به نام خدا

عنوان : اجرای application در داکر

انجام دهنده : زهرا منصوری

شماره دانشجویی : ۹۶۱۵۰۹۱۰۴۸

استاد : استاد فروغی

فهرست

مقدمه

داکر یک پلتفرم متن باز است که بر مبنای سیستم عامل لینوکس راه‌اندازی شده و ابزاری است که می‌تواند فرایند ایجاد، پیاده سازی و اجرای برنامه‌ها را با استفاده از Container ها بسیار ساده کند.

در مقابل ماشین‌های مجازی، Container ها قرار دارند. آن‌ها می‌توانند جایگزین مناسبی برای ماشین‌های مجازی باشند. Container ها محیط‌های اجرایی را جدا کرده و هسته سیستم عامل را به اشتراک می‌گذارند . Container ها نسبت به ماشین‌های مجازی از منابع کمتری استفاده می‌کنند.

Container این امکان را برای توسعه دهندگان فراهم می‌کند تا بسته کاملی از برنامه‌های خود همراه تمامی بخش‌های مورد نیاز آن ایجاد کرده و آن را در قالب یک بسته واحد ارسال کنند.

برای ساخت یک برنامه داکر و همچنین کار با داکر باید از کامپوننت‌های مختلف استفاده کنیم. در ادامه این کامپوننت‌ها را در قالب اجرای یک application به زبان python در داکر معرفی و بررسی می‌کنیم.

توضیح برنامه

برنامه زیر به زبان پایتون نوشته شده و آمار مبتلاین و فوت شدگان بر اثر ویروس کرونا را به تفکیک هر کشور گرفته و در جدول نمایش می دهد. کتابخانه های استفاده شده در این برنامه عبارتند از : requests , texttable , beautifulsoup4 .

در این برنامه ابتدا یک requests.get به آدرس :

“https://www.worldometers.info/coronavirus/countries-where-coronavirus-has-spread / “

می زنیم و از فایل html دریافت شده اطلاعات نام کشور ، تعداد مبتلایان و فوت شدگان و قاره را به دست می آوریم و در نهایت آنها را در قالب یک جدول در خروجی نمایش می دهیم.

```
# By Zahra Mansouri
# importing modules
import requests
from bs4 import BeautifulSoup
import texttable as tt

# URL for scrapping data
url = 'https://www.worldometers.info/coronavirus/countries-where-coronavirus-has-spread/'

# get URL html
page = requests.get(url)
soup = BeautifulSoup(page.text, 'html.parser')

data = []

# soup.find_all('td') will scrape every
# element in the url's table
data_iterator = iter(soup.find_all('td'))

while True:
    try:
        country = next(data_iterator).text
        confirmed = next(data_iterator).text
        deaths = next(data_iterator).text
        continent = next(data_iterator).text

        data.append((
            country,
            confirmed,
            deaths,
            continent
        ))

    except StopIteration:
        break

table = tt.Texttable()
table.add_rows([(' Country ', ' Number of cases ', ' Deaths ', ' Continent ')] + data)
table.set_cols_align(["c", "c", "c", "c"])
print(table.draw())
~
~
~
~
~
"main.py" 40 lines, 968 characters
```

شکل ۱

Docker File

هر container داکر به وسیله یک فایل داکر شروع به کار می‌کند. Dockerfile ها در واقع فایل‌های تنظیمات داکر هستند که با استفاده از آن‌ها می‌توانیم به داکر بگوییم که یک container را چگونه بالا بیاورد و تنظیم کند.

در اولین خط از این فایل، base image را برای Docker مشخص می کنیم. چون برنامه ما به زبان python3.8 نوشته شده است پس لازم است آن را به صورت زیر تعریف کنیم :

```
FROM python:3.8
```

سپس فایل اصلی برنامه را اضافه می کنیم و چون فایل اصلی برنامه و DockerFile در یک directory قرار دارند از نقطه به عنوان نشان directory جاری استفاده می کنیم.

