**DOCKER USEFUL COMMANDS**

**terminal into the docker container**

**docker exec -it <container\_id> bash**

You may need to terminal into a container to do things like run tests or apply migrations.

[13:54:41] (master) selfies

🙋 docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

b5e87b73f6f6 selfies\_web "python manage.py ru…" 2 seconds ago Up 1 second 0.0.0.0:8000->8000/tcp selfies\_web\_1

d8e636ad4805 postgres:10.1-alpine "docker-entrypoint.s…" 3 seconds ago Up 2 seconds 0.0.0.0:5432->5432/tcp selfies\_db\_1

aeb5cba5a482 redis:latest "docker-entrypoint.s…" 3 seconds ago Up 2 seconds 6379/tcp selfies\_redis\_1

[13:54:43] (master) selfies

🙋 docker exec -it b5e87b73f6f6 bash

root@b5e87b73f6f6:/selfies# python manage.py makemigrations

No changes detected

root@b5e87b73f6f6:/selfies#

**2. run the docker container in debug mode**

**docker-compose run --service-ports web**

If you want to debug your server, this command will let you do it. Otherwise you may get an error if you put a debugger in your code.

[13:56:59] (master) selfies

🙋 docker-compose run --service-ports web

Creating network "selfies\_default" with the default driver

Creating selfies\_redis\_1 ... done

Creating selfies\_db\_1 ... done

Performing system checks...

System check identified no issues (0 silenced).

July 24, 2019 - 17:57:11

Django version 2.1.7, using settings 'selfies.settings'

Starting ASGI/Channels version 2.2.0 development server at http://0.0.0.0:8000/

Quit the server with CONTROL-C.

Performing system checks...

System check identified no issues (0 silenced).

July 24, 2019 - 18:12:29

Django version 2.1.7, using settings 'selfies.settings'

Starting ASGI/Channels version 2.2.0 development server at http://0.0.0.0:8000/

Quit the server with CONTROL-C.

HTTP OPTIONS /app/users/ 200 [0.01, 172.25.0.1:60046]

> /selfies/app/views/account\_views.py(48)post()

-> try:

(Pdb)

**3. build the docker container**

**docker-compose build**

This runs everything in the Dockerfile. I usually run this the first time to build the project, and after that only if I add dependencies to my requirements.txt file or change anything within my Dockerfile.

[18:59:42] (master) selfies

// ♥ docker-compose build

db uses an image, skipping

redis uses an image, skipping

Building web

Step 1/7 : FROM python:3.6-stretch

---> 9167692c277e

Step 2/7 : ENV PYTHONUNBUFFERED 1

---> Using cache

---> 0533dfe1c141

Step 3/7 : ENV REDIS\_HOST "redis"

---> Using cache

---> c01adb015773

Step 4/7 : RUN mkdir /selfies

---> Using cache

---> e60377d4e9ee

Step 5/7 : WORKDIR /selfies

---> Using cache

---> 9018fb3984b0

Step 6/7 : ADD . /selfies/

---> Using cache

---> 8c6d291d99a7

Step 7/7 : RUN pip install -r requirements.txt

---> Using cache

---> 7caa2f3bf2ac

Successfully built 7caa2f3bf2ac

Successfully tagged selfies\_web:latest

**4. start the docker container**

**docker-compose up**

This will run your container/s in the terminal and will show the server output

[13:39:32] (master) selfies

🙋 docker-compose up

Creating network "selfies\_default" with the default driver

Creating selfies\_redis\_1 ... done

Creating selfies\_db\_1 ... done

Creating selfies\_web\_1 ... done

Attaching to selfies\_db\_1, selfies\_redis\_1, selfies\_web\_1

db\_1 | 2019-07-24 17:40:36.069 UTC [1] LOG: listening on IPv4 address "0.0.0.0", port 5432

db\_1 | 2019-07-24 17:40:36.069 UTC [1] LOG: listening on IPv6 address "::", port 5432

redis\_1 | 1:C 24 Jul 2019 17:40:36.085 # oO0OoO0OoO0Oo Redis is starting oO0OoO0OoO0Oo

redis\_1 | 1:C 24 Jul 2019 17:40:36.085 # Redis version=5.0.5, bits=64, commit=00000000, modified=0, pid=1, just started

redis\_1 | 1:C 24 Jul 2019 17:40:36.085 # Warning: no config file specified, using the default config. In order to specify a config file use redis-server /path/to/redis.conf

redis\_1 | 1:M 24 Jul 2019 17:40:36.086 \* Running mode=standalone, port=6379.

redis\_1 | 1:M 24 Jul 2019 17:40:36.086 # WARNING: The TCP backlog setting of 511 cannot be enforced because /proc/sys/net/core/somaxconn is set to the lower value of 128.

redis\_1 | 1:M 24 Jul 2019 17:40:36.086 # Server initialized

redis\_1 | 1:M 24 Jul 2019 17:40:36.086 # WARNING you have Transparent Huge Pages (THP) support enabled in your kernel. This will create latency and memory usage issues with Redis. To fix this issue run the command 'echo never > /sys/kernel/mm/transparent\_hugepage/enabled' as root, and add it to your /etc/rc.local in order to retain the setting after a reboot. Redis must be restarted after THP is disabled.

