] RUN pip3 install -r requirements.txt                                    10.7s
=> [5/5] COPY . .                                                                0.3s
=> exporting to image                                                            0.2s
=> => exporting layers                                                            0.2s
=> => writing image sha256:bc04fa51f35eadf940278e91b56dfba1e2b0f699b98e26c93463f3342edef08d 0.0s
=> => naming to docker.io/amirmraei/cloud-cc2                                  0.0s
amiroo@Raei step2 % docker images
REPOSITORY          TAG          IMAGE ID          CREATED           SIZE
amirmraei/cloud-cc2 latest       bc04fa51f35e     30 seconds ago   145MB
amirmraei/cloud-cc1 latest       a7329755a302     24 minutes ago   121MB
gcr.io/k8s-minikube/kicbase v0.0.28     e2a6c047bedd     2 months ago     1.08GB
amiroo@Raei step2 % █

```

حال چک می‌کنیم که آیا image ساخته شده است یا خیر.

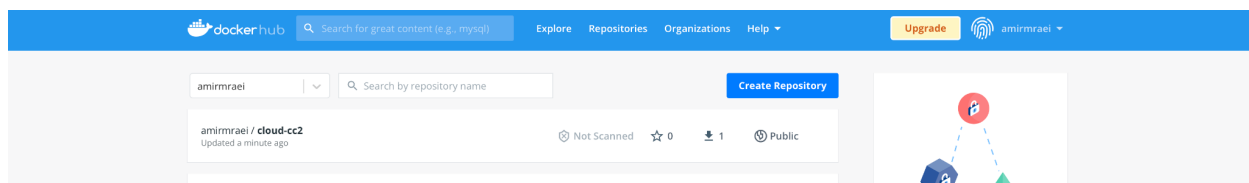
```

amiroo@Raei step2 % docker images
REPOSITORY          TAG          IMAGE ID          CREATED           SIZE
amirmraei/cloud-cc2 latest       bc04fa51f35e     3 minutes ago    145MB
amirmraei/cloud-cc1 latest       a7329755a302     27 minutes ago    121MB
gcr.io/k8s-minikube/kicbase v0.0.28     e2a6c047bedd     2 months ago     1.08GB
amiroo@Raei step2 % █

```

۲-

در این گام با همان دستور `docker push amirmraei/cloud-cc2` ایمپج خود را به داکرهاب ارسال می‌کنیم.



۳-

در این قسمت برای تست کردن صحت برنامه ایجاد شده docker image خود را run می‌کنیم و سپس درخواست به port را می‌فرستیم.

برای اجرای image خود دستور `docker run -d -p 8080:5000 amirmraei/cloud-cc2` وارد می‌کنیم.

```
amiroo@Raei step2 % docker run -d -p 8080:5000 amirmraei/cloud-cc2
cb9ff344048e6b32e9fe90ff80f148887260bb40b06fab16bce539d9afb1e75a
amiroo@Raei step2 % docker ps
CONTAINER ID   IMAGE                                COMMAND                  CREATED        STATUS        PORTS                               NAMES
cb9ff344048e   amirmraei/cloud-cc2                "python3 -m flask ru..." 31 seconds ago Up 30 seconds 0.0.0.0:8080->5000/tcp          modest_agnes1
amiroo@Raei step2 % curl localhost:8080
{"hostname":"amiroo@Raei","time":{"abbreviation":"CET","client_ip":"8.20.127.32","datetime":"2021-12-09T22:09:36.645508+01:00","day_of_week":4,"day_of_year":343,"dst":false,"dst_from":null,"dst_offset":0,"dst_until":null,"raw_offset":3600,"timezone":"Europe/Rome","unixtime":1639884176,"utc_datetime":"2021-12-09T21:09:36.645508+00:00","utc_offset":"+01:00","week_number":49}}
```

گام سوم -

۱-

برنامه دارای deployment, service, configmap می‌باشد. اکنون با دستور `kubectl get all` اطلاعات هر کدام از این بخش‌ها را به دست می‌آوریم.