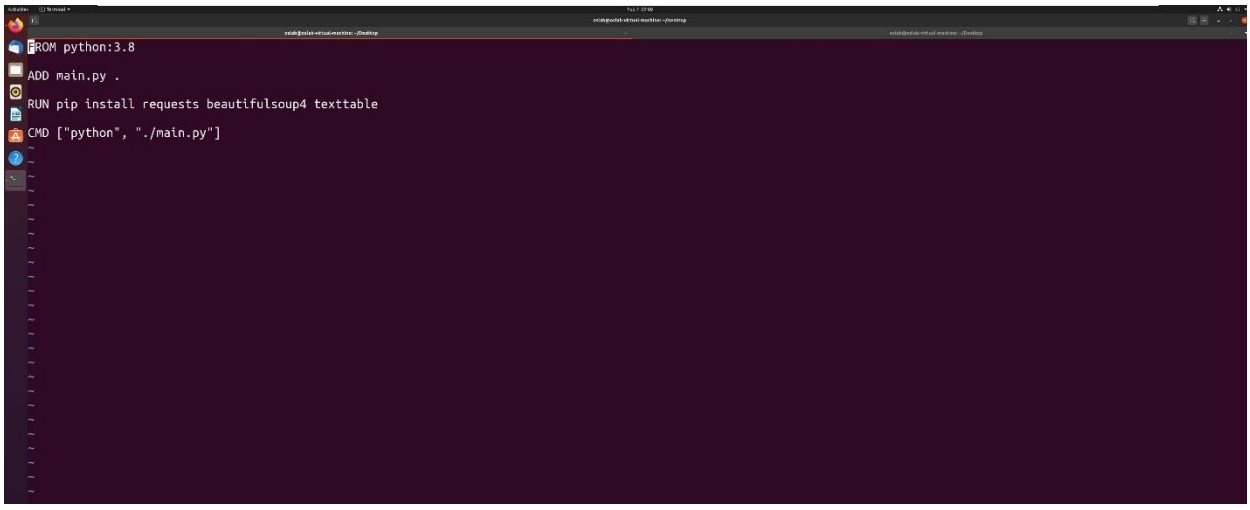
```
ADD main.py .
```

در مرحله بعد نیازمندی های برنامه را (شامل همه کتابخانه هایی که در برنامه استفاده شده است) با دستور RUN تعریف می کنیم. این دستور همه module های گفته شده را داخل image نصب می کند.

```
RUN pip install requests beautifulsoup4 texttable
```

با استفاده از دستور زیر نیز فرمان اجرایی را هنگام اجرای image تعریف می کنیم.

