Name: Kushagra Bansal

Roll\_No: 116

Sap\_ID: 500067414

**Experiment No. 14 – Part A**

**Working with Metadata(Labels) in Docker**

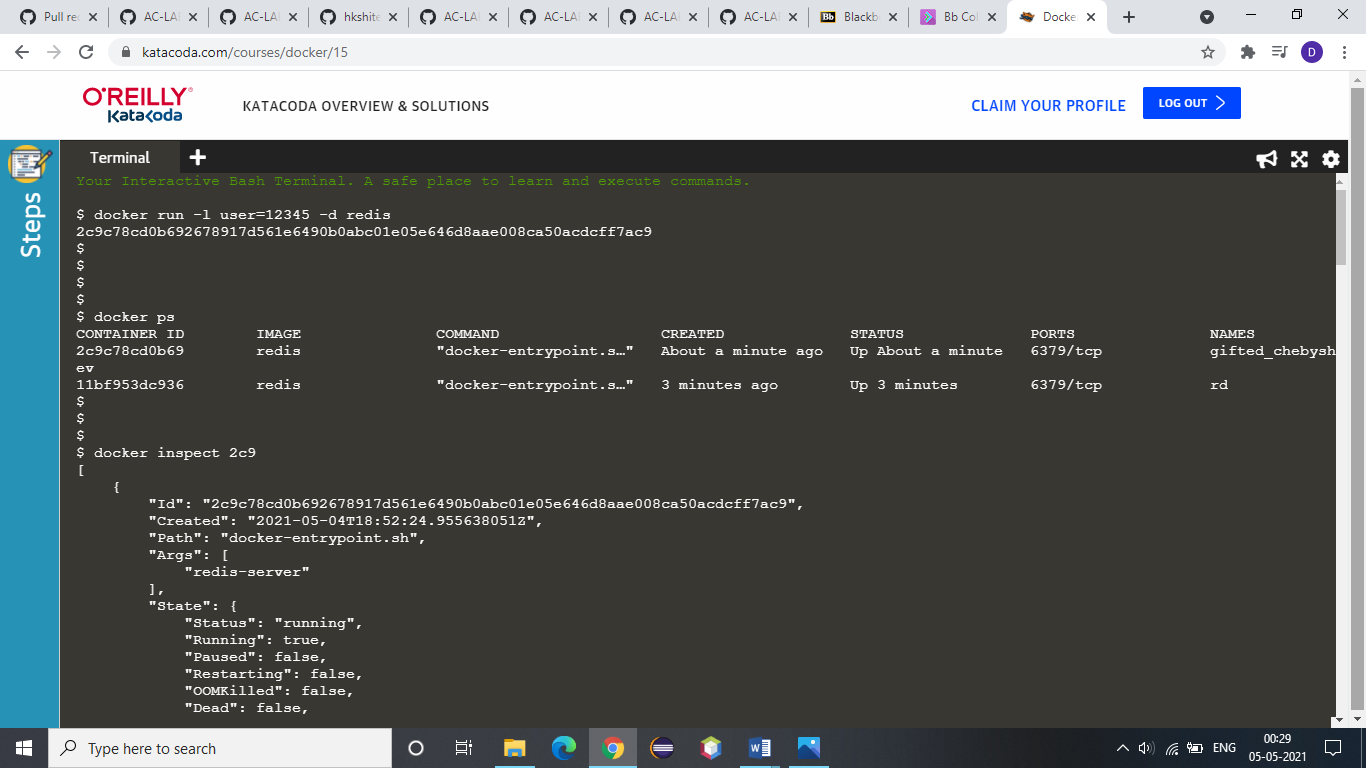
In this experiment, we are going to focus on Metadata, and Labels.

Labels can be attached to containers when they are launched via docker run. A container can have multiple labels attached to them at any one time.

