Task Overview:1. Create container image that’s has Python3 and Keras or numpy installed using dockerfile

2. When we launch this image, it should automatically starts train the model in the container.

3. Create a job chain of job1, job2, job3, job4 and job5 using build pipeline plugin in Jenkins

4. Job1 : Pull the Github repo automatically when some developers push repo to Github.

5. Job2 : By looking at the code or program file, Jenkins should automatically start the respective machine learning software installed interpreter install image container to deploy code and start training( eg. If code uses CNN, then Jenkins should start the container that has already installed all the softwares required for the cnn processing).

6. Job3 : Train your model and predict accuracy or metrics.

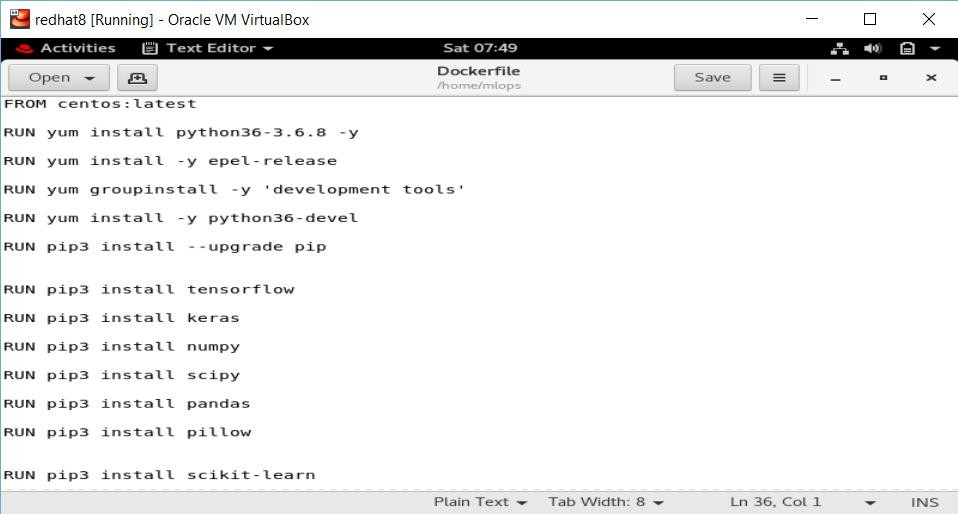
7. Job4 : if metrics accuracy is less than 80% , then tweak the machine learning model architecture.

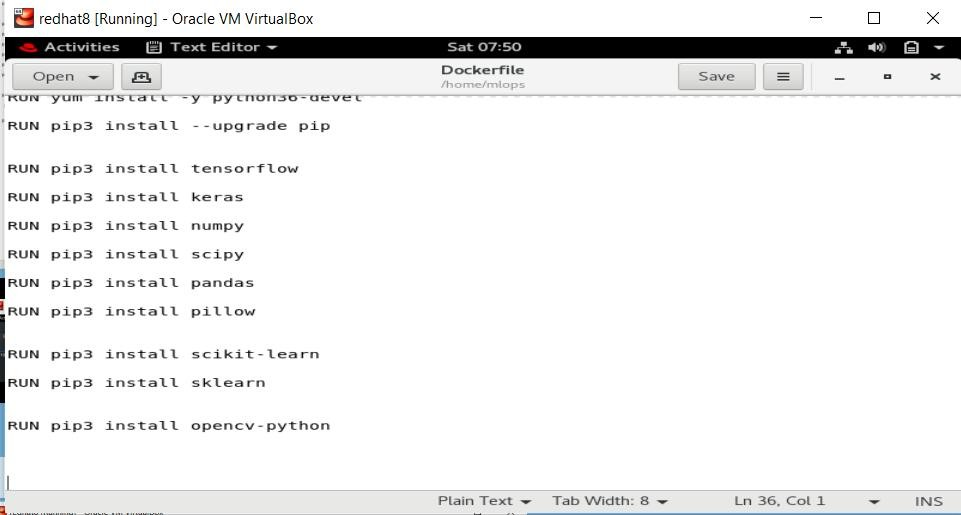
8. Job5: Retrain the model or notify that the best model is being created