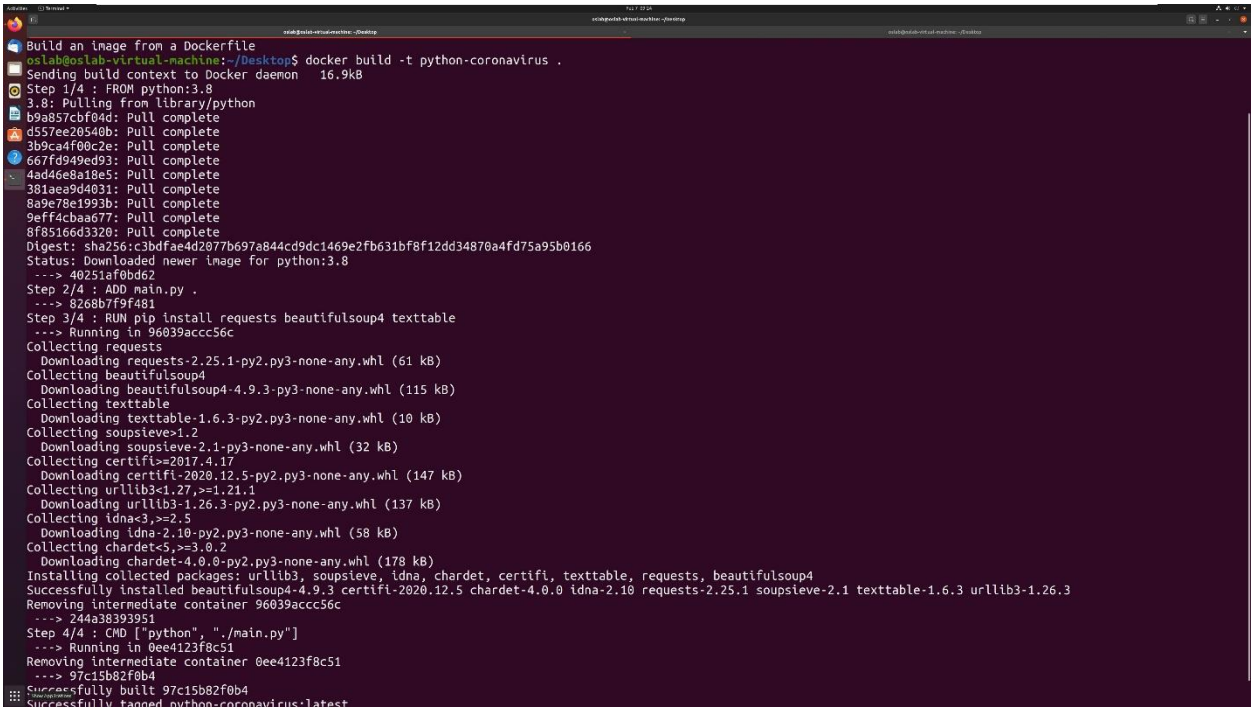
```
CMD [“python”,”./main.py”]
```



شکل ۲

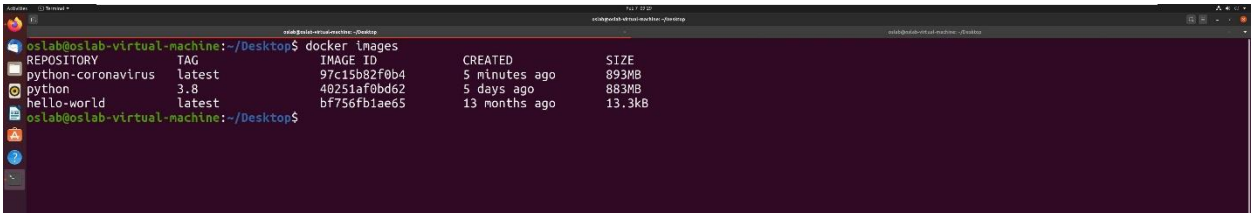
Build an image

پس از ساخت DockerFile با استفاده از دستور docker build ، یک image می سازیم . Image یک فایل قابل حمل و شامل یک سری دستورالعمل است که مشخص می‌کند Container کدام کامپوننت‌های نرم افزاری را اجرا کند و اینکه چطور آن را اجرا کند.