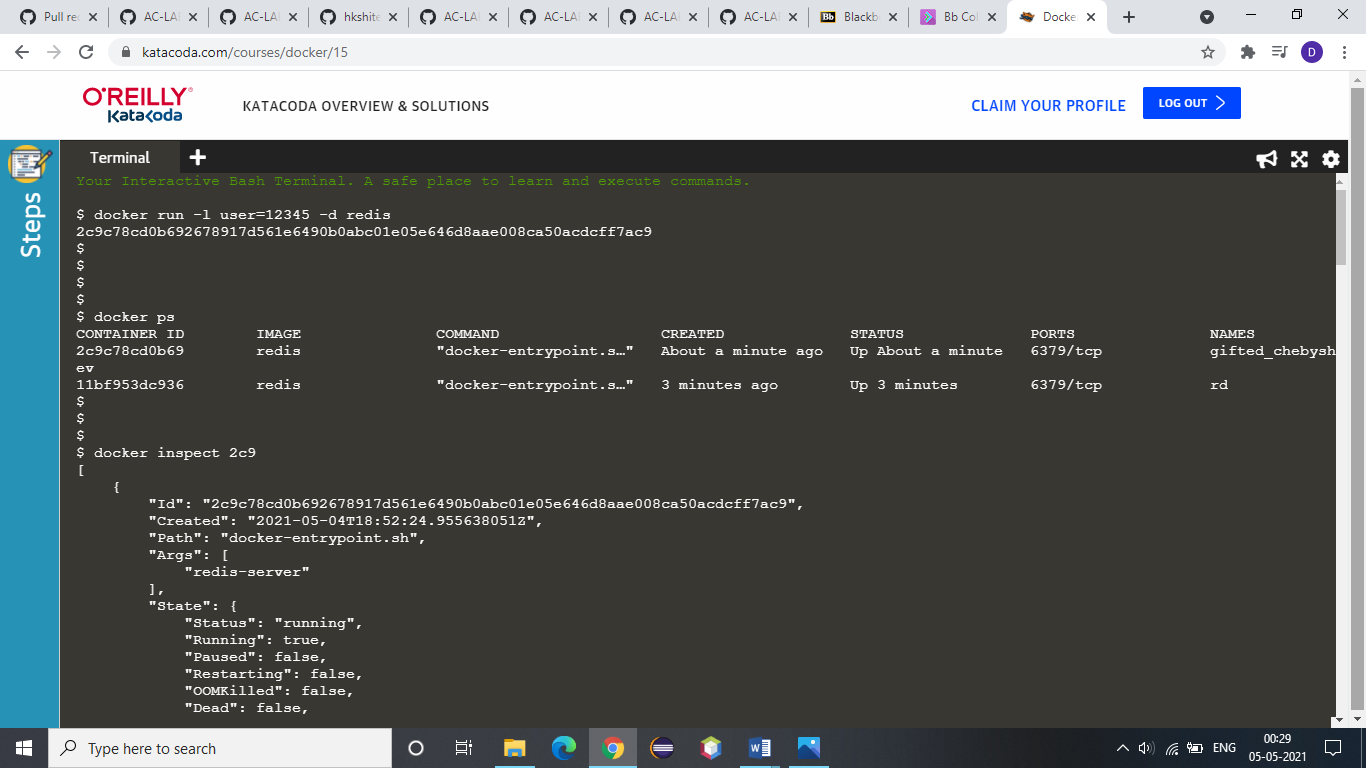
**Follow these steps below:**

1. For Single Label:

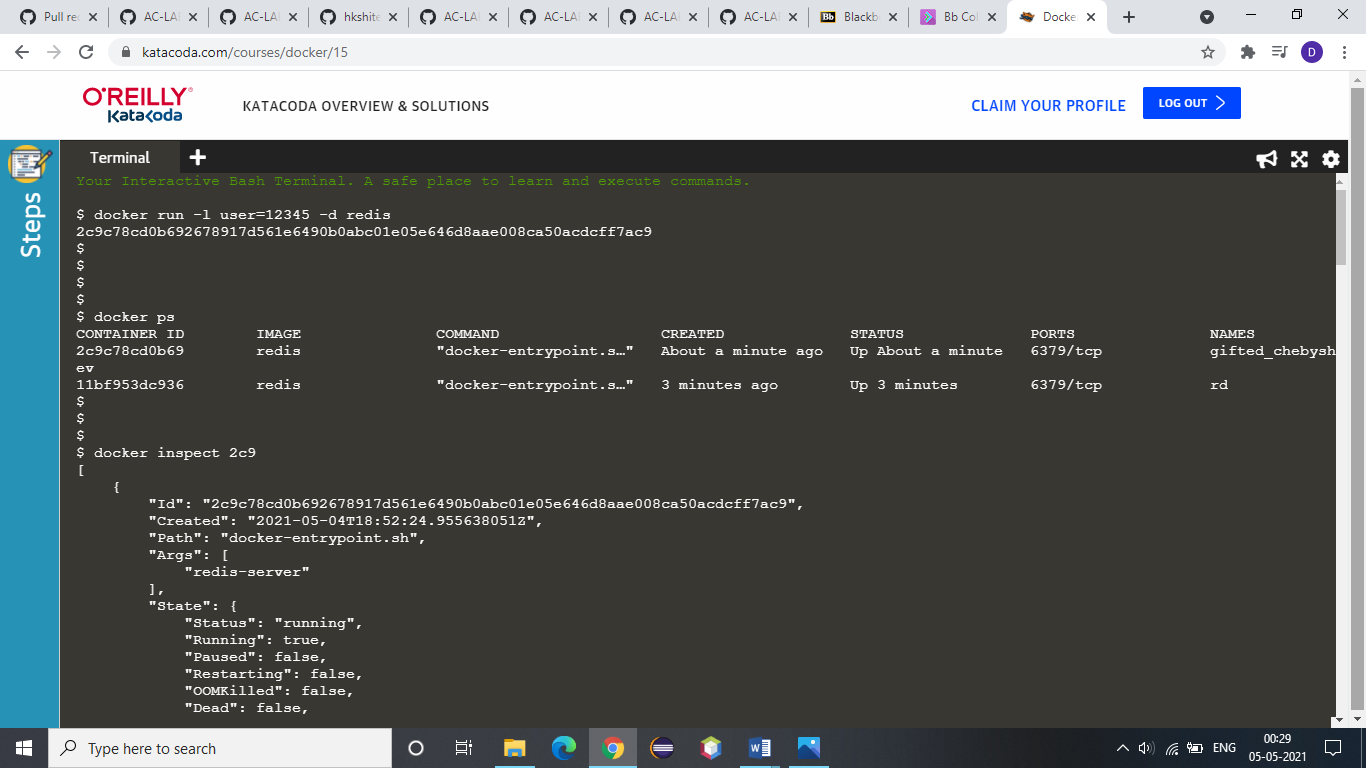
To add a single label you use the *l =<value>* option. The example below assigns a label called user with an ID to the container. This would allow us to query for all the containers running related to that particular user.

