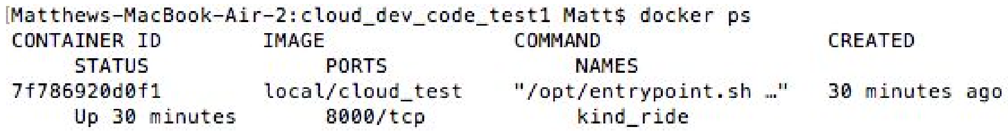
Questions

1) Explain how the config for this app works, and why when using docker compose might this be a good way to configure rather than a config file

The app is first build through the command docker build local/cloud_test . .Then it is configured through the command line with the bash command docker run local/cloud_test, which is running the application in the isolated container that is cloud_test. Created containers can be viewed through the command docker ps.



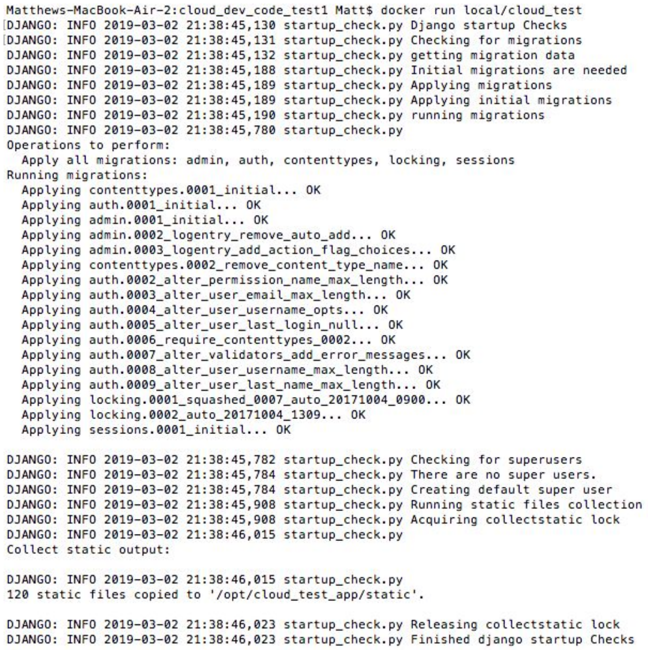
```
[Matthews-MacBook-Air-2:cloud_dev_code_test1 Matt$ docker ps
 CONTAINER ID        IMAGE               COMMAND                  CREATED
      STATUS              PORTS               NAMES
 7f786920d0f1        local/cloud_test    "/opt/entrypoint.sh …"   30 minutes ago
      Up 30 minutes       8000/tcp            kind_ride
```

A great benefit of using docker compose is the compose file that has a defaulted path ./docker-compose.yml. The YAML file stores config information, meaning that the configuration wouldn't be as manually as it would be if you were to use a config file. Docker compose can further be used to define and run multi-container docker applications by syncing them together.

2) Explain why the startup_check.py file exists and what it does

The startup_check.py file exists and to do startup checks for the condition of django. If it gets executed successfully, then the application will run.

The startup_check.py file runs a system of if/else and try/except conditions to check whether there are migrations that are needed to be applied. If further migrations are needed the migration database will be locked until the migrations are added, then unlocked. It then checks whether there is any super users and if there is not it will create a default one. Finally, the code will lock up collectstatic while it collects static files. It outputs the following in terminal:



```
Matthews-MacBook-Air-2:cloud_dev_code_test1 Matt$ docker run local/cloud_test
[DJANGO: INFO 2019-03-02 21:38:45,130 startup_check.py Django startup Checks
[DJANGO: INFO 2019-03-02 21:38:45,131 startup_check.py Checking for migrations
DJANGO: INFO 2019-03-02 21:38:45,132 startup_check.py getting migration data
DJANGO: INFO 2019-03-02 21:38:45,188 startup_check.py Initial migrations are needed
DJANGO: INFO 2019-03-02 21:38:45,189 startup_check.py Applying migrations
DJANGO: INFO 2019-03-02 21:38:45,189 startup_check.py Applying initial migrations
DJANGO: INFO 2019-03-02 21:38:45,190 startup_check.py running migrations
DJANGO: INFO 2019-03-02 21:38:45,780 startup_check.py
Operations to perform:
  Apply all migrations: admin, auth, contenttypes, locking, sessions
Running migrations:
  Applying contenttypes.0001_initial... OK
  Applying auth.0001_initial... OK
  Applying admin.0001_initial... OK
  Applying admin.0002_logentry_remove_auto_add... OK
  Applying admin.0003_logentry_add_action_flag_choices... OK
  Applying contenttypes.0002_remove_content_type_name... OK
  Applying auth.0002_alter_permission_name_max_length... OK
  Applying auth.0003_alter_user_email_max_length... OK
  Applying auth.0004_alter_user_username_opts... OK
  Applying auth.0005_alter_user_last_login_null... OK
  Applying auth.0006_require_contenttypes_0002... OK
  Applying auth.0007_alter_validators_add_error_messages... OK
  Applying auth.0008_alter_user_username_max_length... OK
  Applying auth.0009_alter_user_last_name_max_length... OK
  Applying locking.0001_squashed_0007_auto_20171004_0900... OK
  Applying locking.0002_auto_20171004_1309... OK
  Applying sessions.0001_initial... OK

DJANGO: INFO 2019-03-02 21:38:45,782 startup_check.py Checking for superusers
DJANGO: INFO 2019-03-02 21:38:45,784 startup_check.py There are no super users.
DJANGO: INFO 2019-03-02 21:38:45,784 startup_check.py Creating default super user
DJANGO: INFO 2019-03-02 21:38:45,908 startup_check.py Running static files collection
DJANGO: INFO 2019-03-02 21:38:45,908 startup_check.py Acquiring collectstatic lock
DJANGO: INFO 2019-03-02 21:38:46,015 startup_check.py
Collect static output:

DJANGO: INFO 2019-03-02 21:38:46,015 startup_check.py
120 static files copied to '/opt/cloud_test_app/static'.

DJANGO: INFO 2019-03-02 21:38:46,023 startup_check.py Releasing collectstatic lock
DJANGO: INFO 2019-03-02 21:38:46,023 startup_check.py Finished django startup Checks
```

3) Explain what the entrypoint.sh file does

The entrypoint shell is the first point of call when the bash command is executed and it allows for easy interpretation of the command. It checks for the flags --django-manage, --manage-shell, --start-service and --hot-reload. The first two flags, --django-manage and --manage-shell, are administrative flags. If either of the second two flags, --manage-shell or --start-service, are present then the startup_check.py file will be executed.