شکل ۳

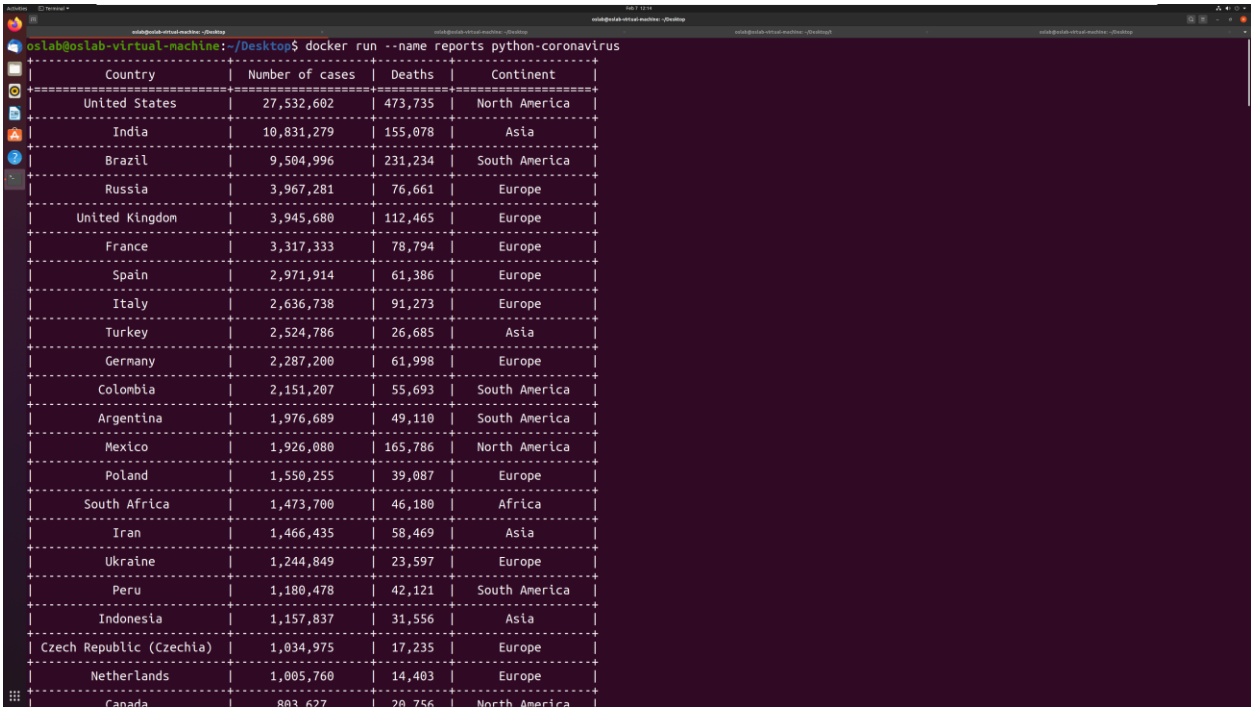
خروجی Docker images



شکل ٤

اجرای image

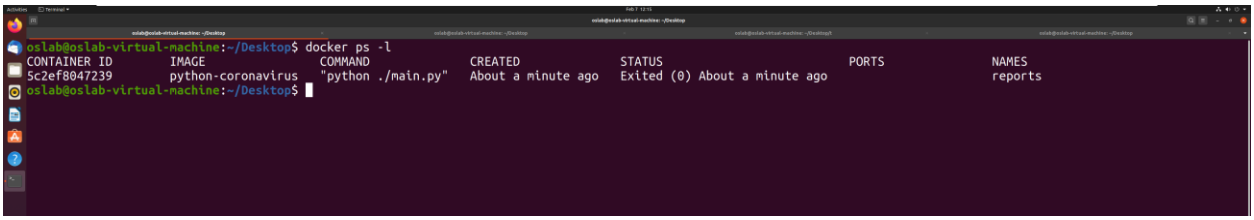
Docker run در واقع نوعی دستور است که container را راه‌اندازی می‌کند.