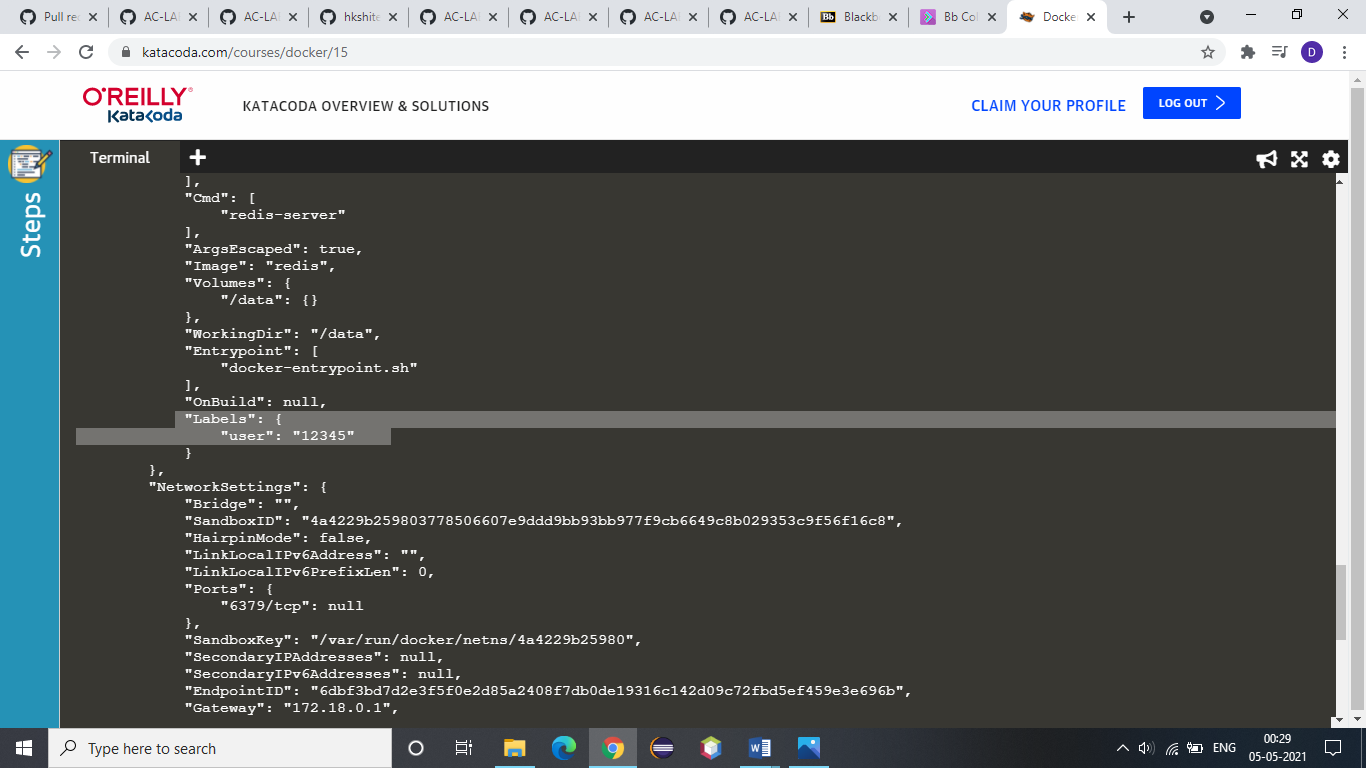


2. Use dockerps command to check:



3. Use Docker Inspect command with the container id