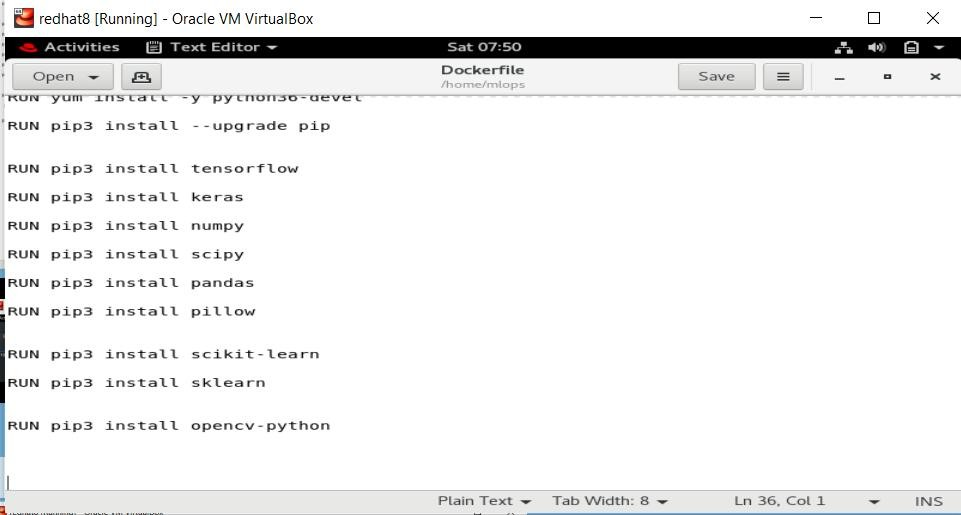
9. Create One extra job job6 for monitor : If container where app is running. fails due to any reason then this job should automatically start the container again from where the last trained model left

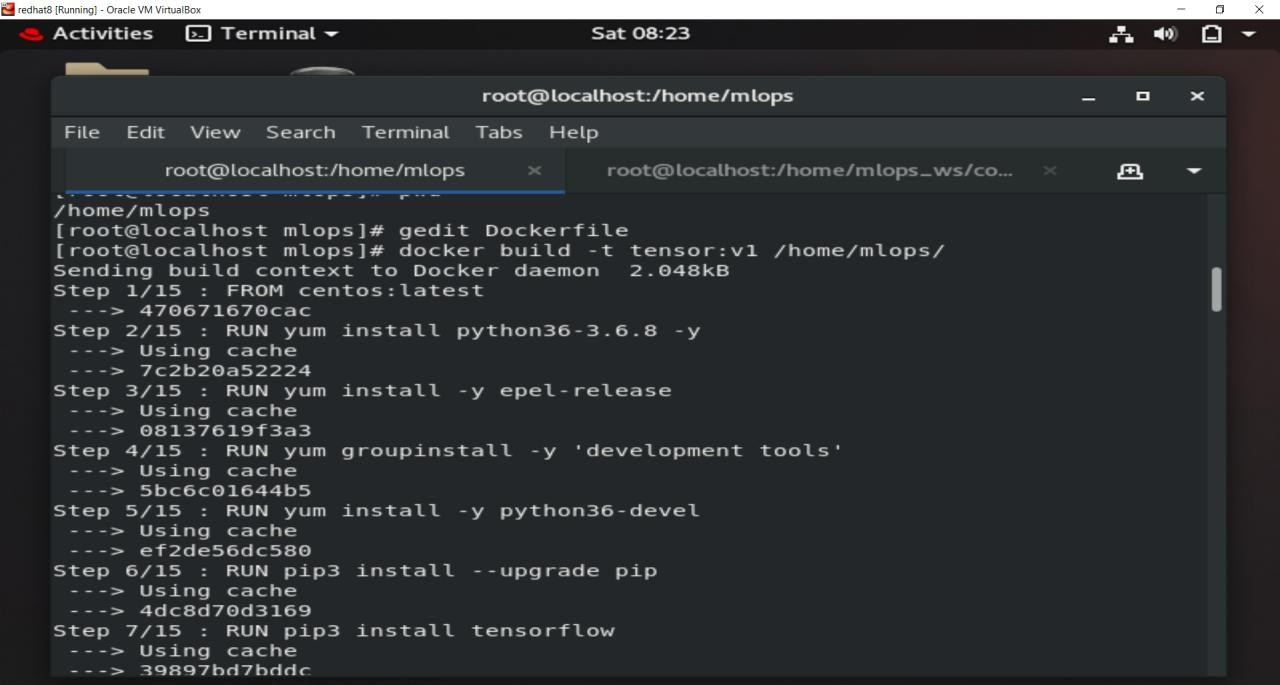
Solution:

