Konteynerler, Docker ve IBM Cloud Container Registry'ye Giriş

Amaçlar

Bu laboratuvar çalışmasında:

- Docker Hub'dan bir imaj çekeceksiniz
- Bir imajı `docker` kullanarak konteyner olarak çalıştıracaksınız
- Bir Dockerfile kullanarak bir imaj oluşturacaksınız
- Bir imajı IBM Cloud Container Registry'ye göndereceksiniz

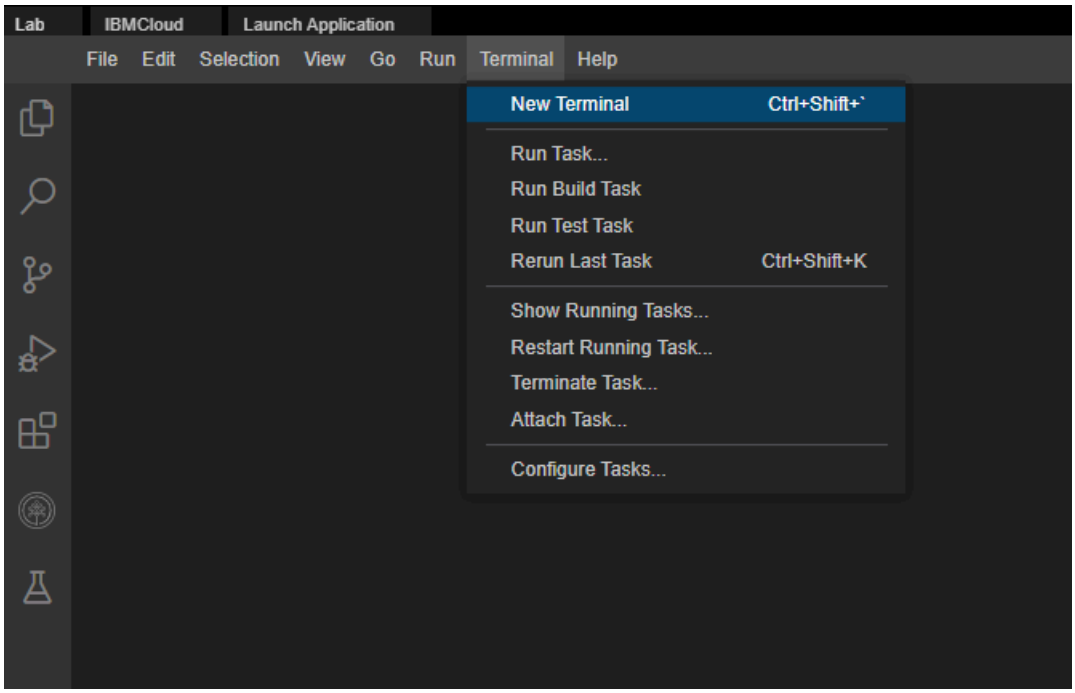
> **Not: Lütfen laboratuvarı tek bir oturumda, ara vermeden tamamlayın çünkü laboratuvar çevrimdışı moda geçebilir ve hatalara neden olabilir. Laboratuvar sürecinde herhangi bir sorun/hata ile karşılaşırsanız, laboratuvar ortamından çıkış yapın. Ardından sistem önbelleğinizi ve çerezlerinizi temizleyin ve laboratuvarı tamamlamaya çalışın.**

Önemli:
Zaten bir IBM Cloud hesabınız olabilir ve hatta IBM Container Registry'de (ICR) bir ad alanınız da olabilir. Ancak, bu laboratuvarda **kendi IBM Cloud hesabınızı veya kendi ICR ad alanınızı kullanmayacaksınız**. Bu alıştırma için sizin adınıza otomatik olarak oluşturulmuş bir IBM Cloud hesabı kullanacaksınız. Laboratuvar ortamı, kişisel IBM Cloud hesabınızdaki herhangi bir kaynağa, ICR ad alanlarına ve imajlara erişim sağlamayacaktır.

