Step two

In this step we wiil create a CryptoCurrency market price tracker. We will use redis as a database and python as a programming language.

also we will use docker to run redis and python-Django.

use redis cache to store data and use Django to get data from API and show it to user.

also use docker Volumes to store data in host machine.

also we nedd to create config file for redis annd Django.

config file contains :

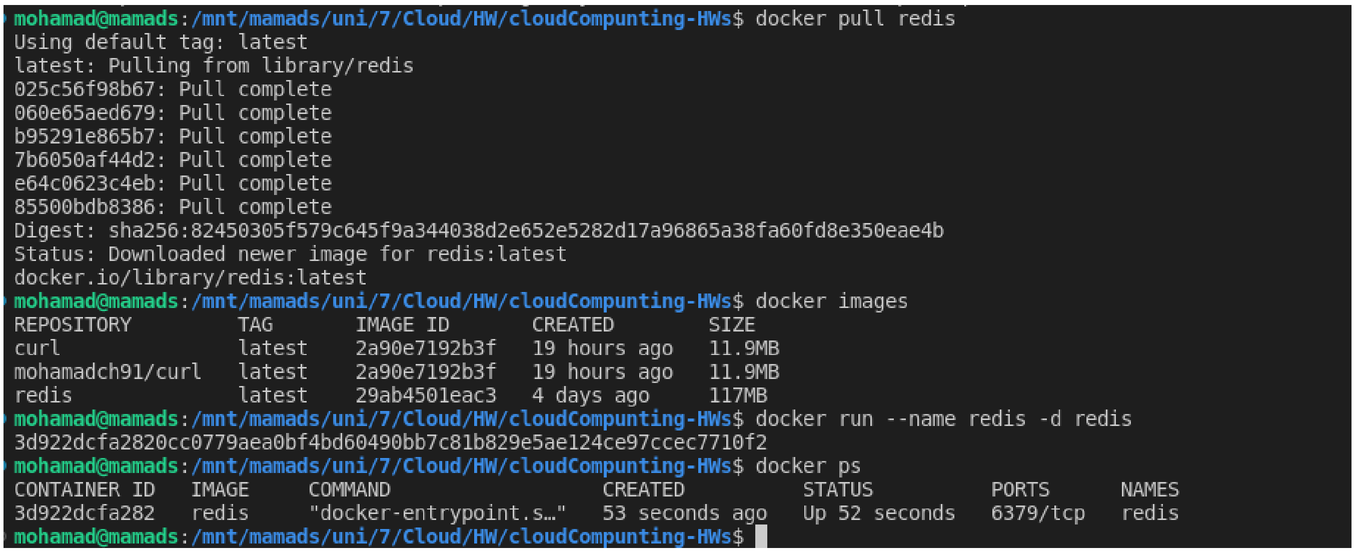
- Django server port
- redis keys expire time : default 5 minutes
- CoinAPI API key

2.1

First we need to pull redis image and make container from pulled image

```
docker pull redis
docker run --name redis_net -d redis
```

Result



```
mohamad@mamads:/mnt/mamads/uni/7/Cloud/HW/cloudCompunting-HWs$ docker pull redis
Using default tag: latest
latest: Pulling from library/redis
025c56f98b67: Pull complete
060e65aed679: Pull complete
b95291e865b7: Pull complete
7b6050af44d2: Pull complete
e64c0623c4eb: Pull complete
85500bdb8386: Pull complete
Digest: sha256:82450305f579c645f9a344038d2e652e5282d17a96865a38fa60fd8e350eae4b
Status: Downloaded newer image for redis:latest
docker.io/library/redis:latest
mohamad@mamads:/mnt/mamads/uni/7/Cloud/HW/cloudCompunting-HWs$ docker images
REPOSITORY         TAG       IMAGE ID       CREATED         SIZE
curl               latest    2a90e7192b3f   19 hours ago    11.9MB
mohamadch91/curl   latest    2a90e7192b3f   19 hours ago    11.9MB
redis              latest    29ab4501eac3   4 days ago      117MB
mohamad@mamads:/mnt/mamads/uni/7/Cloud/HW/cloudCompunting-HWs$ docker run --name redis -d redis
3d922dcfa2820cc0779aea0bf4bd60490bb7c81b829e5ae124ce97ccec7710f2
mohamad@mamads:/mnt/mamads/uni/7/Cloud/HW/cloudCompunting-HWs$ docker ps
CONTAINER ID   IMAGE     COMMAND                  CREATED          STATUS          PORTS      NAMES
3d922dcfa282   redis     "docker-entrypoint.s…"   53 seconds ago   Up 52 seconds   6379/tcp   redis
mohamad@mamads:/mnt/mamads/uni/7/Cloud/HW/cloudCompunting-HWs$
```

2.2

In this step we will create Django project and app and make image from it. then push this image to docker hub.

2.2.1

Create a Django project

```
django-admin startproject crypto
```

2.2.2

Create a Django app

```
cd crypto
python manage.py startapp cryptoApp
```

2.2.2.1

Create a class in Views.py file for get data from API and show it to user

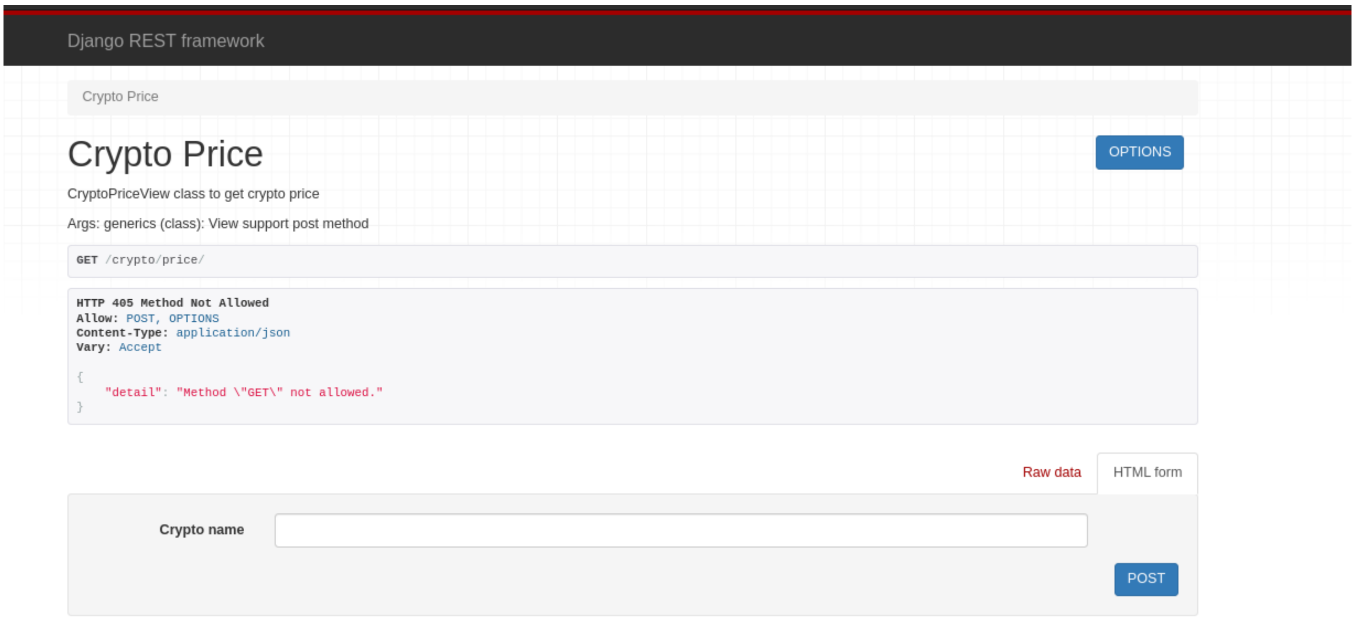
and create a urls.py file for routing

create Serializer class in serializers.py file for serialize data

change settings.py file for add app

and add urls.py file for routing

Results



Crypto Price

CryptoPriceView class to get crypto price

Args: generics (class): View support post method

```
GET /crypto/price/
```

```
HTTP 405 Method Not Allowed
Allow: POST, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "detail": "Method \"GET\" not allowed."
}
```