```
amiroo@Raei step2 % kubectl get all
NAME                                     READY   STATUS    RESTARTS   AGE
pod/cloud-step3-56986765f5-qvv86       1/1     Running   0           33m
pod/cloud-step3-56986765f5-s5sg9       1/1     Running   0           33m

NAME                                     TYPE          CLUSTER-IP   EXTERNAL-IP   PORT(S)          AGE
service/cloud-step3-service            LoadBalancer  10.103.72.236 192.168.0.10  6000:31184/TCP   33m
service/kubernetes                     ClusterIP      10.96.0.1     <none>         443/TCP          34m

NAME                                     READY   UP-TO-DATE   AVAILABLE   AGE
deployment.apps/cloud-step3            2/2     2             2           33m

NAME                                     DESIRED   CURRENT   READY   AGE
replicaset.apps/cloud-step3-56986765f5 2         2         2       33m
amiroo@Raei step2 %
```

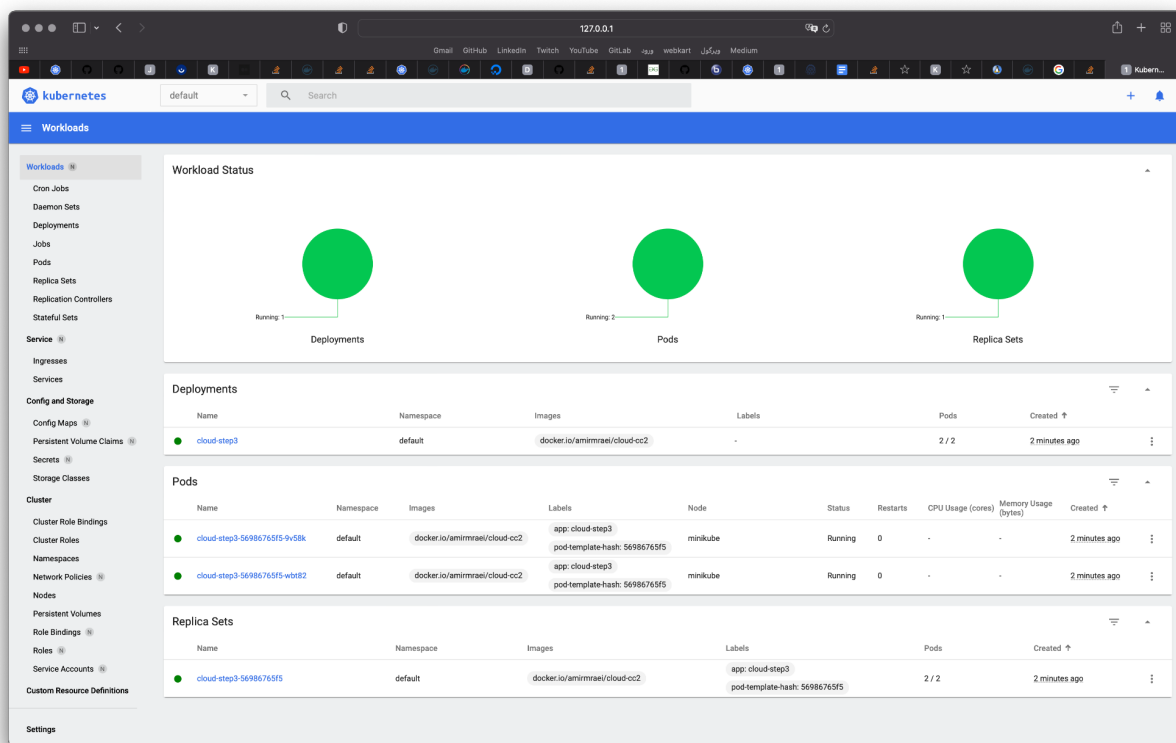
۲-

در اینجا با استفاده از دستور `kubectl get pods -o wide` می‌توان از ip هر کدام از pod ها آگاه شد و از آنجا که ما تعداد replica ها را برابر با ۲ ست کردیم ۲ pod خواهیم داشت. نحوه ارتباط آن‌ها

هم به اینگونه است که اگر یکی از این پادها با مشکل مواجه شود مثلاً پاک شود و یا به سبب یک interrupt متوقف شود، دیگر pod جای آن را می‌گیرد و شروع به کار می‌کند. همچنین با استفاده از دستور `get ep` می‌توان از اطلاعات endpoint مطلع شد. در اینجا ما دو pod داریم که هرکدام ip مخصوص به هم دارند و نحوه ارتباط و جابه‌جایی بین pod ها را نشان می‌دهد.

```
amiroo@Raei step2 % kubectl get pods -o wide
NAME                                READY   STATUS    RESTARTS   AGE   IP            NODE     NOMINATED NODE   READINESS GATES
cloud-step3-56986765f5-9v58k        1/1     Running   0           99s   172.17.0.4    minikube <none>         <none>
cloud-step3-56986765f5-wbt82        1/1     Running   0           99s   172.17.0.3    minikube <none>         <none>
amiroo@Raei step2 %
```

```
amiroo@Raei ~ % kubectl get ep
NAME                                ENDPOINTS                                AGE
cloud-step3-service                172.17.0.3:5000,172.17.0.5:5000        11h
kubernetes                          192.168.49.2:8443                      11h
visitors-service                    172.17.0.6:65432,172.17.0.8:65432      11h
amiroo@Raei ~ %
```



گام چهارم -

-۱

فایل deployment را برای این برنامه می‌نویسیم و سپس اجرا می‌کنیم. حال باید در بخشی pod ببینیم که آیا ساخته شده است یا خیر.