Ortamı ve komut satırı araçlarını doğrulayın

1. Editördeki menüyü kullanarak bir terminal penceresi açın: `Terminal > New Terminal`.

   > **Not: Terminal zaten açıksa, lütfen bu adımı atlayın.**



2. `docker` CLI'nin kurulu olduğunu doğrulayın.

   ```
   docker --version
   ```

Aşağıdaki çıktıyı görmelisiniz, ancak sürüm farklı olabilir:

```
theia@theiadocker-manvig1:/home/project$ docker --version
Docker version 26.1.3, build b72abbb
theia@theiadocker-manvig1:/home/project$
```

3. `ibmcloud` CLI'nin yüklü olduğunu doğrulayın.

   ```
   ibmcloud version
   ```

Aşağıdaki çıktıyı görmelisiniz, ancak sürüm farklı olabilir:

```
theia@theiadocker-manvig1:/home/project$ ibmcloud version
ibmcloud version 2.20.0+f382323-2023-09-19T20:06:39+00:00
```

4. Proje klasörünüze geçin.

   Not: Eğer zaten '/home/project' klasöründeyseniz, bu adımı atlayabilirsiniz.

   ```
   cd /home/project
   ```

5. Bu laboratuvar için gerekli olan eserleri içeren git deposunu, henüz yoksa klonlayın.

   ```
   [ ! -d 'CC201' ] && git clone https://github.com/ibm-developer-skills-network/CC201.git
   ```

```
theia@theiadocker-manvig1:/home/project$ [ ! -d 'CC201' ] && git clone https://github.com/ibm-de
veloper-skills-network/CC201.git
Cloning into 'CC201'...
remote: Enumerating objects: 30, done.
remote: Counting objects: 100% (7/7), done.
remote: Compressing objects: 100% (7/7), done.
remote: Total 30 (delta 4), reused 0 (delta 0), pack-reused 23
Receiving objects: 100% (30/30), 8.67 KiB | 8.67 MiB/s, done.
Resolving deltas: 100% (13/13), done.
```

6. Bu laboratuvar için dizine geçmek üzere aşağıdaki komutu çalıştırın. `cd`, çalışma/geçerli dizini belirtilen isimdeki dizine değiştirecektir, bu durumda **CC201/labs/1_ContainersAndDcoker**.

   ```
   cd CC201/labs/1_ContainersAndDocker/
   ```

7. Bu dizinin içeriğini listeleyin ve bu laboratuvar için olan artefaktları görün.

   ```
   ls
   ```

```
theia@theiadocker-[redacted]:/home/project/CC201/labs/1_ContainersAndDocker$ ls
app.js  Dockerfile  package.json
```

Docker Hub'dan bir görüntü çekin ve bir konteyner olarak çalıştırın

1. Görüntülerinizi listelemek için `docker` CLI'sını kullanın.

   ```
   docker images
   ```

Henüz hiçbir resminiz olmadığı için (sadece başlıklar ile) boş bir tablo görmelisiniz.

```
theia@theiadocker-[redacted]:/home/project/CC201/labs/1_ContainersAndDocker$ docker images
REPOSITORY   TAG       IMAGE ID   CREATED   SIZE
```

2. İlk resminizi Docker Hub'dan çekin.

```
docker pull hello-world
```

```
theia@theiadocker-[redacted]:/home/project/CC201/labs/1_ContainersAndDocker$ docker pull hello-world
Using default tag: latest
latest: Pulling from library/hello-world
2db29710123e: Pull complete
Digest: sha256:bfea6278a0a267fad2634554f4f0c6f31981eea41c553fdf5a83e95a41d40c38
Status: Downloaded newer image for hello-world:latest
docker.io/library/hello-world:latest
```

3. Görüntüleri tekrar listeleyin.

```
docker images
```

Artık tabloda `hello-world` görüntüsünü görmelisiniz.

```
theia@theiadocker-[redacted]:/home/project/CC201/labs/1_ContainersAndDocker$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED        SIZE
hello-world   latest    feb5d9fea6a5   6 months ago   13.3kB
```

4. `hello-world` görüntüsünü bir konteyner olarak çalıştırın.

```
docker run hello-world
```

'Docker'dan Merhaba!' mesajını görmelisiniz.