Crypto name

2.2.3

Create a requirements.txt

```
Django==4.1.4
django-cors-headers==3.13.0
djangorestframework==3.13.1
requests==2.26.0
requests-toolbelt==0.9.1
requests-unixsocket==0.2.0
redis==3.5.3
```

2.2.4

Create a Dockerfile in crypto directory and run Django server

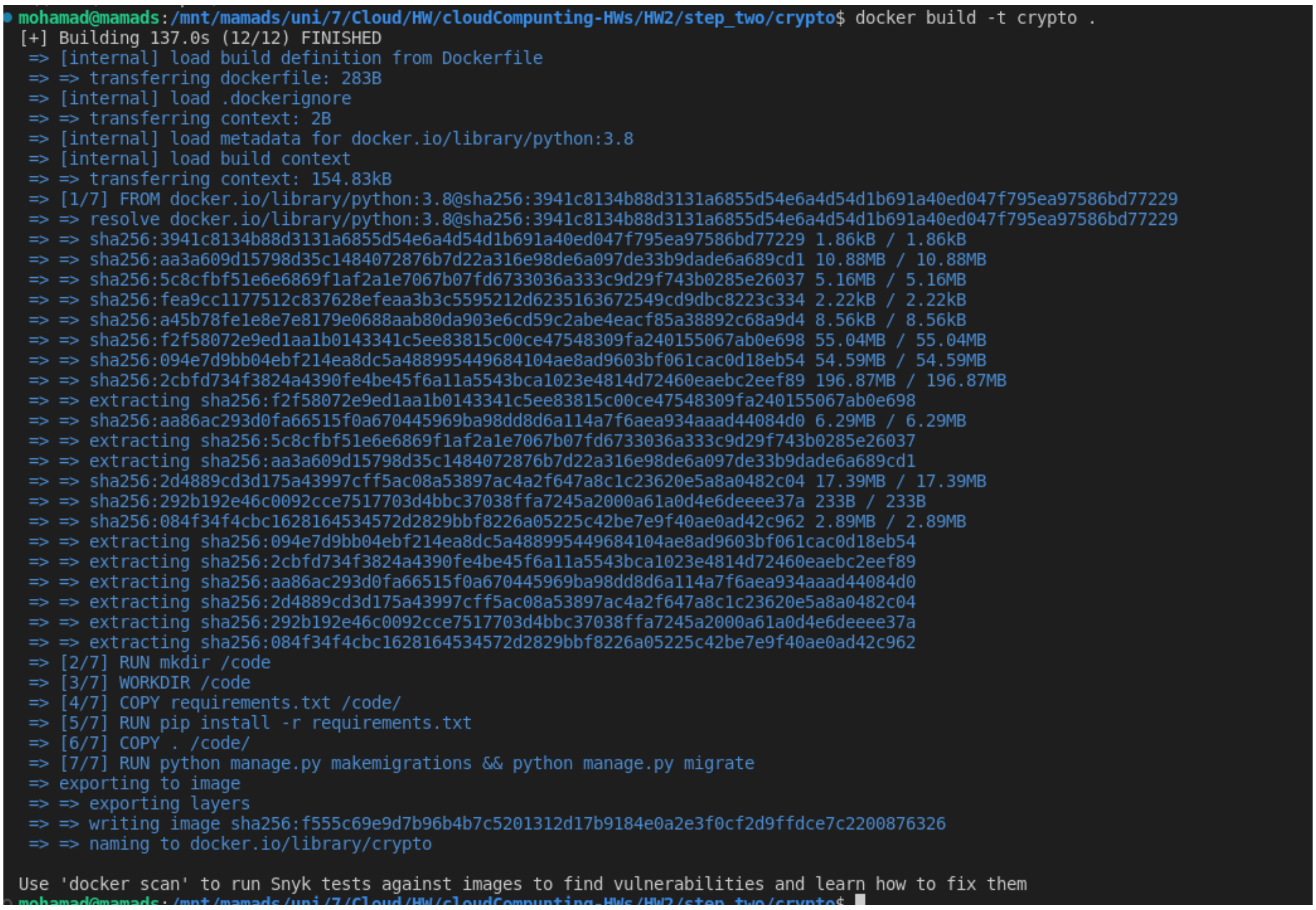
```
FROM python:3.8
ENV PYTHONUNBUFFERED 1
RUN mkdir /code
WORKDIR /code
COPY requirements.txt /code/
RUN pip install -r requirements.txt
COPY . /code/
CMD python manage.py runserver
```

2.2.5

Make image from Dockerfile

```
docker build -t crypto .
```

Result



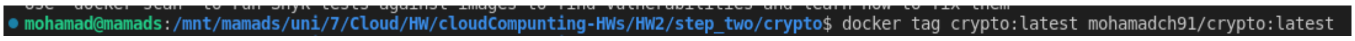
```
mohamad@mamads:/mnt/mamads/uni/7/Cloud/HW/cloudCompunting-HWs/HW2/step_two/crypto$ docker build -t crypto .
[+] Building 137.0s (12/12) FINISHED
 => [internal] load build definition from Dockerfile
 => => transferring dockerfile: 283B
 => [internal] load .dockerignore
 => => transferring context: 2B
 => [internal] load metadata for docker.io/library/python:3.8
 => [internal] load build context
 => => transferring context: 154.83kB
 => [1/7] FROM docker.io/library/python:3.8@sha256:3941c8134b88d3131a6855d54e6a4d54d1b691a40ed047f795ea97586bd77229
 => => resolve docker.io/library/python:3.8@sha256:3941c8134b88d3131a6855d54e6a4d54d1b691a40ed047f795ea97586bd77229
 => => sha256:3941c8134b88d3131a6855d54e6a4d54d1b691a40ed047f795ea97586bd77229 1.86kB / 1.86kB
 => => sha256:aa3a609d15798d35c1484072876b7d22a316e98de6a097de33b9dade6a689cd1 10.88MB / 10.88MB
 => => sha256:5c8cfbf51e6e6869f1af2a1e7067b07fd6733036a333c9d29f743b0285e26037 5.16MB / 5.16MB
 => => sha256:fea9cc1177512c837628efeaa3b3c5595212d6235163672549cd9dbc8223c334 2.22kB / 2.22kB
 => => sha256:a45b78fe1e8e7e8179e0688aab80da903e6cd59c2abe4eacf85a38892c68a9d4 8.56kB / 8.56kB
 => => sha256:f2f58072e9ed1aa1b0143341c5ee83815c00ce47548309fa240155067ab0e698 55.04MB / 55.04MB
 => => sha256:094e7d9bb04ebf214ea8dc5a488995449684104ae8ad9603bf061cac0d18eb54 54.59MB / 54.59MB
 => => sha256:2cbfd734f3824a4390fe4be45f6a11a5543bca1023e4814d72460eaebc2eef89 196.87MB / 196.87MB
 => => extracting sha256:f2f58072e9ed1aa1b0143341c5ee83815c00ce47548309fa240155067ab0e698
 => => sha256:aa86ac293d0fa66515f0a670445969ba98dd8d6a114a7f6aea934aaad44084d0 6.29MB / 6.29MB
 => => extracting sha256:5c8cfbf51e6e6869f1af2a1e7067b07fd6733036a333c9d29f743b0285e26037
 => => extracting sha256:aa3a609d15798d35c1484072876b7d22a316e98de6a097de33b9dade6a689cd1
 => => sha256:2d4889cd3d175a43997cff5ac08a53897ac4a2f647a8c1c23620e5a8a0482c04 17.39MB / 17.39MB
 => => sha256:292b192e46c0092cce7517703d4bbc37038ffa7245a2000a61a0d4e6deeee37a 233B / 233B
 => => sha256:084f34f4cbc1628164534572d2829bbf8226a05225c42be7e9f40ae0ad42c962 2.89MB / 2.89MB
 => => extracting sha256:094e7d9bb04ebf214ea8dc5a488995449684104ae8ad9603bf061cac0d18eb54
 => => extracting sha256:2cbfd734f3824a4390fe4be45f6a11a5543bca1023e4814d72460eaebc2eef89
 => => extracting sha256:aa86ac293d0fa66515f0a670445969ba98dd8d6a114a7f6aea934aaad44084d0
 => => extracting sha256:2d4889cd3d175a43997cff5ac08a53897ac4a2f647a8c1c23620e5a8a0482c04
 => => extracting sha256:292b192e46c0092cce7517703d4bbc37038ffa7245a2000a61a0d4e6deeee37a
 => => extracting sha256:084f34f4cbc1628164534572d2829bbf8226a05225c42be7e9f40ae0ad42c962
 => [2/7] RUN mkdir /code
 => [3/7] WORKDIR /code
 => [4/7] COPY requirements.txt /code/
 => [5/7] RUN pip install -r requirements.txt
 => [6/7] COPY . /code/
 => [7/7] RUN python manage.py makemigrations && python manage.py migrate
 => exporting to image
 => => exporting layers
 => => writing image sha256:f555c69e9d7b96b4b7c5201312d17b9184e0a2e3f0cf2d9ffdce7c2200876326
 => => naming to docker.io/library/crypto

Use 'docker scan' to run Snyk tests against images to find vulnerabilities and learn how to fix them
```