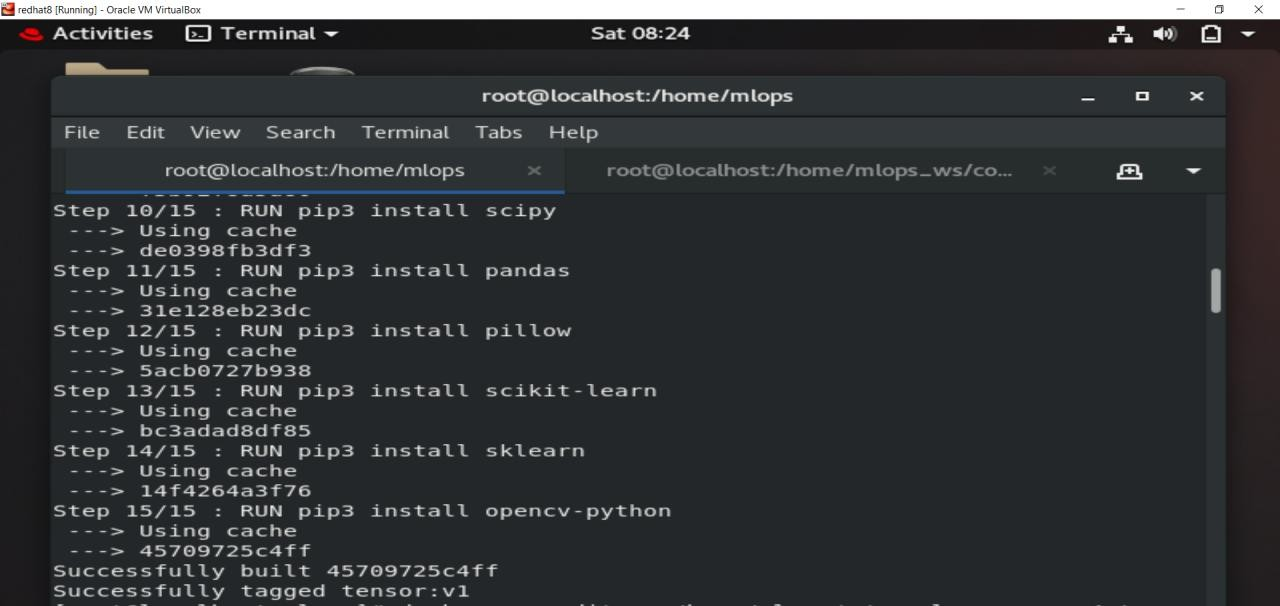
Creating a Dockerfile: Installing all required modules so our Machine learning model can run in docker container.