redis\_1 | 1:M 24 Jul 2019 17:40:36.086 \* Ready to accept connections

db\_1 | 2019-07-24 17:40:36.072 UTC [1] LOG: listening on Unix socket "/var/run/postgresql/.s.PGSQL.5432"

db\_1 | 2019-07-24 17:40:36.086 UTC [18] LOG: database system was shut down at 2019-07-24 17:39:28 UTC

db\_1 | 2019-07-24 17:40:36.090 UTC [1] LOG: database system is ready to accept connections

web\_1 | Performing system checks...

web\_1 |

web\_1 | System check identified no issues (0 silenced).

web\_1 | July 24, 2019 - 17:40:38

web\_1 | Django version 2.1.7, using settings 'selfies.settings'

web\_1 | Starting ASGI/Channels version 2.2.0 development server at http://0.0.0.0:8000/

web\_1 | Quit the server with CONTROL-C.

**5. start the docker container in the background**

**docker-compose up -d**

This will run the container but in the background so you can continue to type in the terminal. I usually run it this way if I don't really need to see what the server is returning.

[13:31:03] (master) selfies

🙋 docker-compose up -d

Creating network "selfies\_default" with the default driver

Creating selfies\_db\_1 ... done

Creating selfies\_redis\_1 ... done

Creating selfies\_web\_1 ... done

**6. see all of the docker containers currently running**

List of active docker containers which is useful for the CONTAINER ID and to know what you have running.

**docker ps**

[13:31:10] (master) selfies

🙋 docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

a2b93a900c36 selfies\_web "python manage.py ru…" 2 seconds ago Up 1 second 0.0.0.0:8000->8000/tcp selfies\_web\_1

2d39a1161aa2 postgres:10.1-alpine "docker-entrypoint.s…" 4 seconds ago Up 2 seconds 0.0.0.0:5432->5432/tcp selfies\_db\_1

62a6f364860e redis:latest "docker-entrypoint.s…" 4 seconds ago Up 2 seconds 6379/tcp selfies\_redis\_1

**7. remove all docker containers in the repository**

**docker-compose down**

I almost always follow this command with docker ps to make sure the containers were successfully removed, out of habit.

[13:37:20] (master) selfies

🙋 docker-compose down

Stopping selfies\_web\_1 ... done

Stopping selfies\_db\_1 ... done

Stopping selfies\_redis\_1 ... done

Removing selfies\_web\_1 ... done

Removing selfies\_db\_1 ... done

Removing selfies\_redis\_1 ... done

Removing network selfies\_default

**8. remove a specific docker container**

**docker kill :container\_id**

The container id is the leftmost column when doing docker ps. I sometimes do this if I need to remove a specific container that I'm not using.

[13:51:43] (master) selfies

🙋 docker ps

CONTAINER ID IMAGE COMMAND CREATED STATUS PORTS NAMES

274b1605ca94 selfies\_web "python manage.py ru…" 11 minutes ago Up 1 second 0.0.0.0:8000->8000/tcp selfies\_web\_1

a641f449edfc postgres:10.1-alpine "docker-entrypoint.s…" 11 minutes ago Up 2 seconds 0.0.0.0:5432->5432/tcp selfies\_db\_1

61b08693e242 redis:latest "docker-entrypoint.s…" 11 minutes ago Up 2 seconds 6379/tcp selfies\_redis\_1

[13:51:44] (master) selfies

🙋 docker kill 274b1605ca94

274b1605ca94

**9. view all of the docker images**

You can see all your builds by running this. For me these are either official "images" like redis or python or old builds of my projects. I only occasionally use this, it's not really a part of my daily development.

**docker images**

[18:28:17] (master) selfies

// ♥ docker images

REPOSITORY TAG IMAGE ID CREATED SIZE

selfies\_web latest 4869d063569e 7 days ago 1.32GB

python 3.6-stretch 9167692c277e 11 days ago 936MB

redis latest 598a6f110d01 12 days ago 118MB

<none> <none> 9c4676224e86 2 months ago 1e+03MB

<none> <none> 6c925f68c3a9 2 months ago 929MB

<none> <none> b44ef8ff52f4 2 months ago 929MB

<none> <none> 903b976cd478 2 months ago 1.47GB

<none> <none> f7009c6f0868 2 months ago 1.49GB

<none> <none> 52750c0c3926 2 months ago 1.48GB

<none> <none> 1c509a380925 2 months ago 1.44GB

**10. clean out any images, builds, etc., that might be hanging**

**docker system prune**

I only occasionally use this, it's not really a part of my daily development.