2.2.6

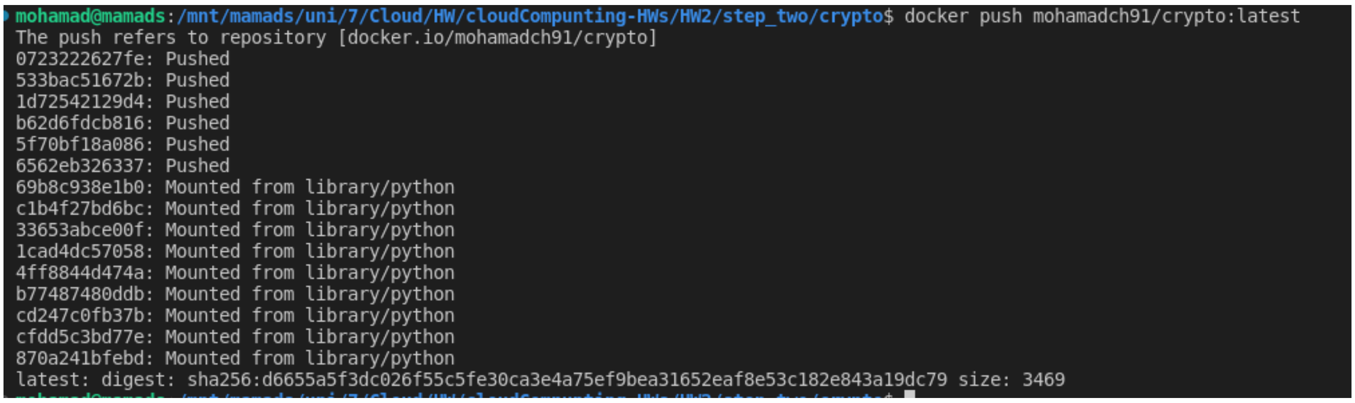
Push image to docker hub

```
docker tag crypto:latest mohamadch91/crypto:latest
docker push mohamadch91/crypto:latest
```

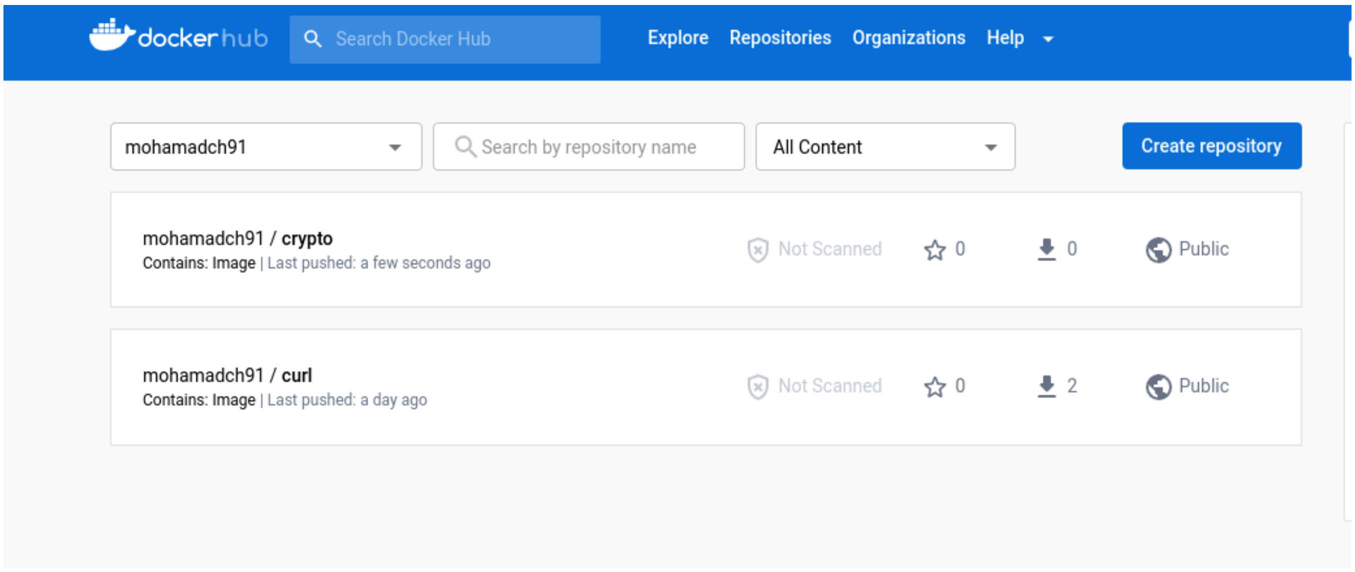
Result



```
mohamad@mamads:/mnt/mamads/uni/7/Cloud/HW/cloudCompunting-HWs/HW2/step_two/crypto$ docker tag crypto:latest mohamadch91/crypto:latest
```



```
mohamad@mamads:/mnt/mamads/uni/7/Cloud/HW/cloudCompunting-HWs/HW2/step_two/crypto$ docker push mohamadch91/crypto:latest
The push refers to repository [docker.io/mohamadch91/crypto]
0723222627fe: Pushed
533bac51672b: Pushed
1d72542129d4: Pushed
b62d6fdcb816: Pushed
5f70bf18a086: Pushed
6562eb326337: Pushed
69b8c938e1b0: Mounted from library/python
c1b4f27bd6bc: Mounted from library/python
33653abce00f: Mounted from library/python
1cad4dc57058: Mounted from library/python
4ff8844d474a: Mounted from library/python
b77487480ddb: Mounted from library/python
cd247c0fb37b: Mounted from library/python
cfdd5c3bd77e: Mounted from library/python
870a241bfebd: Mounted from library/python
latest: digest: sha256:d6655a5f3dc026f55c5fe30ca3e4a75ef9bea31652eaf8e53c182e843a19dc79 size: 3469
```



mohamadch91 / crypto
Contains: Image | Last pushed: a few seconds ago
Not Scanned 0 0 Public

mohamadch91 / curl
Contains: Image | Last pushed: a day ago
Not Scanned 0 2 Public

2.3

In this step we need to create config file for Django and redis

Create a .env file for Django and redis in Dockerfile

```
CACHE_TTL=300
REDIS_HOST=redis
REDIS_PORT=6379
DJANGO_PORT=8000
COINAPI_KEY=YOUR_API
```