شکل ۵

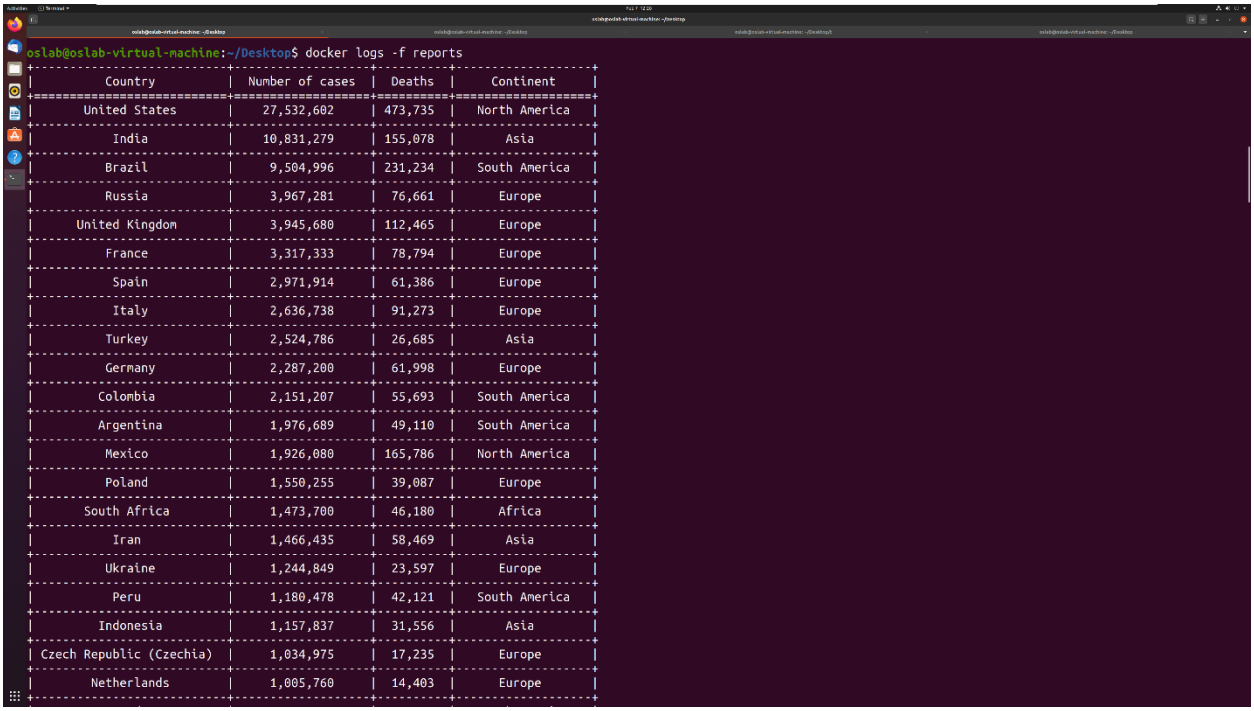
خروجی docker ps

این دستور لیست container های روی سرویس‌دهنده‌ی داکر شما را نمایش می‌دهد. با اضافه کردن l– به انتهای این دستور آخرین container راه‌اندازی شده در هر حالتی که باشد را به شما نمایش می‌دهد.



شکل ۶

خروجی logs –f



```
oslab@oslab-virtual-machine:~/Desktop$ docker logs -f reports
+--------------------------+--------------------+----------+--------------------+
|         Country          |  Number of cases   |  Deaths  |     Continent      |
+==========================+====================+==========+====================+
|       United States      |     27,532,602     | 473,735  |   North America    |
+--------------------------+--------------------+----------+--------------------+
|          India           |     10,831,279     | 155,078  |        Asia        |
+--------------------------+--------------------+----------+--------------------+
|          Brazil          |      9,504,996     | 231,234  |   South America    |
+--------------------------+--------------------+----------+--------------------+
|          Russia          |      3,967,281     |  76,661  |       Europe       |
+--------------------------+--------------------+----------+--------------------+
|      United Kingdom      |      3,945,680     | 112,465  |       Europe       |
+--------------------------+--------------------+----------+--------------------+
|          France          |      3,317,333     |  78,794  |       Europe       |
+--------------------------+--------------------+----------+--------------------+
|          Spain           |      2,971,914     |  61,386  |       Europe       |
+--------------------------+--------------------+----------+--------------------+
|          Italy           |      2,636,738     |  91,273  |       Europe       |
+--------------------------+--------------------+----------+--------------------+
|          Turkey          |      2,524,786     |  26,685  |        Asia        |
+--------------------------+--------------------+----------+--------------------+
|          Germany         |      2,287,200     |  61,998  |       Europe       |
+--------------------------+--------------------+----------+--------------------+
|         Colombia         |      2,151,207     |  55,693  |   South America    |
+--------------------------+--------------------+----------+--------------------+
|         Argentina        |      1,976,689     |  49,110  |   South America    |
+--------------------------+--------------------+----------+--------------------+
|          Mexico          |      1,926,080     | 165,786  |   North America    |
+--------------------------+--------------------+----------+--------------------+
|          Poland          |      1,550,255     |  39,087  |       Europe       |
+--------------------------+--------------------+----------+--------------------+
|       South Africa       |      1,473,700     |  46,180  |       Africa       |
+--------------------------+--------------------+----------+--------------------+
|          Iran            |      1,466,435     |  58,469  |        Asia        |
+--------------------------+--------------------+----------+--------------------+
|          Ukraine         |      1,244,849     |  23,597  |       Europe       |
+--------------------------+--------------------+----------+--------------------+
|          Peru            |      1,180,478     |  42,121  |   South America    |
+--------------------------+--------------------+----------+--------------------+
|         Indonesia        |      1,157,837     |  31,556  |        Asia        |
+--------------------------+--------------------+----------+--------------------+
| Czech Republic (Czechia) |      1,034,975     |  17,235  |       Europe       |
+--------------------------+--------------------+----------+--------------------+
|        Netherlands       |      1,005,760     |  14,403  |       Europe       |
+--------------------------+--------------------+----------+--------------------+
```

شکل ۷