```
amiroo@Raei ~ % kubectl get po
```

NAME	READY	STATUS	RESTARTS	AGE
cloud-step3-56986765f5-7knjd	1/1	Running	0	10m
cloud-step3-56986765f5-dg62m	1/1	Running	0	10m
ubuntu-5895b8b77-449fp	1/1	Running	0	12m
visitors-f79b7894-fjs59	1/1	Running	0	14m
visitors-f79b7894-lbspd	1/1	Running	0	14m

```
amiroo@Raei ~ %
```

می‌بینیم که به درستی پاد ساخته شده است.

۲-

سرویس تعیین شده به نام cloud-step3-service نوشته شده است و port آن برابر با 6000 است. حال باید به service و port آن دستور curl بزنیم.

```
root@ubuntu-5895b8b77-449fp:~# curl cloud-step3-service:6000
{"hostname":"amiroo@Raei","time":{"abbreviation":"CET","client_ip":"8.20.127.32","datetime":"2021-12-10T00:52:39.508401+01:00","day_of_week":5,"day_of_year":344,"dst":false,"dst_from":null,"dst_offset":0,"dst_until":null,"raw_offset":3600,"timezone":"Europe/Rome"},"unixtime":1639093959,"utc_datetime":"2021-12-09T23:52:39.508401+00:00","utc_offset":"+01:00","week_number":49}}
root@ubuntu-5895b8b77-449fp:~# curl cloud-step3-service:6000
{"hostname":"amiroo@Raei","time":{"abbreviation":"CET","client_ip":"8.20.127.32","datetime":"2021-12-10T00:52:40.515728+01:00","day_of_week":5,"day_of_year":344,"dst":false,"dst_from":null,"dst_offset":0,"dst_until":null,"raw_offset":3600,"timezone":"Europe/Rome"},"unixtime":1639093960,"utc_datetime":"2021-12-09T23:52:40.515728+00:00","utc_offset":"+01:00","week_number":49}}
root@ubuntu-5895b8b77-449fp:~# curl cloud-step3-service:6000
{"hostname":"amiroo@Raei","time":{"abbreviation":"CET","client_ip":"8.20.127.32","datetime":"2021-12-10T00:52:41.482923+01:00","day_of_week":5,"day_of_year":344,"dst":false,"dst_from":null,"dst_offset":0,"dst_until":null,"raw_offset":3600,"timezone":"Europe/Rome"},"unixtime":1639093961,"utc_datetime":"2021-12-09T23:52:41.482923+00:00","utc_offset":"+01:00","week_number":49}}
root@ubuntu-5895b8b77-449fp:~# curl cloud-step3-service:6000
{"hostname":"amiroo@Raei","time":{"abbreviation":"CET","client_ip":"8.20.127.32","datetime":"2021-12-10T00:52:42.448757+01:00","day_of_week":5,"day_of_year":344,"dst":false,"dst_from":null,"dst_offset":0,"dst_until":null,"raw_offset":3600,"timezone":"Europe/Rome"},"unixtime":1639093962,"utc_datetime":"2021-12-09T23:52:42.448757+00:00","utc_offset":"+01:00","week_number":49}}
root@ubuntu-5895b8b77-449fp:~# curl cloud-step3-service:6000
{"hostname":"amiroo@Raei","time":{"abbreviation":"CET","client_ip":"8.20.127.32","datetime":"2021-12-10T00:52:43.387098+01:00","day_of_week":5,"day_of_year":344,"dst":false,"dst_from":null,"dst_offset":0,"dst_until":null,"raw_offset":3600,"timezone":"Europe/Rome"},"unixtime":1639093963,"utc_datetime":"2021-12-09T23:52:43.387098+00:00","utc_offset":"+01:00","week_number":49}}
root@ubuntu-5895b8b77-449fp:~# curl cloud-step3-service:6000
{"hostname":"amiroo@Raei","time":{"abbreviation":"CET","client_ip":"8.20.127.32","datetime":"2021-12-10T00:52:44.305815+01:00","day_of_week":5,"day_of_year":344,"dst":false,"dst_from":null,"dst_offset":0,"dst_until":null,"raw_offset":3600,"timezone":"Europe/Rome"},"unixtime":1639093964,"utc_datetime":"2021-12-09T23:52:44.305815+00:00","utc_offset":"+01:00","week_number":49}}
root@ubuntu-5895b8b77-449fp:~# curl cloud-step3-service:6000