2.3.1

Change setting.py file for env

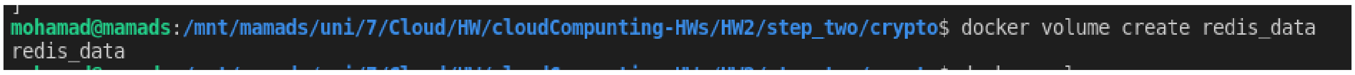
```
runserver.default_port = os.environ.get('DJANGO_PORT', '8001')

COINAPI_KEY = os.environ.get('COINAPI_KEY', 'Your key')

CACHE_TTL=os.environ.get('CACHE_TTL', '350')
```

2.4

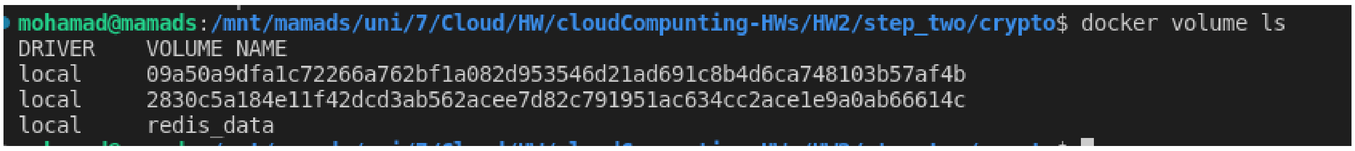
In this step we need to create volumes for redis

~~~
```bash
docker volume create redis_data
```
~~~

Result



```
mohamad@mamads:/mnt/mamads/uni/7/Cloud/HW/cloudCompunting-HWs/HW2/step_two/crypto$ docker volume create redis_data
redis_data
```



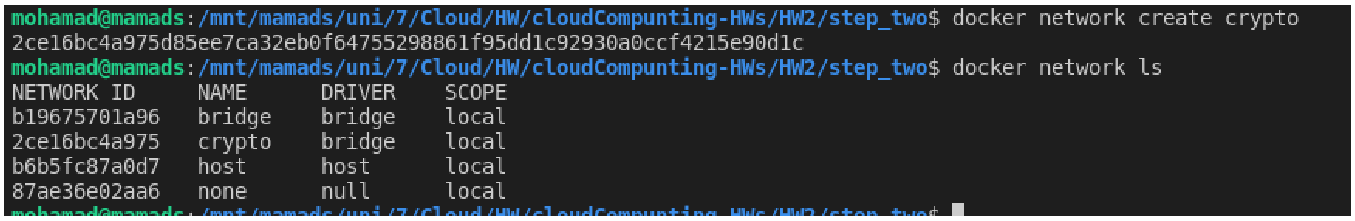
```
mohamad@mamads:/mnt/mamads/uni/7/Cloud/HW/cloudCompunting-HWs/HW2/step_two/crypto$ docker volume ls
DRIVER    VOLUME NAME
local     09a50a9dfa1c72266a762bf1a082d953546d21ad691c8b4d6ca748103b57af4b
local     2830c5a184e11f42dcd3ab562acee7d82c791951ac634cc2ace1e9a0ab66614c
local     redis_data
```

2.5

In this step we need to create network for redis and Django

```
docker network create crypto
```

Result



```
mohamad@mamads:/mnt/mamads/uni/7/Cloud/HW/cloudCompunting-HWs/HW2/step_two$ docker network create crypto
2ce16bc4a975d85ee7ca32eb0f64755298861f95dd1c92930a0ccf4215e90d1c
mohamad@mamads:/mnt/mamads/uni/7/Cloud/HW/cloudCompunting-HWs/HW2/step_two$ docker network ls
NETWORK ID     NAME      DRIVER    SCOPE
b19675701a96   bridge    bridge    local
2ce16bc4a975   crypto    bridge    local
b6b5fc87a0d7   host      host      local
87ae36e02aa6   none      null      local
```

2.6

now we need to write docker-compose.yml file for run redis and Django to volume and network

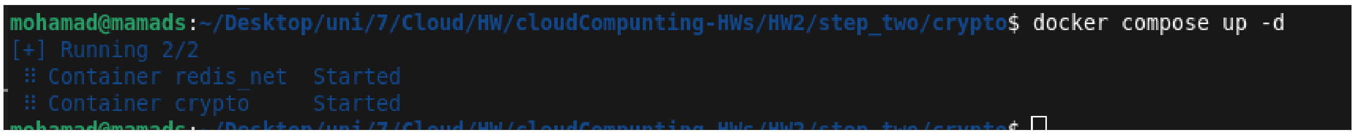
- compose file

2.7

now we run de compose file

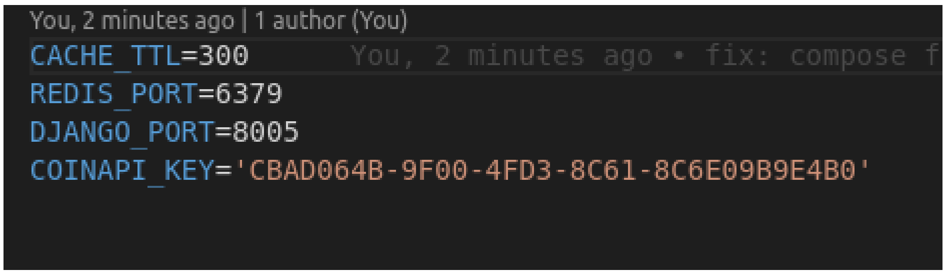
```
docker compose up -d
```

Results



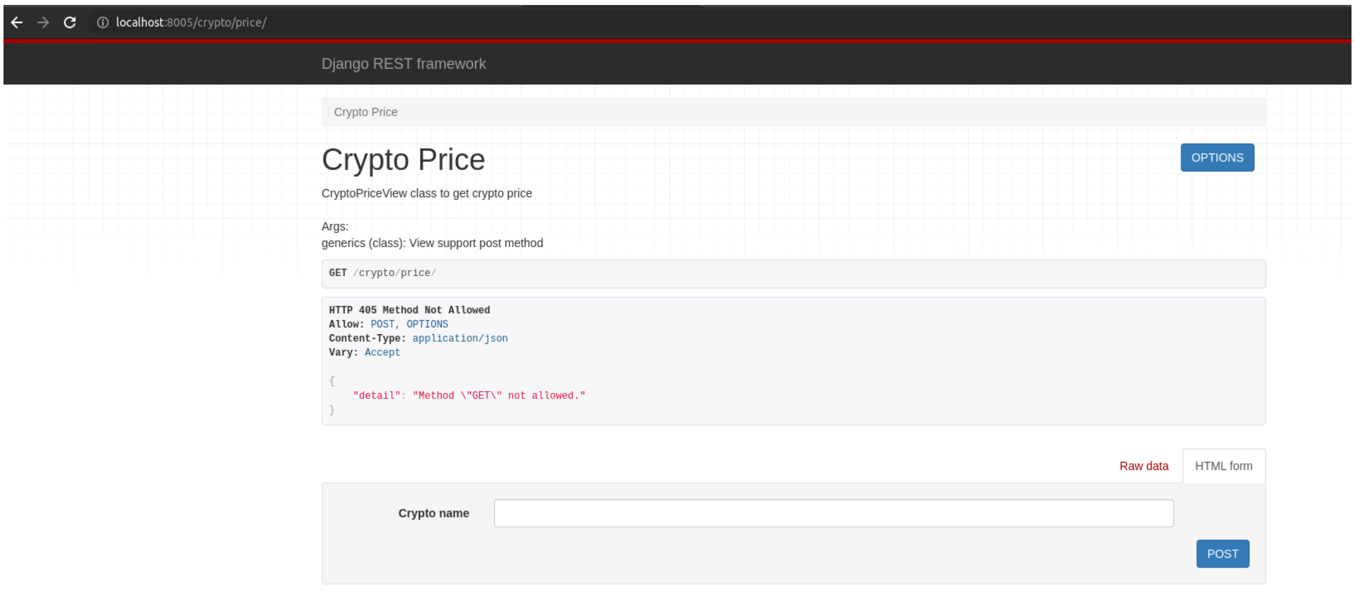
```
mohamad@mamads:~/Desktop/uni/7/Cloud/HW/cloudCompunting-HWs/HW2/step_two/crypto$ docker compose up -d
[+] Running 2/2
 ⠿ Container redis_net  Started
 ⠿ Container crypto     Started
```