to check the label:

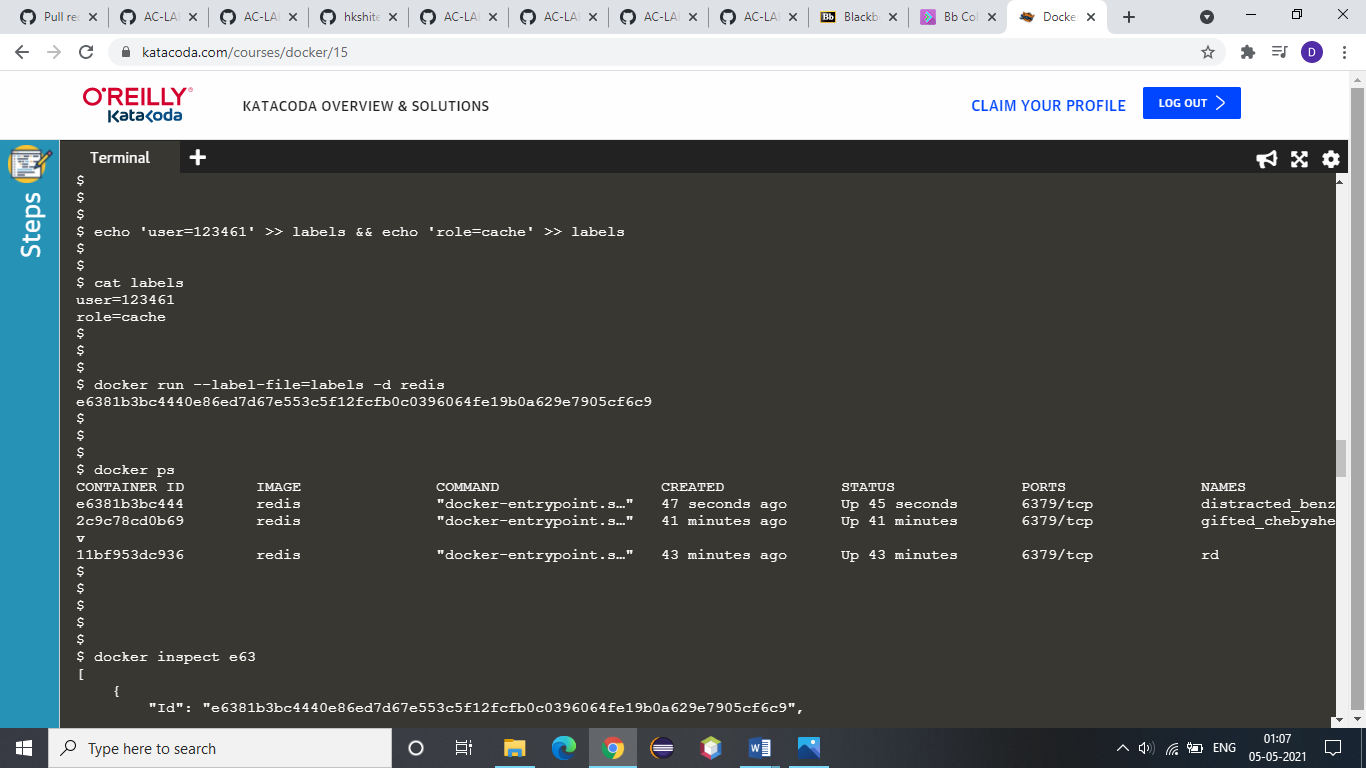




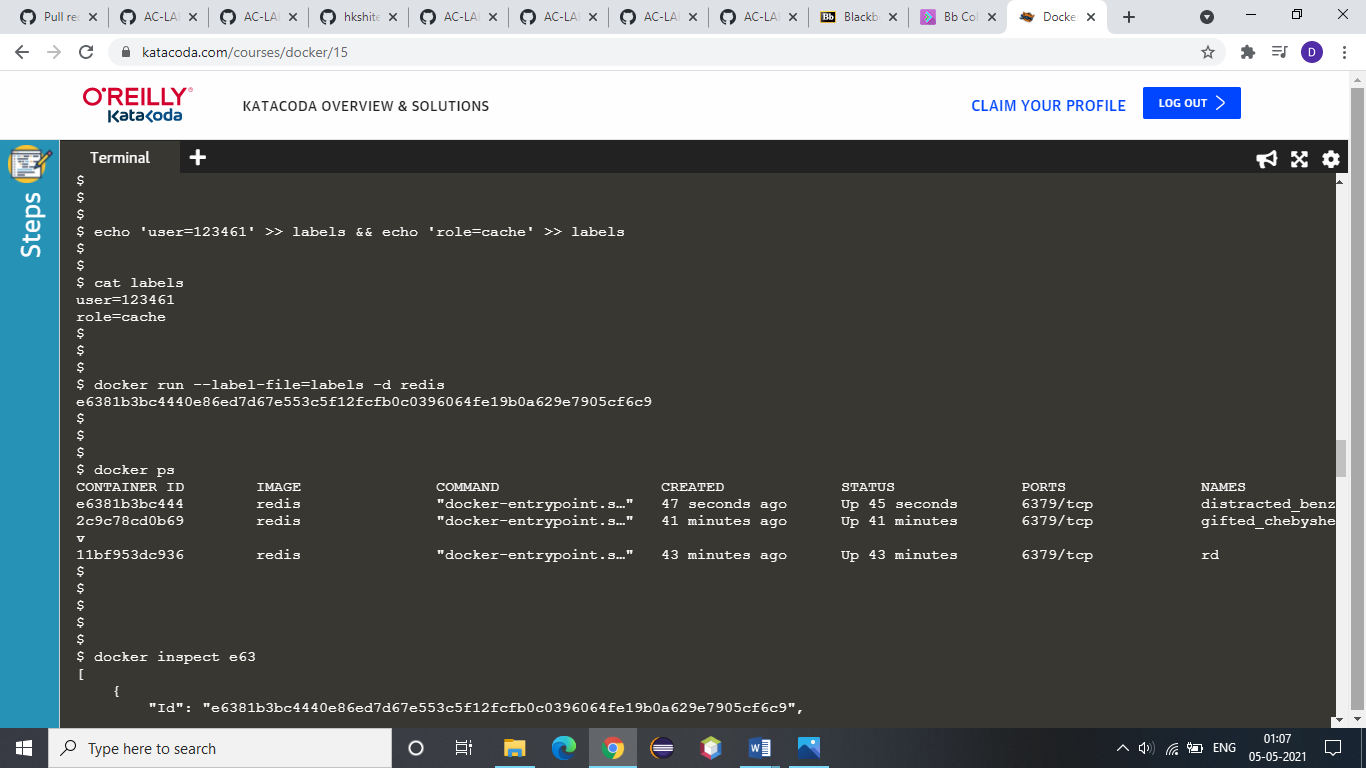
Now, for multiple labels:

If you're adding multiple labels, then these can come from an external file. The file needs to have a label on each line, and then these will be attached to the running container.

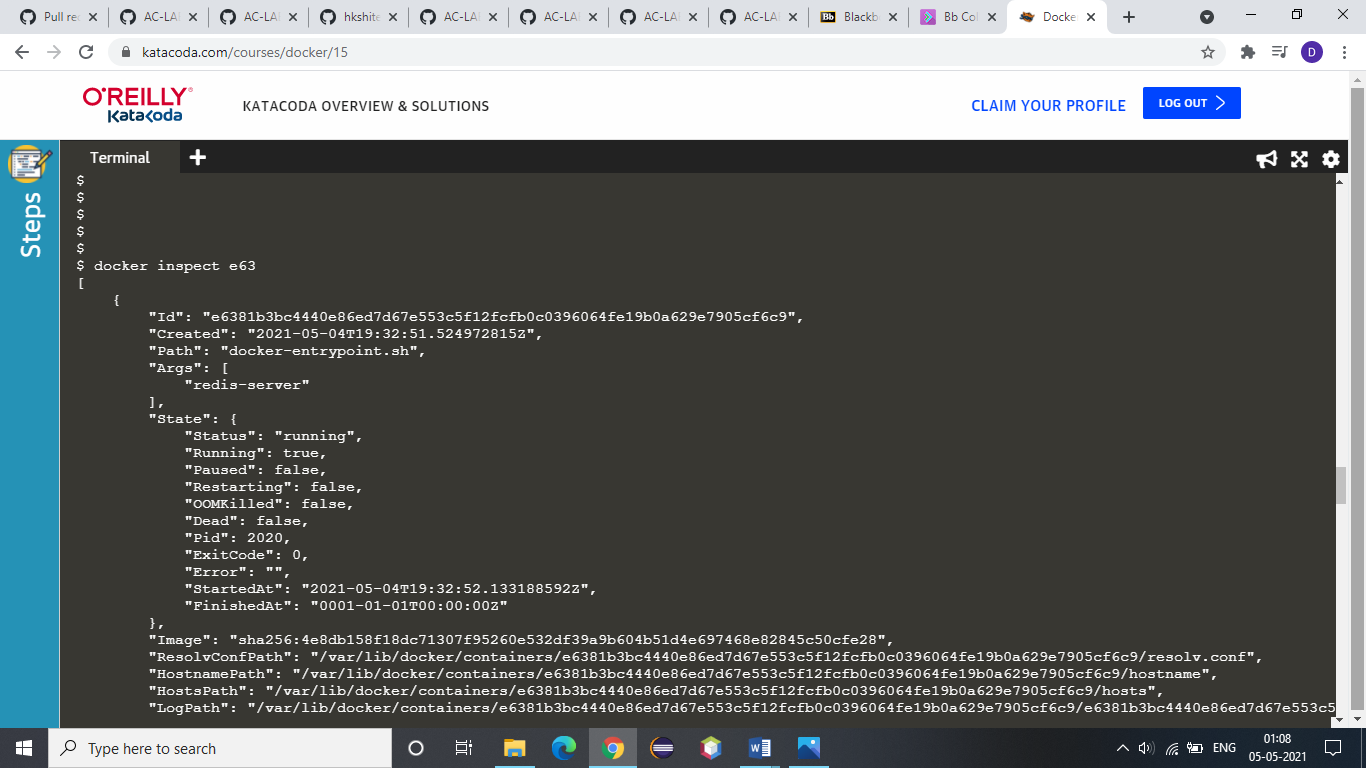
4. This line creates two labels in the file, one for the user and the second assigning a role.

