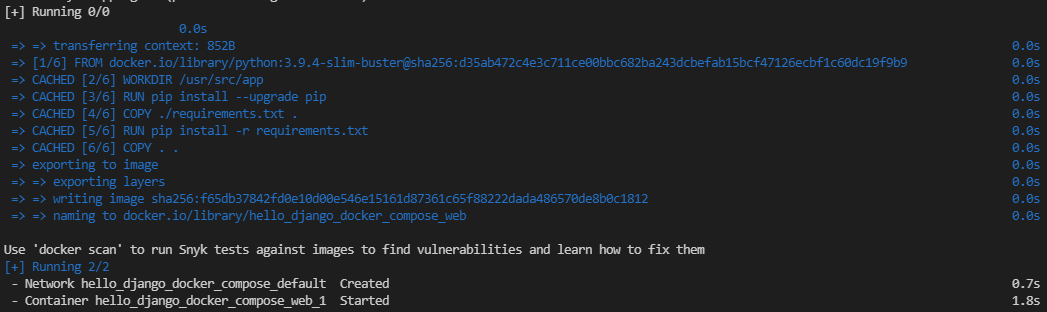
**Build the Docker image and start/stop container from CLI**

* The next command must be entered in your bash terminal from the **hello\_django\_docker\_compose/**folder.
* If your bash terminal is open to the **app/** folder, enter **cd ..** in order to return to the **hello\_django\_docker\_compose/** folder.
* Enter this command to build the Docker image based on the **docker-compose.yml** file, and to run it as a detached (background) process:

docker compose up -d



* If this command does not work, confirm that you are executing the command from the **hello\_django\_docker\_compose/** folder and that the **docker-compose.yml** file exists in that folder.
* Note that if we had defined multiple services in the **docker-compose.yml**file, this command would build the images for *all*services defined, then create and start containers for *all*of them, not only one. You will see an example of this in this week's workshop. For this example, however, we are only using one image and one container.
* You can confirm that this container is running in several ways:
  + It should show up in both **docker ps** and **docker compose ps**.
    - The **docker ps** command will show any docker containers that are running on the system.
    - On the other hand, the **docker compose ps**command will only show containers created from images built from the **docker-compose.yml** file in the current directory, or any parent directory.
  + You should also be able to see both the image and container in VS Code's Docker panel, as well as in your Docker Desktop application.
* Feel free to experiment with this container, and with the Django app itself.
* For example, can you figure out how to change the "**Hello, World!**" text in the browser to another message? Hint: Look through the files in the **helloapp/** subfolder.

**Stopping and removing containers**

* When you are done, be sure to stop the container before you continue, using this command:

docker compose stop

* + Note that, unlike the **docker stop** command, no container ID is required as an argument for **docker compose stop**. It will stop all containers that were started with the **docker compose up** command.
  + Also, where **docker stop <containerID>**can be used from any directory, in order to use the **docker compose stop** command, you must be in the same directory as the **docker-compose.yml**file that was used to start the containers, or a child directory.
* You do not have to do this, but for reference, you can go one step further and remove the containers with the command **docker compose rm**. Composed containers must first be stopped before being able to be removed in this way.
* Another command, **docker compose down**, is a shortcut to both **docker compose stop** and **docker compose rm** in a single command.