- env file



```
You, 2 minutes ago | 1 author (You)
CACHE_TTL=300        You, 2 minutes ago • fix: compose f
REDIS_PORT=6379
DJANGO_PORT=8005
COINAPI_KEY='CBAD064B-9F00-4FD3-8C61-8C6E09B9E4B0'
```

we see upcoming on port 8005



Crypto Price

CryptoPriceView class to get crypto price

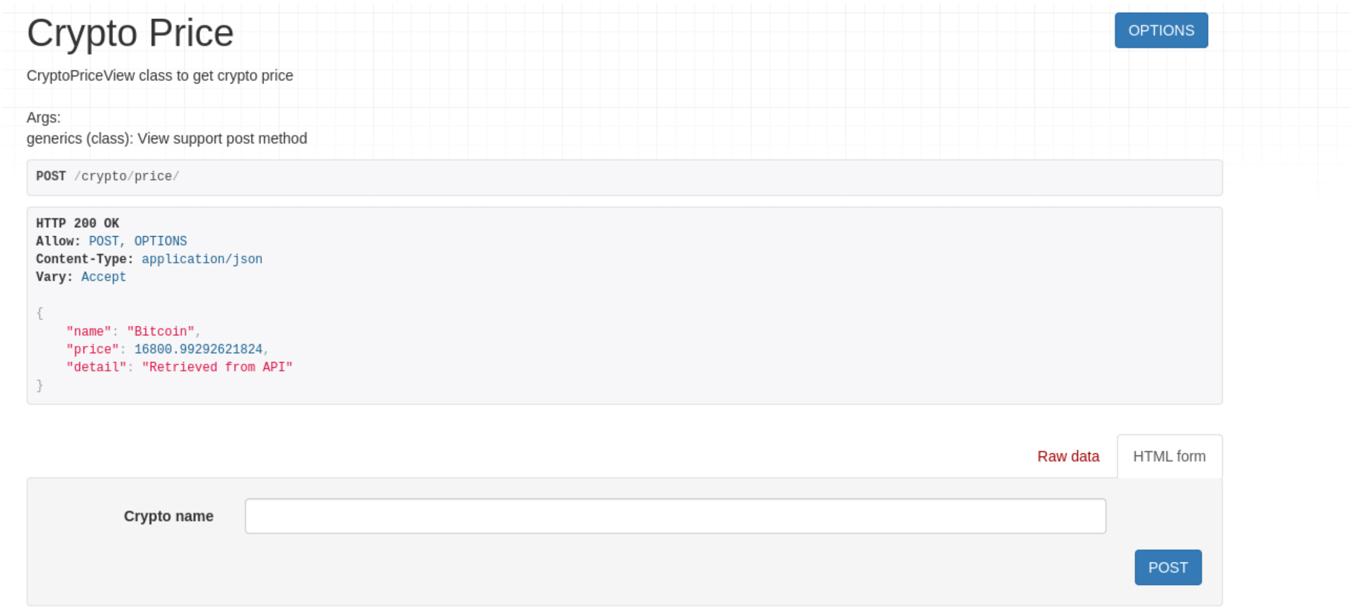
Args:
generics (class): View support post method

```
GET /crypto/price/
```

```
HTTP 405 Method Not Allowed
Allow: POST, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "detail": "Method \"GET\" not allowed."
}
```

Crypto name

Get btc data



Crypto Price

CryptoPriceView class to get crypto price

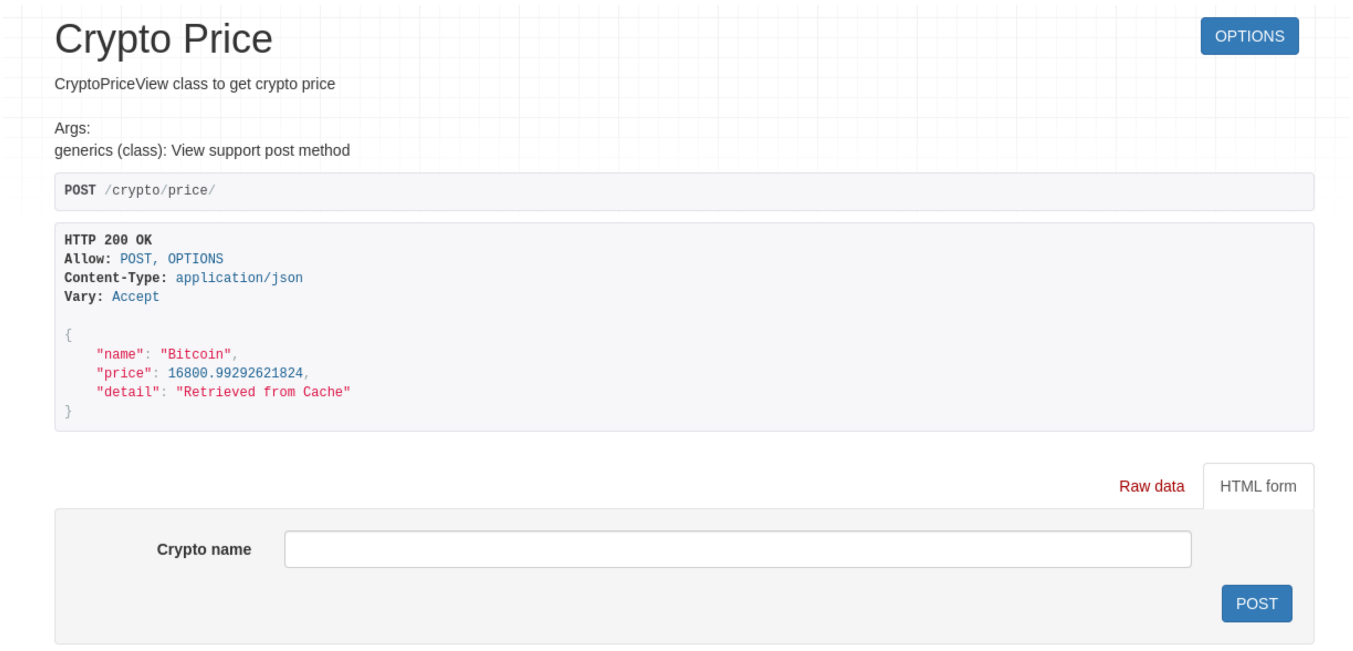
Args:
generics (class): View support post method

```
POST /crypto/price/
```

```
HTTP 200 OK
Allow: POST, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "name": "Bitcoin",
    "price": 16800.99292621824,
    "detail": "Retrieved from API"
}
```

Crypto name

Get btc for second time



Crypto Price

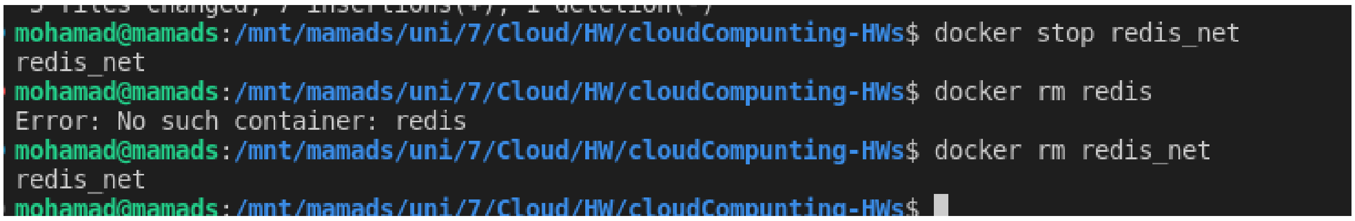
CryptoPriceView class to get crypto price