Bu mesajı oluşturmak için Docker'ın ne yaptığını açıklayan bir bilgi de olacak.

```
theia@theiadocker-[redacted]:/home/project/CC201/labs/1_ContainersAndDocker$ docker run hello-world

Hello from Docker!
This message shows that your installation appears to be working correctly.

To generate this message, Docker took the following steps:
 1. The Docker client contacted the Docker daemon.
 2. The Docker daemon pulled the "hello-world" image from the Docker Hub.
    (amd64)
 3. The Docker daemon created a new container from that image which runs the
    executable that produces the output you are currently reading.
 4. The Docker daemon streamed that output to the Docker client, which sent it
    to your terminal.

To try something more ambitious, you can run an Ubuntu container with:
 $ docker run -it ubuntu bash

Share images, automate workflows, and more with a free Docker ID:
 https://hub.docker.com/

For more examples and ideas, visit:
 https://docs.docker.com/get-started/
```

5. Konteynerleri listeleyin, böylece konteynerinizin başarıyla çalıştığını ve kapandığını görebilirsiniz.

```
docker ps -a
```

Bu konteyner için, diğer şeylerin yanı sıra bir konteyner kimliği, görüntü adı (`hello-world`) ve konteynerin başarılı bir şekilde kapandığını belirten bir durum görmelisiniz.

```
theia@theiadocker-manvig1:/home/project/CC201/labs/1_ContainersAndDocker$ docker ps -a
CONTAINER ID   IMAGE         COMMAND    CREATED          STATUS                      PORTS     NAMES
3e0a94c8908f   hello-world   "/hello"   50 seconds ago   Exited (0) 49 seconds ago             friendly_davinci
```

6. Önceki çıktıdan CONTAINER ID'yi not edin ve aşağıdaki komuttaki **<container_id>** etiketini bu değerle değiştirin. Bu komut konteynerinizi kaldırır.

```
docker container rm <container_id>
```

```
theia@theiadocker-            :/home/project/CC201/labs/1_ContainersAndDocker$ docker container rm 5e1756c09910
5e1756c09910
```

7. Konteynerin kaldırıldığını doğrulayın. Aşağıdaki komutu çalıştırın.

```
docker ps -a
```

```
theia@theiadocker-            :/home/project/CC201/labs/1_ContainersAndDocker$ docker ps -a
CONTAINER ID   IMAGE     COMMAND   CREATED   STATUS    PORTS     NAMES
```