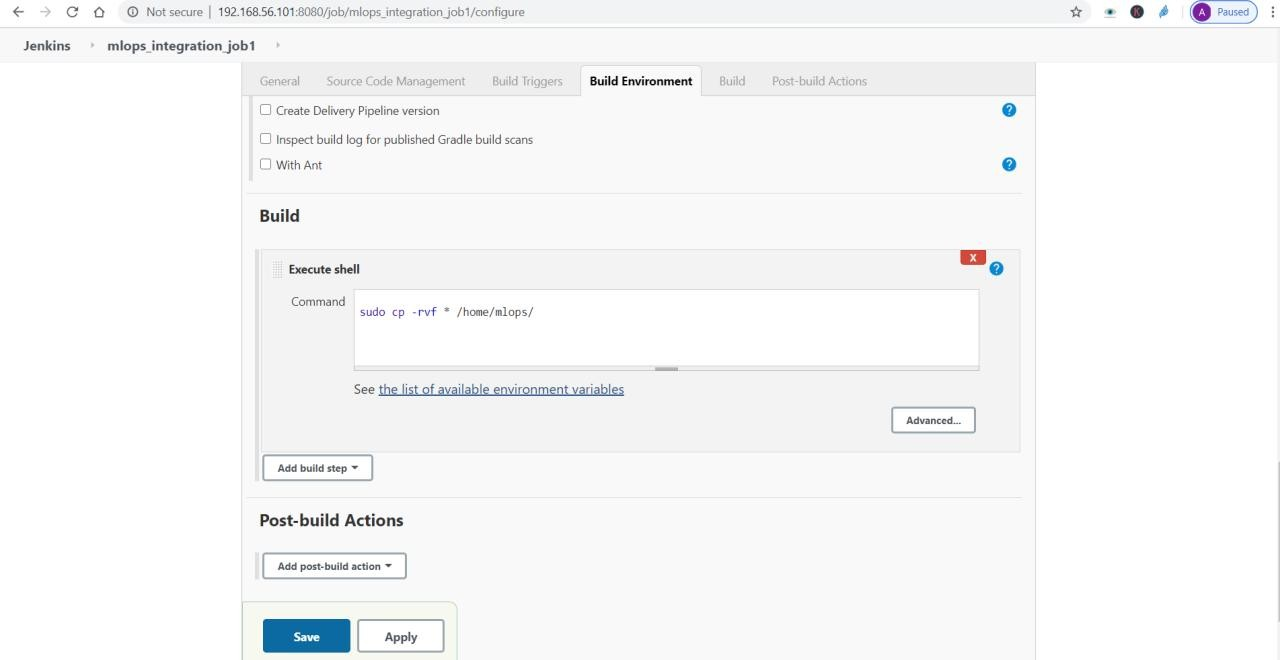




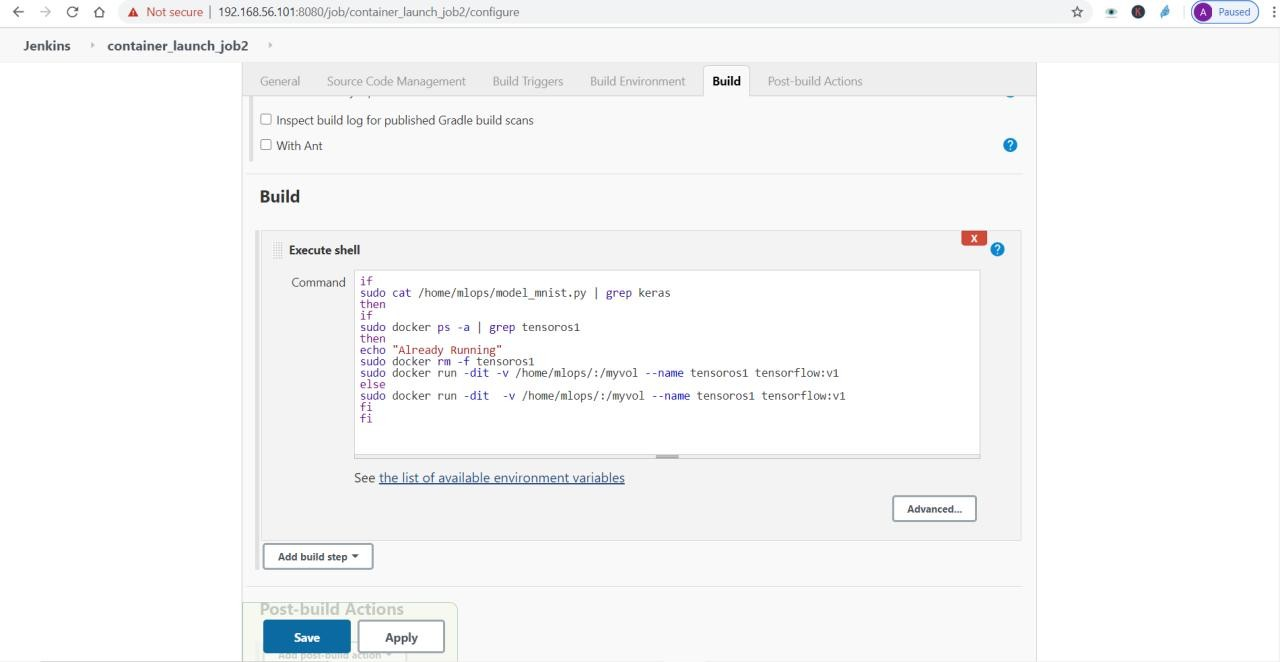




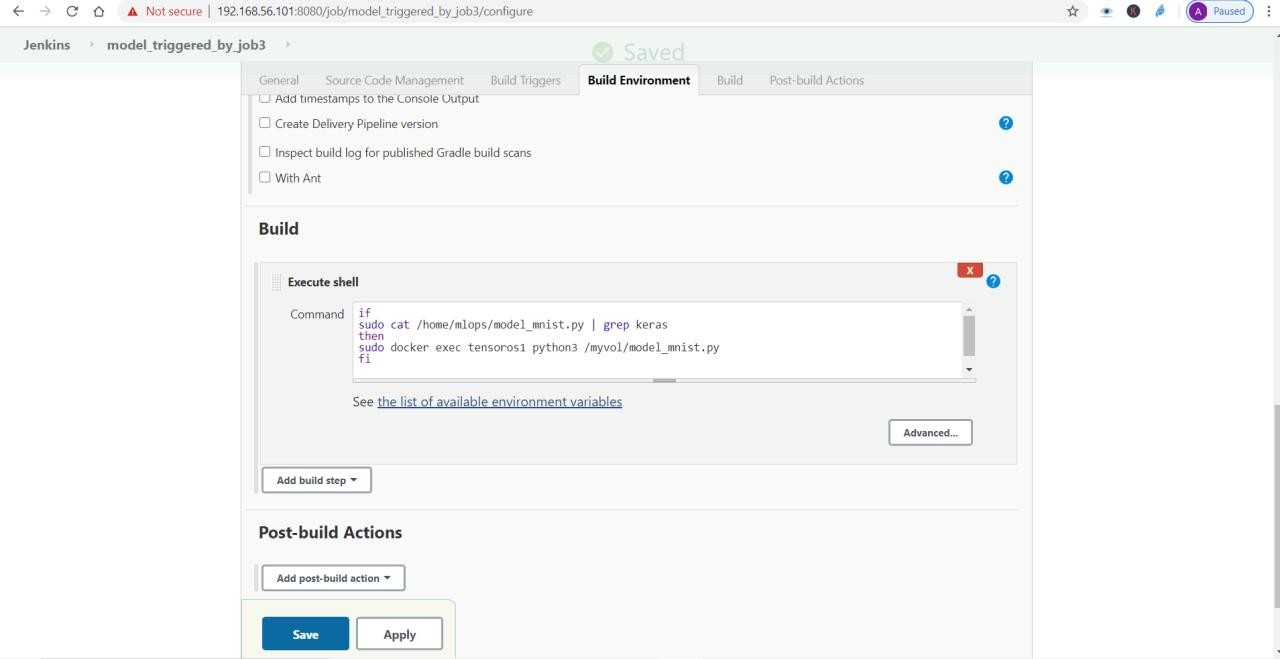
Creating Job1: In my case job 1 named as mlops\_integration\_job1 , whenever a user