Args:
generics (class): View support post method

```
POST /crypto/price/
```

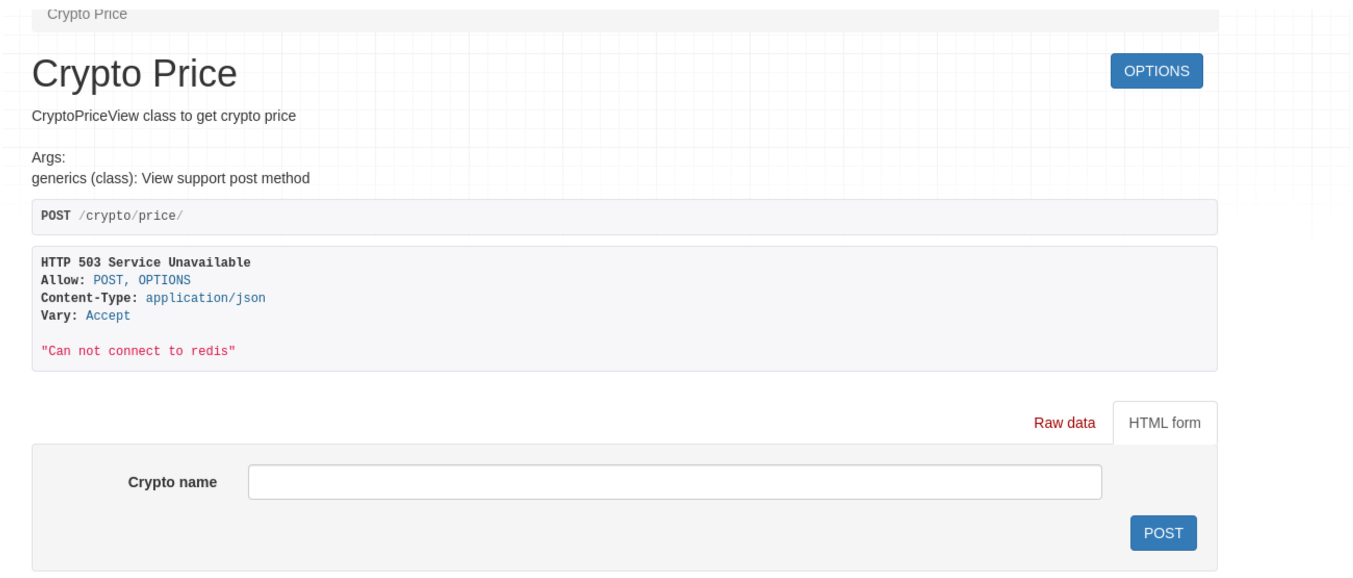
```
HTTP 200 OK
Allow: POST, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "name": "Bitcoin",
    "price": 16800.99292621824,
    "detail": "Retrieved from Cache"
}
```

Remove redis



```
mohamad@mamads:/mnt/mamads/uni/7/Cloud/HW/cloudComputing-HWs$ docker stop redis_net
redis_net
mohamad@mamads:/mnt/mamads/uni/7/Cloud/HW/cloudComputing-HWs$ docker rm redis
Error: No such container: redis
mohamad@mamads:/mnt/mamads/uni/7/Cloud/HW/cloudComputing-HWs$ docker rm redis_net
redis_net
mohamad@mamads:/mnt/mamads/uni/7/Cloud/HW/cloudComputing-HWs$
```

API response on no redis



Crypto Price

Crypto Price

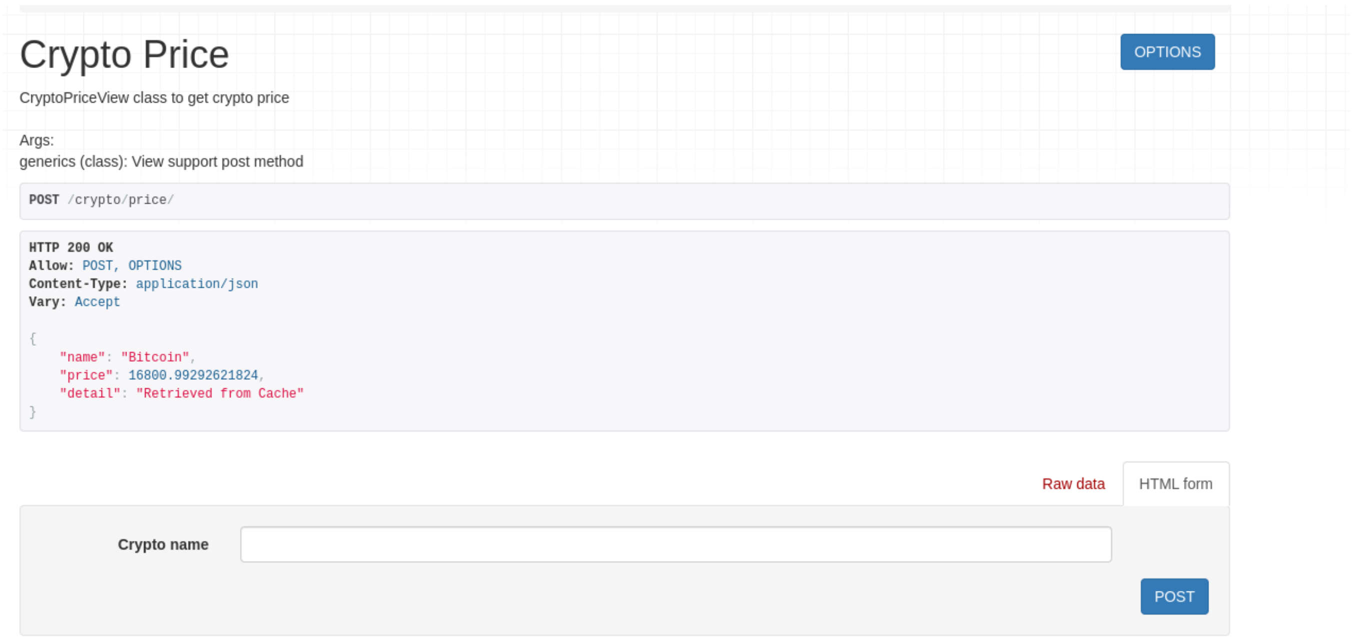
CryptoPriceView class to get crypto price

Args:
generics (class): View support post method

```
POST /crypto/price/
```

```
HTTP 503 Service Unavailable
Allow: POST, OPTIONS
Content-Type: application/json
Vary: Accept

"Can not connect to redis"
```