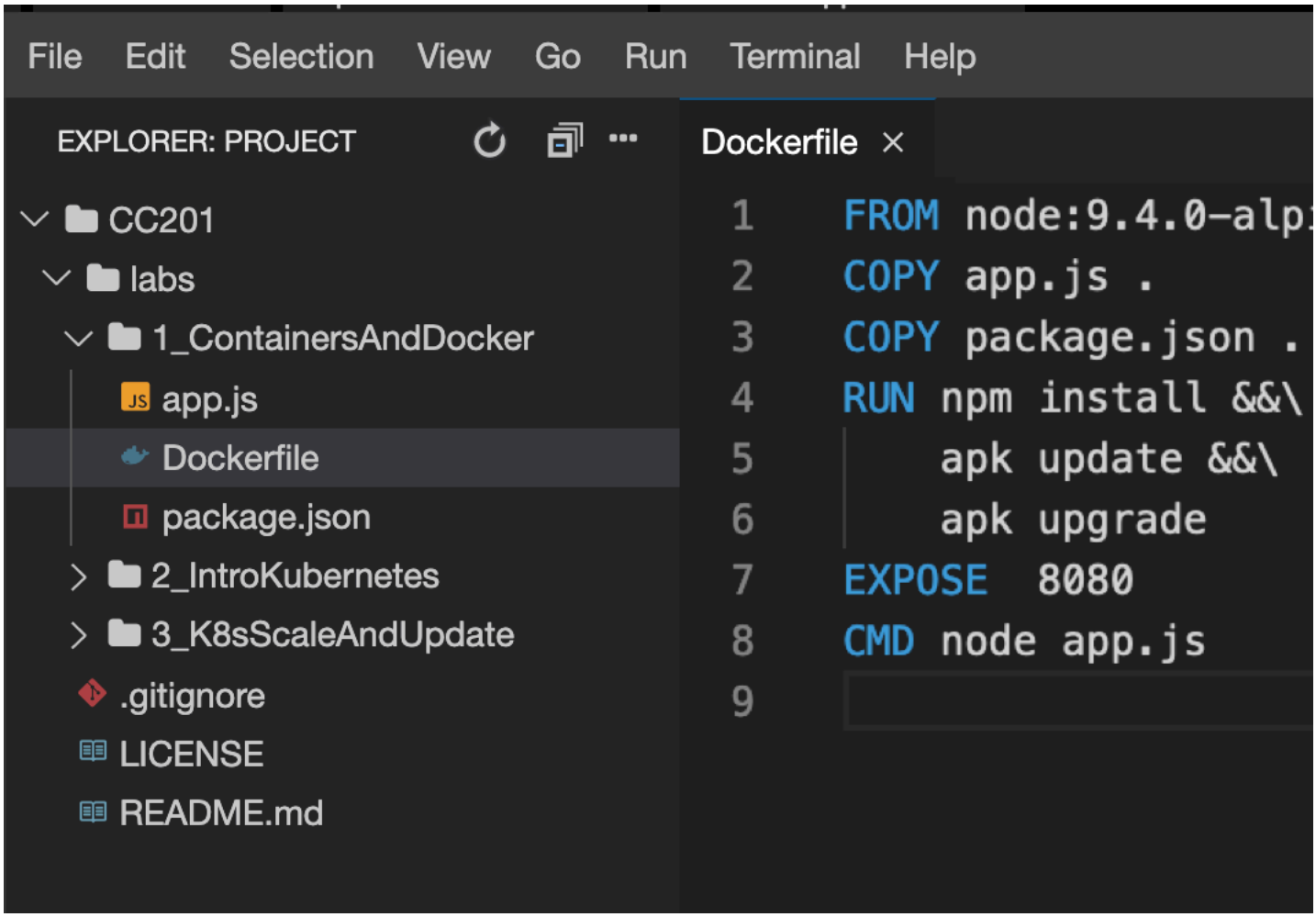
Docker Hub'dan bir imaj çektiğiniz ve ilk konteynerinizi çalıştırdığınız için tebrikler! Şimdi kendi imajımızı oluşturmaya çalışalım.

Bir Dockerfile Kullanarak Bir Görüntü Oluşturun

1. Mevcut çalışma dizini, bir konteynerde çalıştıracağımız basit bir Node.js uygulamasını içeriyor. Uygulama, bir merhaba mesajı ile birlikte ana bilgisayar adını yazdıracak. Uygulamayı bir konteynerde çalıştırmak için aşağıdaki dosyalar gereklidir:

- app.js, basitçe bir merhaba dünya mesajı ile yanıt veren ana uygulamadır.
- package.json, uygulamanın bağımlılıklarını tanımlar.
- Dockerfile, Docker'ın görüntüyü oluşturmak için kullandığı talimatları tanımlar.

2. Bu uygulama için gerekli dosyaları görüntülemek için Gezgini kullanın. Pencerenin sol tarafındaki Gezgini simgesine (bir kağıt yaprağı gibi görünür) tıklayın ve ardından bu laboratuvarın dizinine gidin: `CC201 > labs > 1_ContainersAndDocker`. Görüntüyü oluşturmak için gereken komutları görüntülemek için `Dockerfile`'a tıklayın.



Aşağıda Dockerfile'da belirtilen komutları yeniden gözden geçirebilirsiniz:

FROM talimatı, yeni bir yapı aşamasını başlatır ve sonraki talimatların üzerine inşa edileceği temel görüntüyü belirtir.