push its code to github ,this job 1 will copy files from github



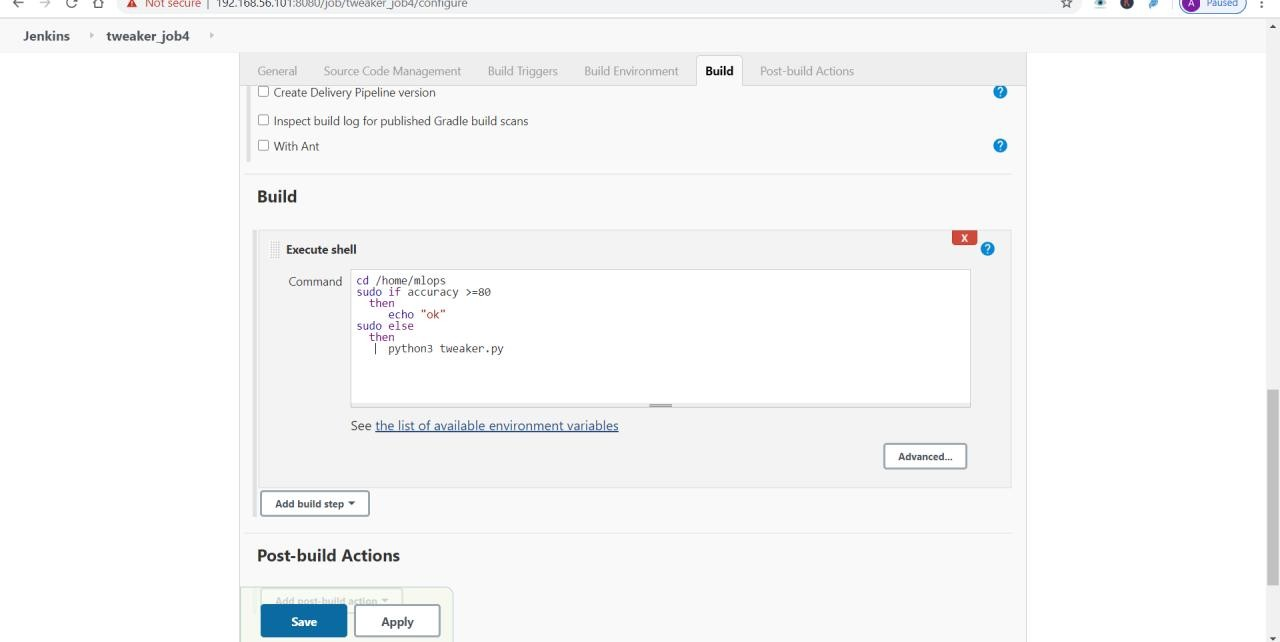
Creating Job 2 : this will launch a container , in which it will automatically start training the model