Start again redis and check API



Crypto Price

CryptoPriceView class to get crypto price

Args:
generics (class): View support post method

```
POST /crypto/price/
```

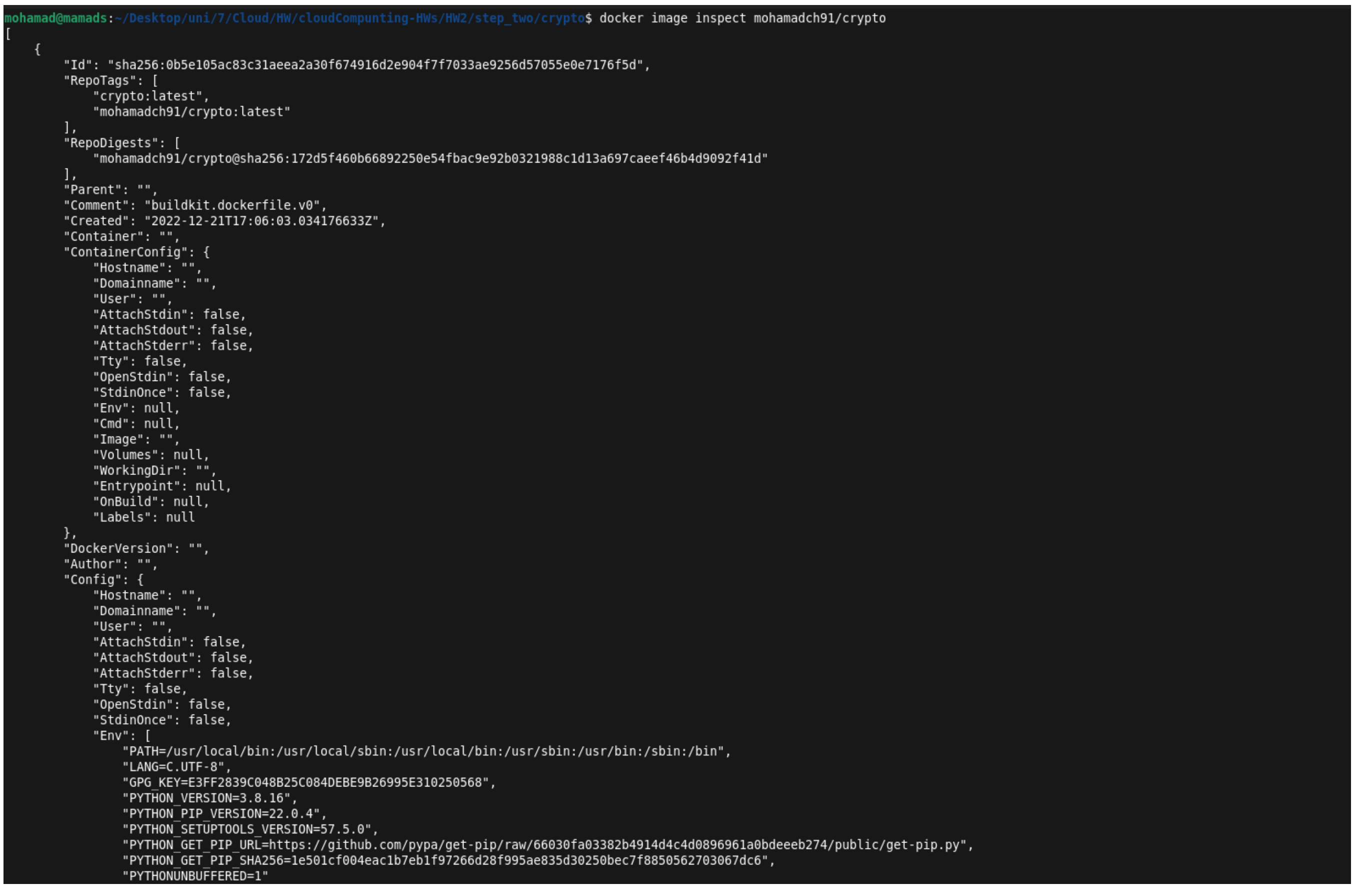
```
HTTP 200 OK
Allow: POST, OPTIONS
Content-Type: application/json
Vary: Accept

{
    "name": "Bitcoin",
    "price": 16800.99292621824,
    "detail": "Retrieved from Cache"
}
```

Report details

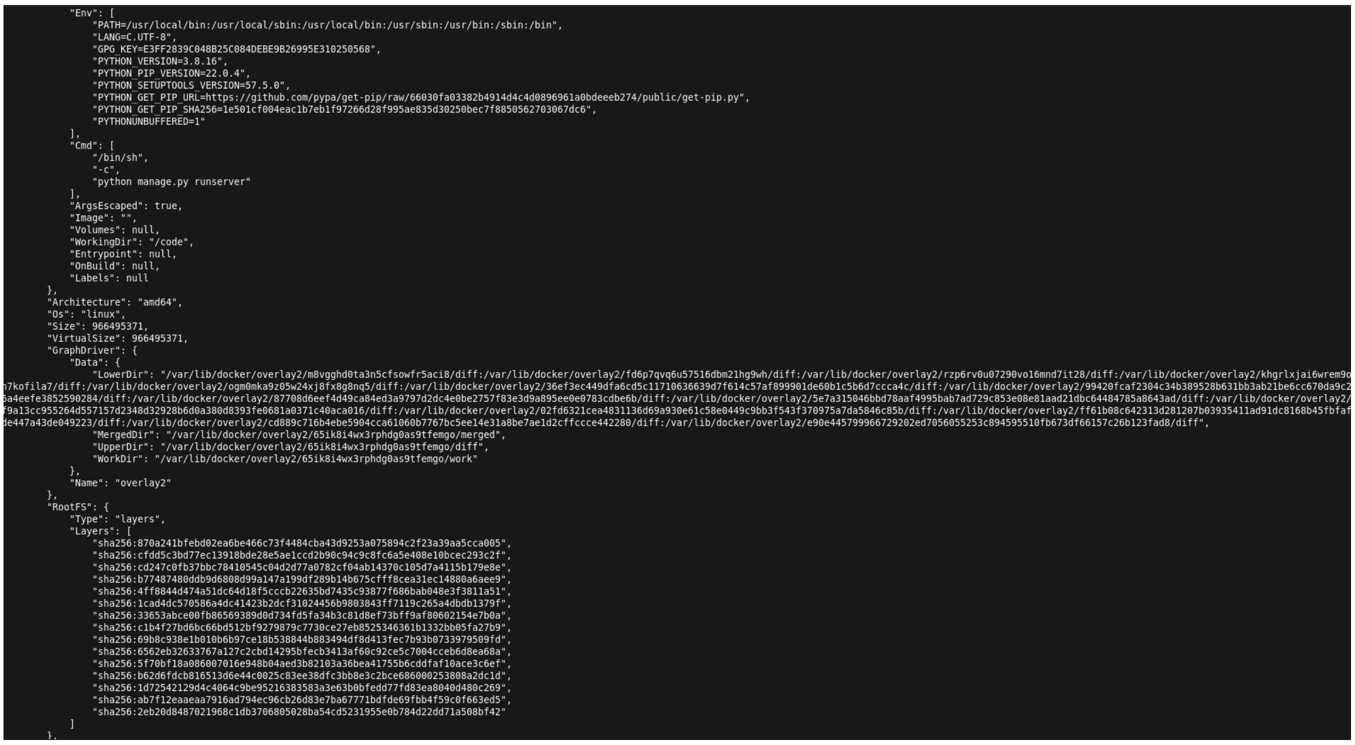
Inspect server

Inspect server image with

```
docker image inspect mohamadch91/crypto
```



```
mohamad@mamads:~/Desktop/uni/7/Cloud/HW/cloudCompunting-HWs/HW2/step_two/crypto$ docker image inspect mohamadch91/crypto
[
    {
        "Id": "sha256:0b5e105ac83c31aeea2a30f674916d2e904f7f7033ae9256d57055e0e7176f5d",
        "RepoTags": [
            "crypto:latest",
            "mohamadch91/crypto:latest"
        ],
        "RepoDigests": [
            "mohamadch91/crypto@sha256:172d5f460b66892250e54fbac9e92b0321988c1d13a697caeef46b4d9092f41d"
        ],
        "Parent": "",
        "Comment": "buildkit.dockerfile.v0",
        "Created": "2022-12-21T17:06:03.034176633Z",
        "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,
            "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=3.8.16",
                "PYTHON_PIP_VERSION=22.0.4",
                "PYTHON_SETUPTOOLS_VERSION=57.5.0",
                "PYTHON_GET_PIP_URL=https://github.com/pypa/get-pip/raw/66030fa03382b4914d4c4d0896961a0bdeeeb274/public/get-pip.py",
                "PYTHON_GET_PIP_SHA256=1e501cf004eac1b7eb1f97266d28f995ae835d30250bec7f8850562703067dc6",
                "PYTHONUNBUFFERED=1"
```