COPY komutu, dosyaları görüntümüze kopyalamamıza olanak tanır.

RUN talimatı, komutları çalıştırır.

EXPOSE talimatı, belirli bir portu belirli bir protokolle bir Docker Konteyneri içinde açar.

CMD talimatı, bir konteyneri çalıştırmak için varsayılan bir değer sağlar veya diğer bir deyişle, konteynerinizde çalıştırılması gereken bir yürütülebilir dosyadır.

3. Görüntüyü oluşturmak için aşağıdaki komutu çalıştırın:

```
docker build . -t myimage:v1
```

Modül videolarında görüldüğü gibi, çıktı her bir talimat için Dockerfile'da yeni bir katman oluşturur.

```
theia@theiadocker-manvig1:/home/project/CC201/labs/1_ContainersAndDocker$ docker build . -t myi
age:v1
[+] Building 10.7s (10/10) FINISHED                                             docker:default
 => [internal] load build definition from Dockerfile                                      0.0s
 => => transferring dockerfile: 180B                                                      0.0s
 => [internal] load .dockerignore                                                         0.0s
 => => transferring context: 2B                                                           0.0s
 => [internal] load metadata for docker.io/library/node:9.4.0-alpine                      0.4s
 => [auth] library/node:pull token for registry-1.docker.io                               0.0s
 => [1/4] FROM docker.io/library/node:9.4.0-alpine@sha256:9cd67a00ed111285460a83847720132  5.2s
 => => resolve docker.io/library/node:9.4.0-alpine@sha256:9cd67a00ed111285460a83847720132  0.0s
 => => sha256:b5f94997f35f4d1ba6221656d90dbe1d9f0ce6e3bc7a41a890b4627e13a 4.94kB / 4.94kB  0.0s
 => => sha256:605ce1bd3f3164f2949a30501cc596f52a72de05da1306ab360055f0d71 1.99MB / 1.99MB  0.2s
 => => sha256:fe58b30348fe37cda551e7f3a63375c46977493a48b98f70f708747ab 19.70MB / 19.70MB  1.3s
 => => sha256:46ef8987ccbdd5d2e0127b7eccca7b618fd9b17f6abb5c178ef5008de5c 1.02MB / 1.02MB  0.3s
 => => sha256:9cd67a00ed111285460a83847720132204185e9321ec35dacec0d8b9bf6 1.39kB / 1.39kB  0.0s
 => => sha256:359a2efa481b9edeff9ca120128f89387ce13dafe30b05f762ec63c7f770e41 951B / 951B  0.0s
 => => extracting sha256:605ce1bd3f3164f2949a30501cc596f52a72de05da1306ab360055f0d7130c32  0.2s
 => => extracting sha256:fe58b30348fe37cda551e7f3a63375c46977493a48b98f70f708747ab0b2a282  2.1s
 => => extracting sha256:46ef8987ccbdd5d2e0127b7eccca7b618fd9b17f6abb5c178ef5008de5cae4ef  0.2s
 => [internal] load build context                                                         0.0s
 => => transferring context: 573B                                                         0.0s
 => [2/4] COPY app.js .                                                                   0.0s
 => [3/4] COPY package.json .                                                             0.0s
 => [4/4] RUN npm install &&    apk update &&    apk upgrade                              3.8s
 => exporting to image                                                                    1.1s
 => => exporting layers                                                                   1.1s
 => => writing image sha256:6f46ce33aecfb71942f0a6056a93be8d4d489bab3dc949accfd464021cf17  0.0s
 => => naming to docker.io/library/myimage:v1                                             0.0s
```

4. Tabloyu görmek için görüntüleri listeleyin ve `myimage:v1` etiketli görüntünüzü kontrol edin.

   ```
   docker images
   ```

```
theia@theiadocker-manvig1:/home/project/CC201/labs/1_ContainersAndDocker$ docker images
REPOSITORY    TAG       IMAGE ID       CREATED          SIZE
myimage       v1        6f46ce33aecf   48 seconds ago   77.5MB
hello-world   latest    d2c94e258dcb   13 months ago    13.3kB
```

`hello-world` imajına kıyasla, bu imajın farklı bir imaj kimliği olduğunu unutmayın. Bu, iki imajın farklı katmanlardan oluştuğu anlamına gelir - diğer bir deyişle, aynı imaj değiller.