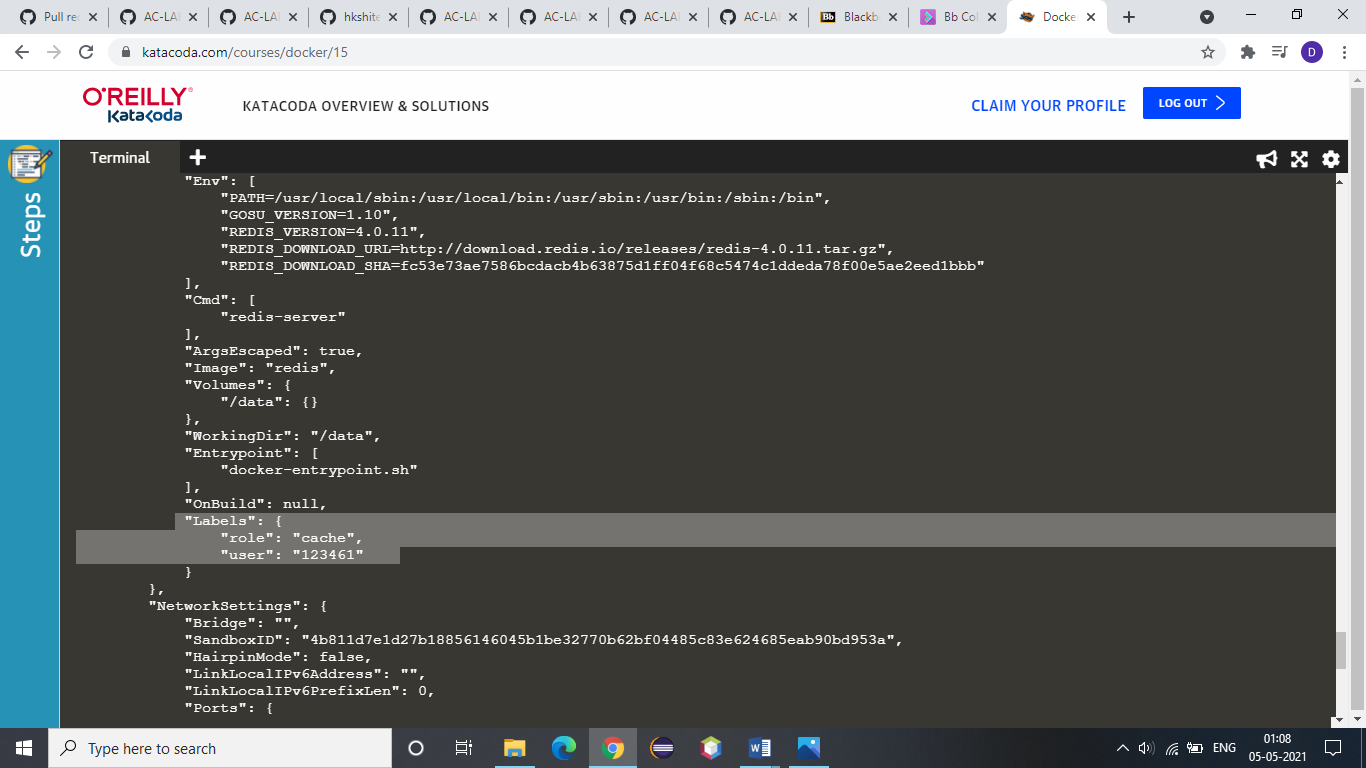


5. Now run it. The *--label-file=<filename>* option will create a label for each line in the file.

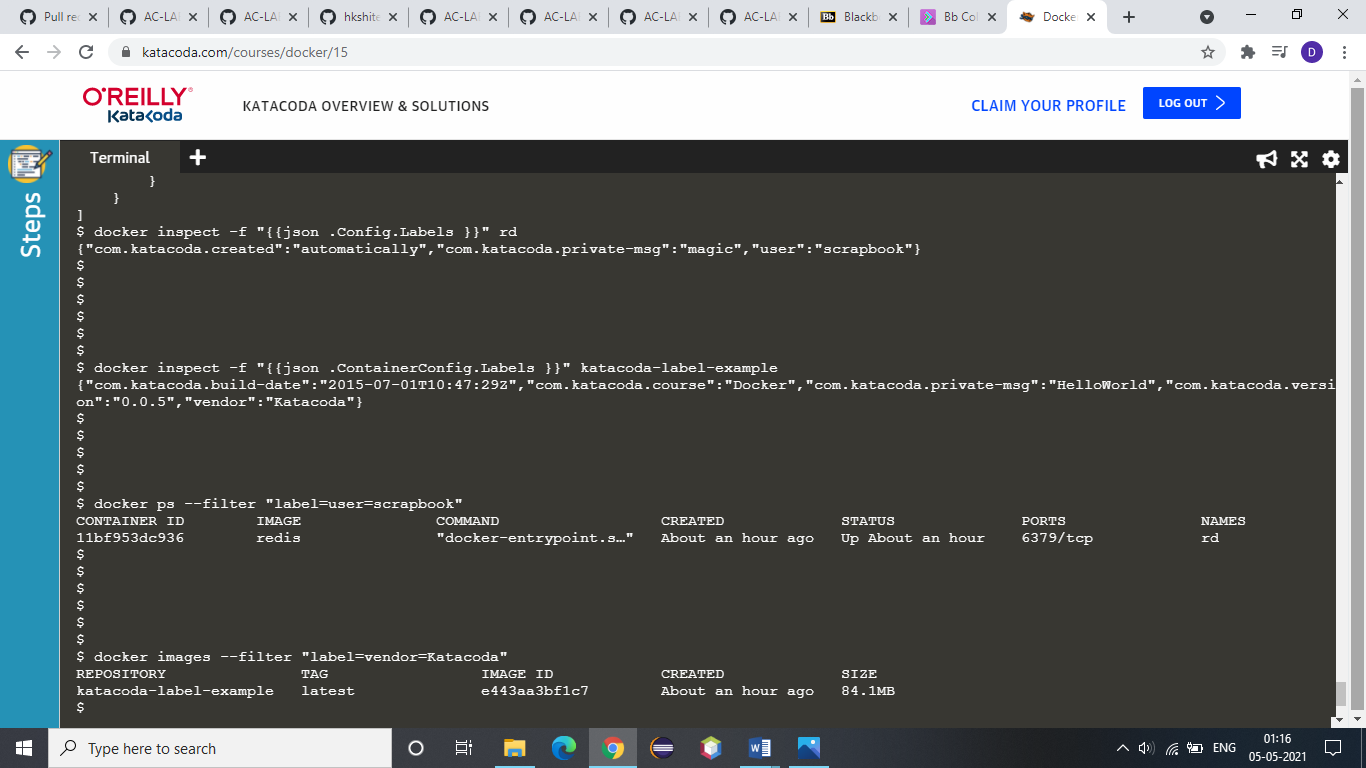


6. Now use Docker Inspect <Container Id> to check for multiple labels:

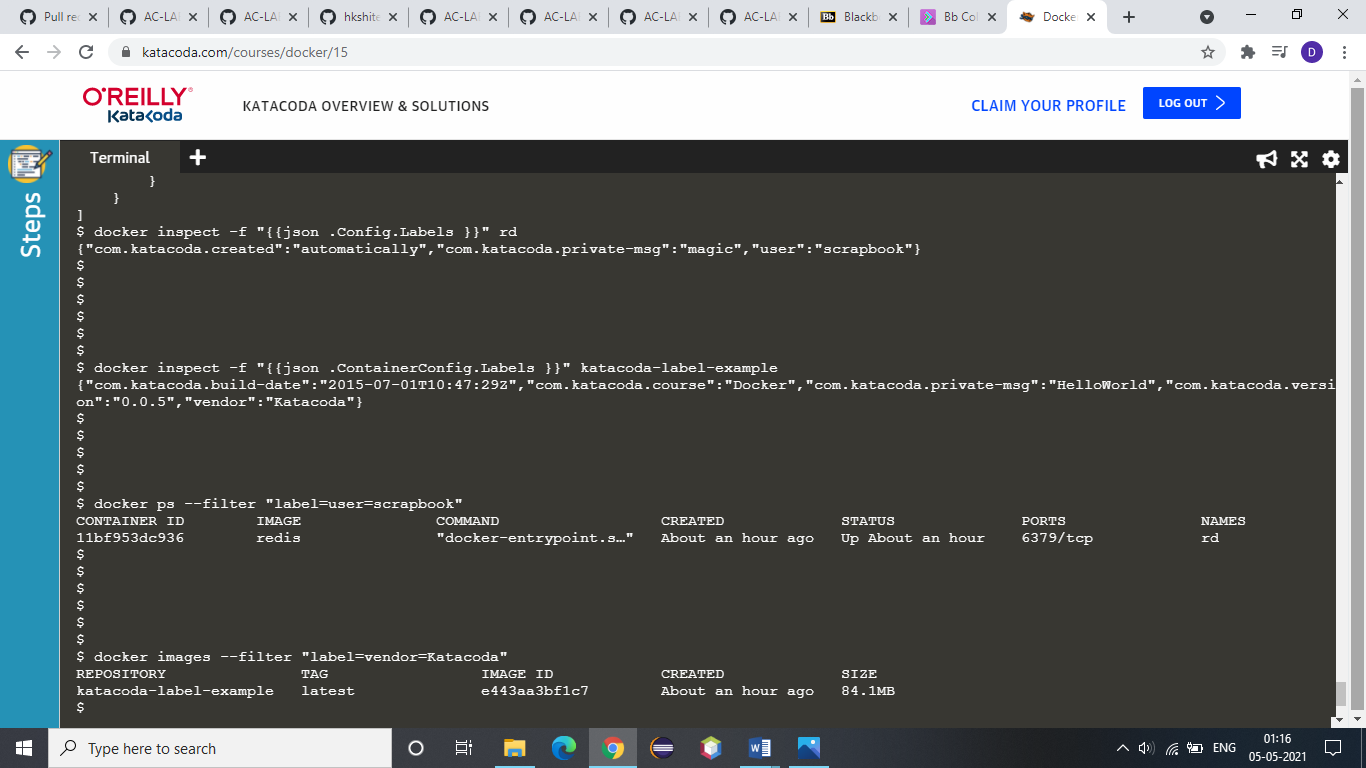