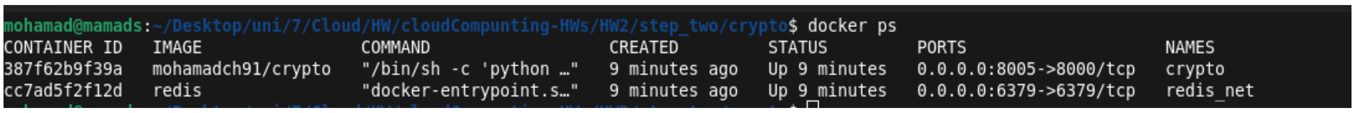


```
            "Env": [
                "PATH=/usr/local/bin:/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin",
                "LANG=C.UTF-8",
                "GPG_KEY=E3FF2839C048B25C084DEBE9B26995E310250568",
                "PYTHON_VERSION=3.8.16",
                "PYTHON_PIP_VERSION=22.0.4",
                "PYTHON_SETUPTOOLS_VERSION=57.5.0",
                "PYTHON_GET_PIP_URL=https://github.com/pypa/get-pip/raw/66030fa03382b4914d4c4d0896961a0bdeeeb274/public/get-pip.py",
                "PYTHON_GET_PIP_SHA256=1e501cf004eac1b7eb1f97266d28f995ae835d30250bec7f8850562703067dc6",
                "PYTHONUNBUFFERED=1"
            ],
            "Cmd": [
                "/bin/sh",
                "-c",
                "python manage.py runserver"
            ],
            "ArgsEscaped": true,
            "Image": "",
            "Volumes": null,
            "WorkingDir": "/code",
            "Entrypoint": null,
            "OnBuild": null,
            "Labels": null
        },
        "Architecture": "amd64",
        "Os": "linux",
        "Size": 966495371,
        "VirtualSize": 966495371,
        "GraphDriver": {
            "Data": {
                "LowerDir": "/var/lib/docker/overlay2/m8vgghd0ta3n5cfsowfr5aci8/diff:/var/lib/docker/overlay2/fd6p7qvq6u57516dbm21hg9wh/diff:/var/lib/docker/overlay2/rzp6rv0u07290vo16mnd7it28/diff:/var/lib/docker/overlay2/khgrlxjai6wrem9o
n7kofila7/diff:/var/lib/docker/overlay2/ogm0mka9z05w24xj8fx8g8nq5/diff:/var/lib/docker/overlay2/36ef3ec449dfa6cd5c11710636639d7f614c57af899901de60b1c5b6d7ccca4c/diff:/var/lib/docker/overlay2/99420fcaf2304c34b389528b631bb3ab21be6cc670da9c2
5a4eefe3852590284/diff:/var/lib/docker/overlay2/87708d6eef4d49ca84ed3a9797d2dc4e0be2757f83e3d9a895ee0e0783cdbe6b/diff:/var/lib/docker/overlay2/5e7a315046bbd78aaf4995bab7ad729c853e08e81aad21dbc64484785a8643ad/diff:/var/lib/docker/overlay2/
f9a13cc955264d557157d2348d32928b6d0a380d8393fe0681a0371c40aca016/diff:/var/lib/docker/overlay2/02fd6321cea4831136d69a930e61c58e0449c9bb3f543f370975a7da5846c85b/diff:/var/lib/docker/overlay2/ff61b08c642313d281207b03935411ad91dc8168b45fbfaf
de447a43de049223/diff:/var/lib/docker/overlay2/cd889c716b4ebe5904cca61060b7767bc5ee14e31a8be7ae1d2cffccce442280/diff:/var/lib/docker/overlay2/e90e445799966729202ed7056055253c894595510fb673df66157c26b123fad8/diff",
                "MergedDir": "/var/lib/docker/overlay2/65ik8i4wx3rphdg0as9tfemgo/merged",
                "UpperDir": "/var/lib/docker/overlay2/65ik8i4wx3rphdg0as9tfemgo/diff",
                "WorkDir": "/var/lib/docker/overlay2/65ik8i4wx3rphdg0as9tfemgo/work"
            },
            "Name": "overlay2"
        },
        "RootFS": {
            "Type": "layers",
            "Layers": [
                "sha256:870a241bfebd02ea6be466c73f4484cba43d9253a075894c2f23a39aa5cca005",
                "sha256:cfdd5c3bd77ec13918bde28e5ae1ccd2b90c94c9c8fc6a5e408e10bcec293c2f",
                "sha256:cd247c0fb37bbc78410545c04d2d77a0782cf04ab14370c105d7a4115b179e8e",
                "sha256:b77487480ddb9d6808d99a147a199df289b14b675cfff8cea31ec14880a6aee9",
                "sha256:4ff8844d474a51dc64d18f5cccb22635bd7435c93877f686bab048e3f3811a51",
                "sha256:1cad4dc570586a4dc41423b2dcf31024456b9803843ff7119c265a4dbdb1379f",
                "sha256:33653abce00fb86569389d0d734fd5fa34b3c81d8ef73bff9af80602154e7b0a",
                "sha256:c1b4f27bd6bc66bd512bf9279879c7730ce27eb8525346361b1332bb05fa27b9",
                "sha256:69b8c938e1b010b6b97ce18b538844b883494df8d413fec7b93b0733979509fd",
                "sha256:6562eb32633767a127c2cbd14295bfecb3413af60c92ce5c7004cceb6d8ea68a",
                "sha256:5f70bf18a086007016e948b04aed3b82103a36bea41755b6cddfaf10ace3c6ef",
                "sha256:b62d6fdcb816513d6e44c0025c83ee38dfc3bb8e3c2bce686000253808a2dc1d",
                "sha256:1d72542129d4c4064c9be95216383583a3e63b0bfedd77fd83ea8040d480c269",
                "sha256:ab7f12eaaeaa7916ad794ec96cb26d83e7ba67771bdfde69fbb4f59c0f663ed5",
                "sha256:2eb20d8487021968c1db3706805028ba54cd5231955e0b784d22dd71a508bf42"
            ]
        },
```

Containers list

get list of containers

```
docker ps
```

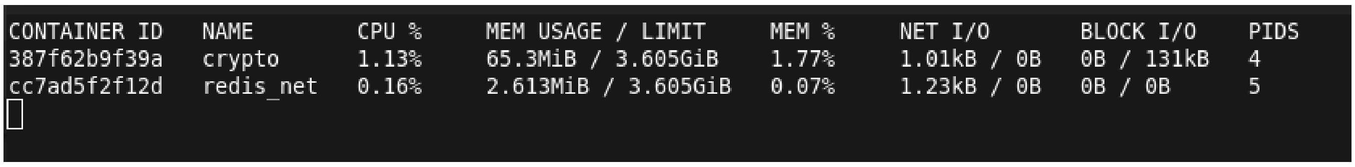


```
mohamad@mamads:~/Desktop/uni/7/Cloud/HW/cloudCompunting-HWs/HW2/step_two/crypto$ docker ps
CONTAINER ID   IMAGE               COMMAND                  CREATED         STATUS         PORTS                    NAMES
387f62b9f39a   mohamadch91/crypto  "/bin/sh -c 'python …"   9 minutes ago   Up 9 minutes   0.0.0.0:8005->8000/tcp   crypto
cc7ad5f2f12d   redis               "docker-entrypoint.s…"   9 minutes ago   Up 9 minutes   0.0.0.0:6379->6379/tcp   redis_net
```

System stats

get status of system with

```
docker stats
```



```
CONTAINER ID   NAME        CPU %     MEM USAGE / LIMIT     MEM %     NET I/O         BLOCK I/O     PIDS
387f62b9f39a   crypto      1.13%     65.3MiB / 3.605GiB    1.77%     1.01kB / 0B     0B / 131kB    4
cc7ad5f2f12d   redis_net   0.16%     2.613MiB / 3.605GiB   0.07%     1.23kB / 0B     0B / 0B       5
```