Görüntüyü bir konteyner olarak çalıştır

1. Artık görüntünüz oluşturuldu, aşağıdaki komutla bir konteyner olarak çalıştırın:

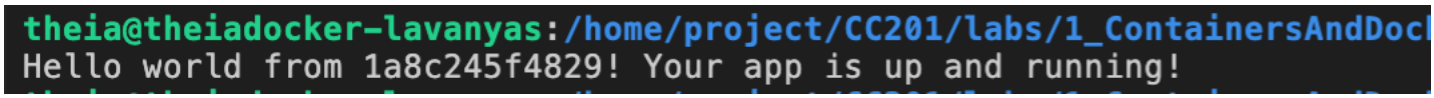
   ```
   docker run -dp 8080:8080 myimage:v1
   ```

```
theia@theiadocker-lavanyas:/home/project/CC201/labs/1_ContainersAndDock
1a8c245f482950cba52bcdb72686a8435e6c8916c6446434da55f5faac2372f3
theia@theiadocker-lavanyas:/home/project/CC201/labs/1_ContainersAndDock
```

Çıktı, çalıştırdığınız uygulama için docker tarafından tahsis edilen benzersiz bir koddur.

2. Aşağıda verilen şekilde uygulamayı pinglemek için `curl` komutunu çalıştırın.

   ```
   curl localhost:8080
   ```



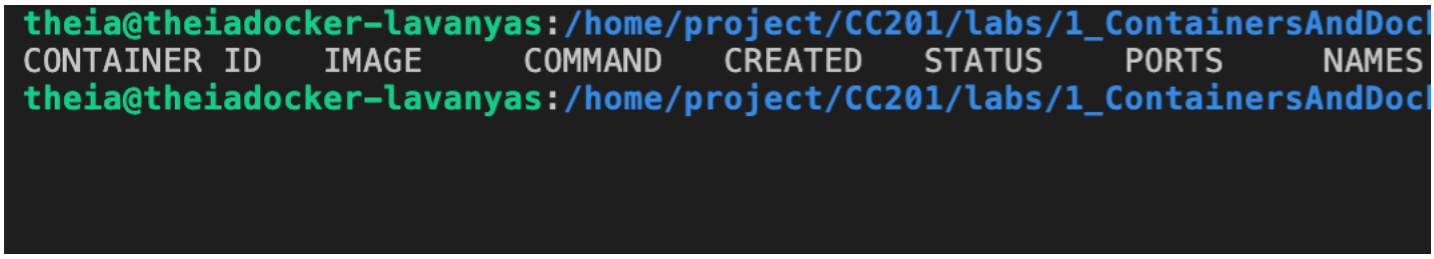
Eğer yukarıdaki gibi bir çıktı görüyorsanız, bu **'Uygulamanız çalışıyor!'** anlamına gelir.

3. Şimdi konteyneri durdurmak için `docker stop` komutunu kullanıyoruz ve ardından konteyner kimliğini ekliyoruz. Aşağıdaki komut, çalışan tüm konteynerlerin listesini geçirmek için `docker ps -q` kullanır:

   ```
   docker stop $(docker ps -q)
   ```

4. Konteynerin durup durmadığını kontrol etmek için aşağıdaki komutu çalıştırın.

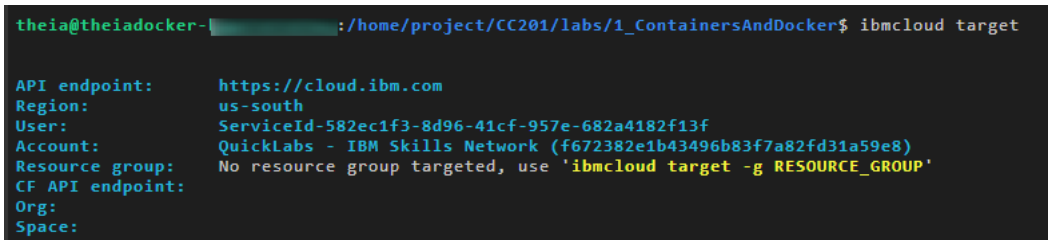
   ```
   docker ps
   ```



Görüntüyü IBM Cloud Container Registry'ye Yükle

1. Ortam, Skills Network Labs ortamı tarafından sizin için otomatik olarak oluşturulmuş IBM Cloud hesabınıza zaten giriş yapmış olmalıdır. Aşağıdaki komut, hedeflediğiniz hesap hakkında bilgi verecektir:

   ```
   ibmcloud target
   ```



2. Ortam ayrıca sizin için bir IBM Cloud Container Registry (ICR) ad alanı oluşturdu. Container Registry çoklu kiracı desteklediğinden, ad alanları kaydı birden fazla kullanıcı arasında bölmek için kullanılır. Erişim izniniz olan ad alanlarını görmek için aşağıdaki komutu kullanın:

   ```
   ibmcloud cr namespaces
   ```

```
theia@theiadocker-[redacted]:/home/project/CC201/labs/1_ContainersAndDocker$ ibmcloud cr namespaces
Listing namespaces for account 'QuickLabs - IBM Skills Network' in registry 'us.icr.io'...

Namespace
sn-labs-[redacted]
sn-labsassets

OK
theia@theiadocker-[redacted]:/home/project/CC201/labs/1_ContainersAndDocker$
```

`sn-labs` ile başlayan iki ad alanı görmelisiniz:

- İlk olarak, kullanıcı adınızla birlikte olan ad alanı sadece sizin için. Bu ad alanına tam *okuma* ve *yazma* erişiminiz var.
- İkinci ad alanı, paylaşılan bir ad alanıdır ve size yalnızca Okuma Erişimi sağlar.

3. Bulut hesabınıza uygun olan bölgeyi hedeflediğinizden emin olun, örneğin bu ad alanlarının bulunduğu `us-south` bölgesi gibi, `ibmcloud target` komutunun çıktısında gördüğünüz gibi.

   ```
   ibmcloud cr region-set us-south
   ```

```
theia@theiadocker-[redacted]:/home/project/CC201/labs/1_ContainersAndDocker$ ibmcloud cr region-set us-south
The region is set to 'us-south', the registry is 'us.icr.io'.

OK
```

4. Yerel Docker daemon'unuzu IBM Cloud Container Registry'ye giriş yapın, böylece kayıt defterine yükleme ve indirme yapabilirsiniz.

   ```
   ibmcloud cr login
   ```

```
theia@theiadocker-manvig1:/home/project/CC201/labs/1_ContainersAndDocker$ ibmcloud cr login
Logging 'docker' in to 'us.icr.io'...
Logged in to 'us.icr.io'.

OK
```

5. İsim alanınızı sonraki komutlarda kullanılabilmesi için bir ortam değişkeni olarak dışa aktarın.

   ```
   export MY_NAMESPACE=sn-labs-$USERNAME
   ```

```
theia@theiadocker-[redacted]:/home/project/CC201/labs/1_ContainersAndDocker$ export MY_NAMESPACE=sn-labs-$USERNAME
```

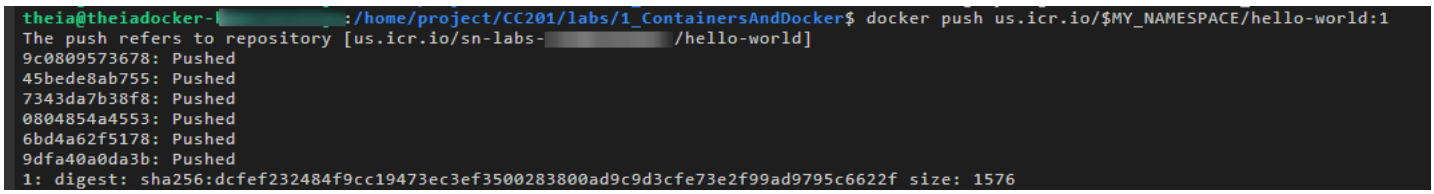
6. Görüntünüzü IBM Cloud Container Registry'ye itilebilmesi için etiketleyin.