[18:18:35] (master) selfies

// ♥ docker system prune

WARNING! This will remove:

- all stopped containers

- all networks not used by at least one container

- all dangling images

- all build cache

Are you sure you want to continue? [y/N] y

Deleted Containers:

1f570bbf6828dadfdaa97655165943fd0b93ce6c185df2531f61a82982ec24f2

40f79091bc943c86f5f3ab7bc9d484ac9b576effc686a8a0fdb0031a465b16b4

f3399b0cd4f8c30159b15e1a2027aadeeaf9006efa4b759e4e3586d4589a0004

e0d18cddd40241de3d3b03d695afe667819c884a12fd36465c4c6d584df5aaa6

b7757cef15b49ecfe58ac6a8de3f8fcae91c71d793b7942955ccaf7c266bff92

a8fc94cd1af14f27b0ae22c1728b33fc8cef4090aa7deef1c2549c33755ed114

4de12efd1f88b9ee3541049c9b6fb3df5b1ade3b5787f888f5cf05f2e02c3cec

21d97261412949c06a4fcfb9846a6dee22f92e6a928cd6fb715ee81924917d43

073ca414e8c79e593c82f46f565adbae92a159994699dbbd0fb9df8044b3b1bb

b62f36a4a182479d25c2251e8d90fb7d2b612f31a03e943e83882ab436981879

456510a02d7ad5137dcb907484b2c8d9e07f51946be1103d294e4e253bc0e664

effee0d0a9afa74825f8f820bd363ebe1b7b54948aed38d57e84342482c367fd

5d930ece03ed8a2c2cb9178b81c9b42ef4c7397ef90d29b040ed3b1220f8ba27

a6ccf4f68831a19ab7381354f5643f3afd94cfbe0d066ff9ce4b248b90139a63

1a65f709f9b9c7d9749a04e88d3c4d18539a495160e143337c25025b10f9b24c

e98fbb022cb920dd4550751fdb735c1f73ae1dd6a049b9855bd77db1a2cbb3d1

18cb48bcb6428d4a9d5ca2f4b9c0a2fbedb3ef98c1d1fe6788029d7abea8efe9

97d0ee4625df7d107bc543318b1b45701f17720c41c7bb9614183e2cca0e26c7

5fafd4e2df31a0b84657ce1a0450c9cc8d8caa56ede9b10cddbfa6b6f55f6b50

212dae9eab339a84817b590cceb345d66290fa7f0fd7e347bffda322eb60400e

14cdd7e40ca412e1b091ddf640513ecf3f5f8bfd51cbc826e59b4901b1a0e213

ceeeb46b3aa52660a7080949543654e6436b0c050c865ef6cbed83f43bba8cc3

6c96d28cffed031a8bdd836670401b74eb52fa8f03c1dca83b73b8d0ba8530ec

9c89d36c6ee1f8fbc9d0c542d14d6d5b924817abf47a656941062eed95e607f6

58d212388ac7f15468306e306977661169e8aacdeeaba2fea4fc9cf7ebbb9e46

71963edbfe38d4578ba1338a0d03ed94f81664b202f969acf72d38ceeded5fed

4838aa870e8962e6a1c8c0e679dad8d25f5ad08824bfd45df150d148ae7097ba

abab59a1cd1fcf0a14e1d8f45d03f1787b218233a118af59f604ad890cc64dc1

5375b62e20d2660f8347f5f7355122f9fb528a1b603b1e58ceb7d7b811740410

29d7238e6d81f522f8c98b4d2e478bb304aec7a00892b8339bf1b4a0483e7040

**docker system prune --volumes**

will also remove old volumes, that aren't connected to any currently running container. However, be cautious as this might lead to data loss.

Stop and clean up your docker-compose services

**docker-compose down -v --rmi all**

Build and start services in single line:

**docker-compose up --build**

Build and start in background:

**docker-compose up -d --build**

Run command with docker-compose:

**# docker-compose exec [service\_name] [command]**

**docker-compose exec django python manage.py shell**

**docker-compose logs -f [service\_name]**

Good commands, for log I like to add --tail in order to avoid to see only last X logs  
**docker-compose logs -f --tail [number] [service\_name]**

**docker-compose up --build**

This brings the container up and builds if there are changes. This is a shorthand for docker-compose build and docker-compose up together. Comes in handy when you're making changes to your compose file.

**run all images**

**docker start $(docker ps -aq)**

**stop all images**

**docker stop $(docker ps -aq)**

**CLEANUP DOCKER IMAGES/PROCESSES/VOLUMES**

PS C:\data\minikubetests\MessageBoardApp> **docker system prune**

PS C:\data\minikubetests\MessageBoardApp> **docker system prune -a**

PS C:\data\minikubetests\MessageBoardApp> **docker image prune**

PS C:\data\minikubetests\MessageBoardApp> **docker ps -a**

PS C:\data\minikubetests\MessageBoardApp> **docker rm $(docker ps -a -f status=exited -q)**

PS C:\data\minikubetests\MessageBoardApp> **docker stop $(docker ps -a -q)**

PS C:\data\minikubetests\MessageBoardApp> **docker volume ls**

PS C:\data\minikubetests\MessageBoardApp> **docker volume prune**

PS C:\data\minikubetests\MessageBoardApp> **docker volume ls -f dangling=true**

PS C:\data\minikubetests\MessageBoardApp> **docker volume ls**

PS C:\data\minikubetests\MessageBoardApp> **docker images -a**