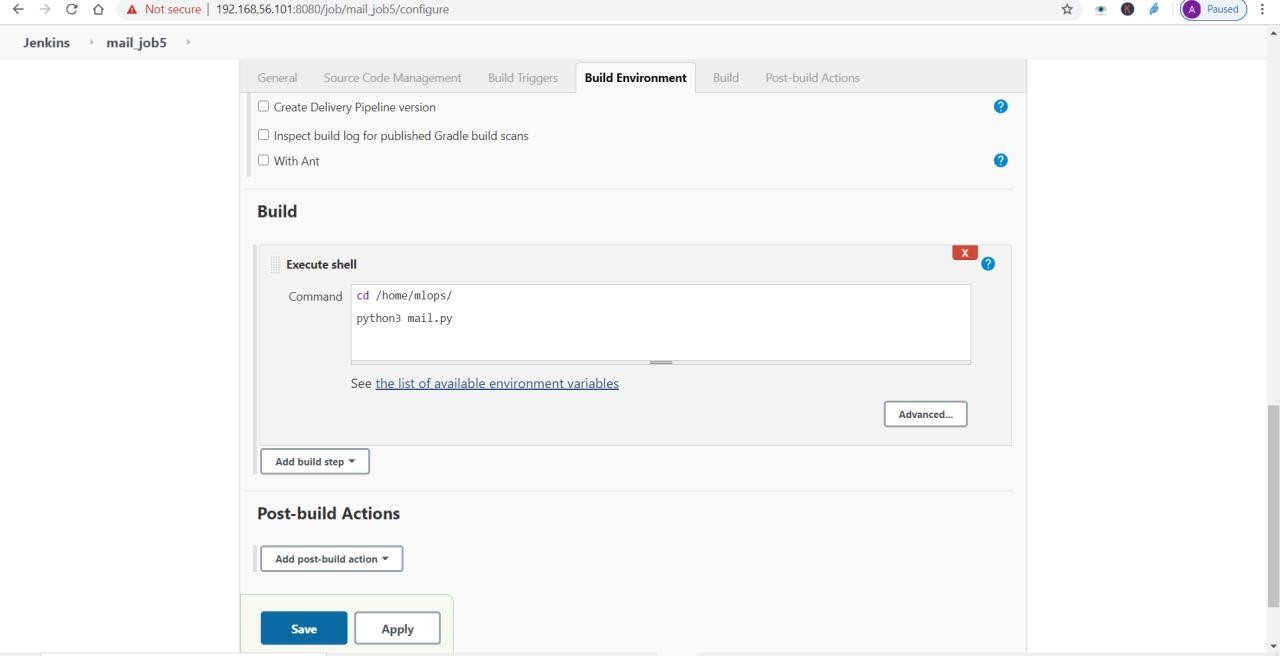
Creating Job3 : training model



Creating Job4: Tweaking the model if accuracy is lower than desired accuracy



Creating Job 5: It will send email , if accuracy is desired



N

Creating Job6: It will monitor all the jobs , if any job get fails , it will start it again