{"hostname":"amiroo@Raei","time":{"abbreviation":"CET","client_ip":"8.20.127.32","datetime":"2021-12-10T00:52:45.221935+01:00","day_of_week":5,"day_of_year":344,"dst":false,"dst_from":null,"dst_offset":0,"dst_until":null,"raw_offset":3600,"timezone":"Europe/Rome"},"unixtime":1639093965,"utc_datetime":"2021-12-09T23:52:45.221935+00:00","utc_offset":"+01:00","week_number":49}}
root@ubuntu-5895b8b77-449fp:~# curl cloud-step3-service:6000
{"hostname":"amiroo@Raei","time":{"abbreviation":"CET","client_ip":"8.20.127.32","datetime":"2021-12-10T00:52:46.163379+01:00","day_of_week":5,"day_of_year":344,"dst":false,"dst_from":null,"dst_offset":0,"dst_until":null,"raw_offset":3600,"timezone":"Europe/Rome"},"unixtime":1639093966,"utc_datetime":"2021-12-09T23:52:46.163379+00:00","utc_offset":"+01:00","week_number":49}}
root@ubuntu-5895b8b77-449fp:~# curl cloud-step3-service:6000
{"hostname":"amiroo@Raei","time":{"abbreviation":"CET","client_ip":"8.20.127.32","datetime":"2021-12-10T00:52:47.088819+01:00","day_of_week":5,"day_of_year":344,"dst":false,"dst_from":null,"dst_offset":0,"dst_until":null,"raw_offset":3600,"timezone":"Europe/Rome"},"unixtime":1639093967,"utc_datetime":"2021-12-09T23:52:47.088819+00:00","utc_offset":"+01:00","week_number":49}}
root@ubuntu-5895b8b77-449fp:~# curl cloud-step3-service:6000
{"hostname":"amiroo@Raei","time":{"abbreviation":"CET","client_ip":"8.20.127.32","datetime":"2021-12-10T00:52:48.273330+01:00","day_of_week":5,"day_of_year":344,"dst":false,"dst_from":null,"dst_offset":0,"dst_until":null,"raw_offset":3600,"timezone":"Europe/Rome"},"unixtime":1639093968,"utc_datetime":"2021-12-09T23:52:48.273330+00:00","utc_offset":"+01:00","week_number":49}}
root@ubuntu-5895b8b77-449fp:~# curl cloud-step3-service:6000
{"hostname":"amiroo@Raei","time":{"abbreviation":"CET","client_ip":"8.20.127.32","datetime":"2021-12-10T00:52:49.256015+01:00","day_of_week":5,"day_of_year":344,"dst":false,"dst_from":null,"dst_offset":0,"dst_until":null,"raw_offset":3600,"timezone":"Europe/Rome"},"unixtime":1639093969,"utc_datetime":"2021-12-09T23:52:49.256015+00:00","utc_offset":"+01:00","week_number":49}}
root@ubuntu-5895b8b77-449fp:~# curl cloud-step3-service:6000
{"hostname":"amiroo@Raei","time":{"abbreviation":"CET","client_ip":"8.20.127.32","datetime":"2021-12-10T00:52:50.545561+01:00","day_of_week":5,"day_of_year":344,"dst":false,"dst_from":null,"dst_offset":0,"dst_until":null,"raw_offset":3600,"timezone":"Europe/Rome"},"unixtime":1639093970,"utc_datetime":"2021-12-09T23:52:50.545561+00:00","utc_offset":"+01:00","week_number":49}}
root@ubuntu-5895b8b77-449fp:~# curl cloud-step3-service:6000
{"hostname":"amiroo@Raei","time":{"abbreviation":"CET","client_ip":"8.20.127.32","datetime":"2021-12-10T01:00:04.160637+01:00","day_of_week":5,"day_of_year":344,"dst":false,"dst_from":null,"dst_offset":0,"dst_until":null,"raw_offset":3600,"timezone":"Europe/Rome"},"unixtime":1639094405,"utc_datetime":"2021-12-10T00:00:04.160637+00:00","utc_offset":"+01:00","week_number":49}}
root@ubuntu-5895b8b77-449fp:~#
```

۳-

برای اجرا ایمج گام اول کافیتست که تنها دستور `kubectl exec <pod-name> -it -- bash` را اجرا کنیم که در اینجا pod name با `ubuntu-5895b8b77-449fp` است.