   ```
   docker tag myimage:v1 us.icr.io/$MY_NAMESPACE/hello-world:1
   ```

```
theia@theiadocker-[redacted]:/home/project/CC201/labs/1_ContainersAndDocker$ docker push us.icr.io/$MY_NAMESPACE/hello-world:1
```

7. Yeni etiketlenmiş görüntüyü IBM Cloud Container Registry'ye gönderin.

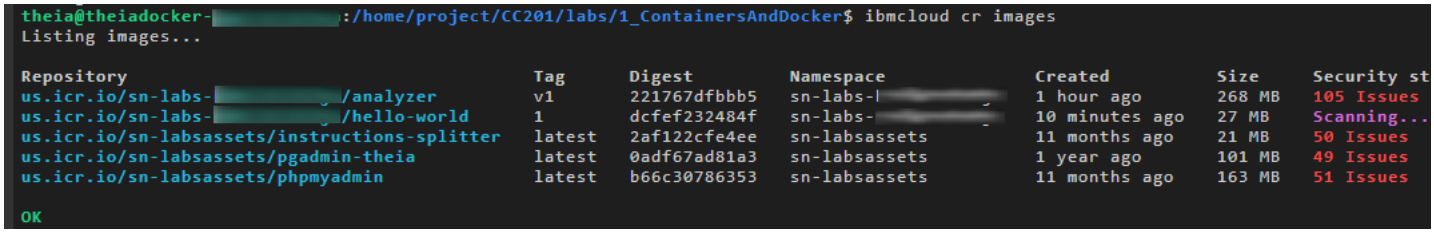
   ```
   docker push us.icr.io/$MY_NAMESPACE/hello-world:1
   ```



Not: Eğer bu laboratuvarı daha önce denediyseniz, önceki oturumun hala devam ediyor olma ihtimali vardır. Bu durumda, yukarıdaki çıktıda **'Pushed'** mesajı yerine **'Layer already Exists'** mesajını göreceksiniz. Laboratuvarın sonraki adımlarına geçmenizi öneririz.

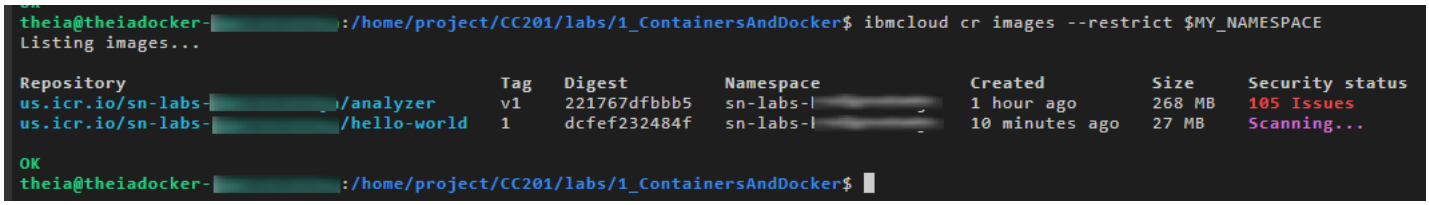
8. Görüntünün başarıyla yüklendiğini doğrulamak için Container Registry'deki görüntüleri listeleyin.

```
ibmcloud cr images
```



İsteğe bağlı olarak, yalnızca belirli bir ad alanındaki görüntüleri görüntülemek için.

```
ibmcloud cr images --restrict $MY_NAMESPACE
```



Çıktıda resim adınızı görmelisiniz.

Tebrikler! Bu kursun birinci modülündeki ikinci laboratuvarı tamamladınız.