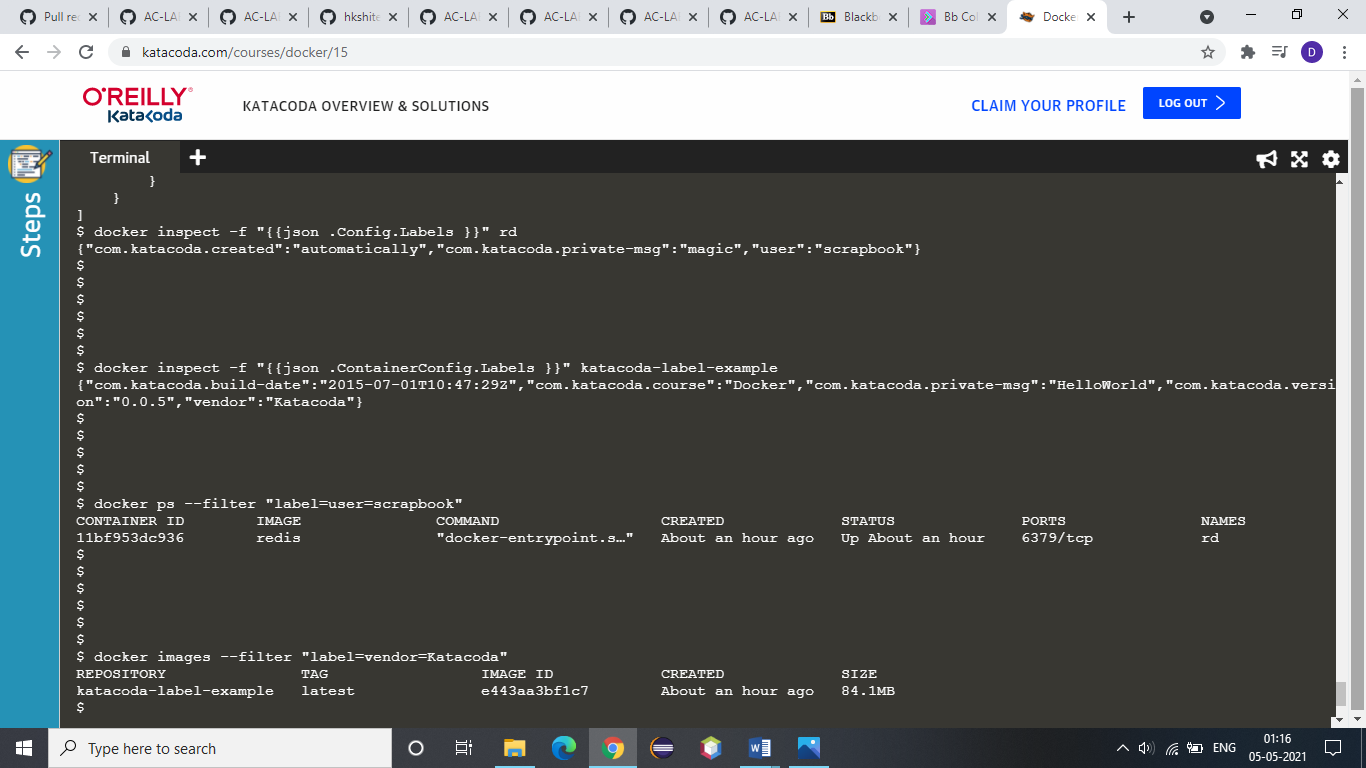
7. Using the *-f* option you can filter the JSON response to just the Labels section we're interested in.



8. Inspecting images works in the same way however the JSON format is slightly different, naming it *ContainerConfig* instead of *Config*.



9. The dockerps command allows you to specify a filter based on a label name and value. For example, the query below will return all the containers which have a *user* label key with the value *katacoda*.



10. The same filter approach can be applied to images based on the labels used when the image was built.

