

**TASK 3- Machine Learning Integration With DevOps (to select best Hyperparameter for dataset)**

### Automation with Jenkins and Docker-Container

In Machine Learning or Deep Learning, data scientists need to change the model several times to find the best accuracy model manually. This took a lot of time and manpower for making a machine learning or deep learning model precisely. In data science, there is no shortage of cool stuff to do the shiny new algorithms to throw at data.

**Now we can** **find the best accuracy model automatically by integrating with Jenkins and Docker-Container.**

## Task Overview:

**1.** Create a container image that 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 software required for the CNN processing).

**6.** Job3 : Train your model and predict accuracy or metrics. If metrics accuracy is less than 80%, then tweak the machine learning model architecture. Retrain the model and get the train model.

**7.** Job4: This job sent the notification to developer.

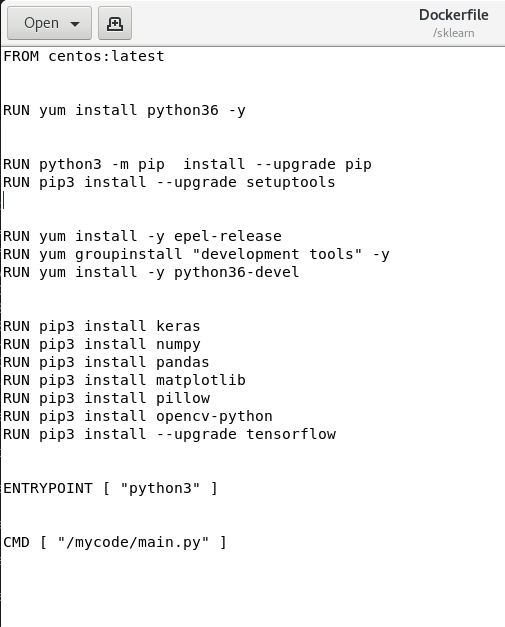
**9.** Create One extra job job5 for monitor: If container, where app is running, fails due to any reason then this job should automatically start the container again. And also sent a mail to developer.

## Project Description:

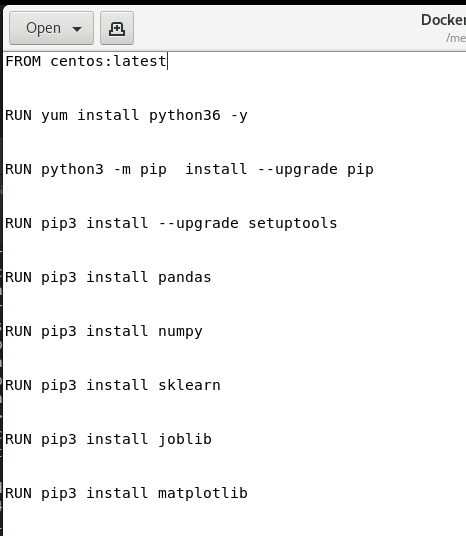
## **1. Build Docker images for TensorFlow and sklearn installed using Dockerfile:**

I use Rhel8 as BaseOs and CentOS for creating my Dockerfile.

**🡪 Dockerfile for keras models:**



Run command **"docker build -t keras:v1 ."**for creating your image



**Run command "docker build -t sklearn:v1 ." for creating the image.**

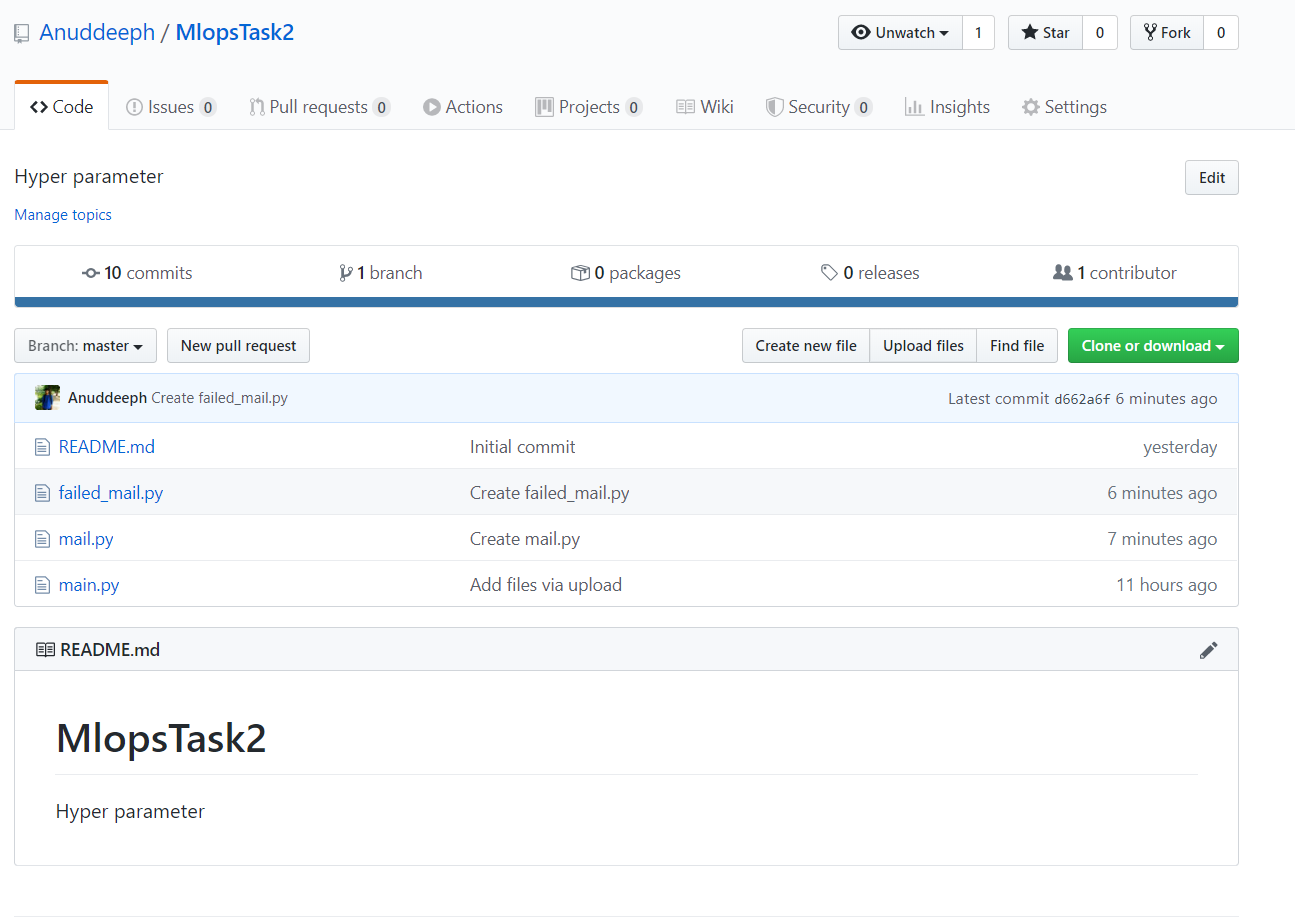
Finally my docker images is created.

## ***2. Jobs in Jenkins:***

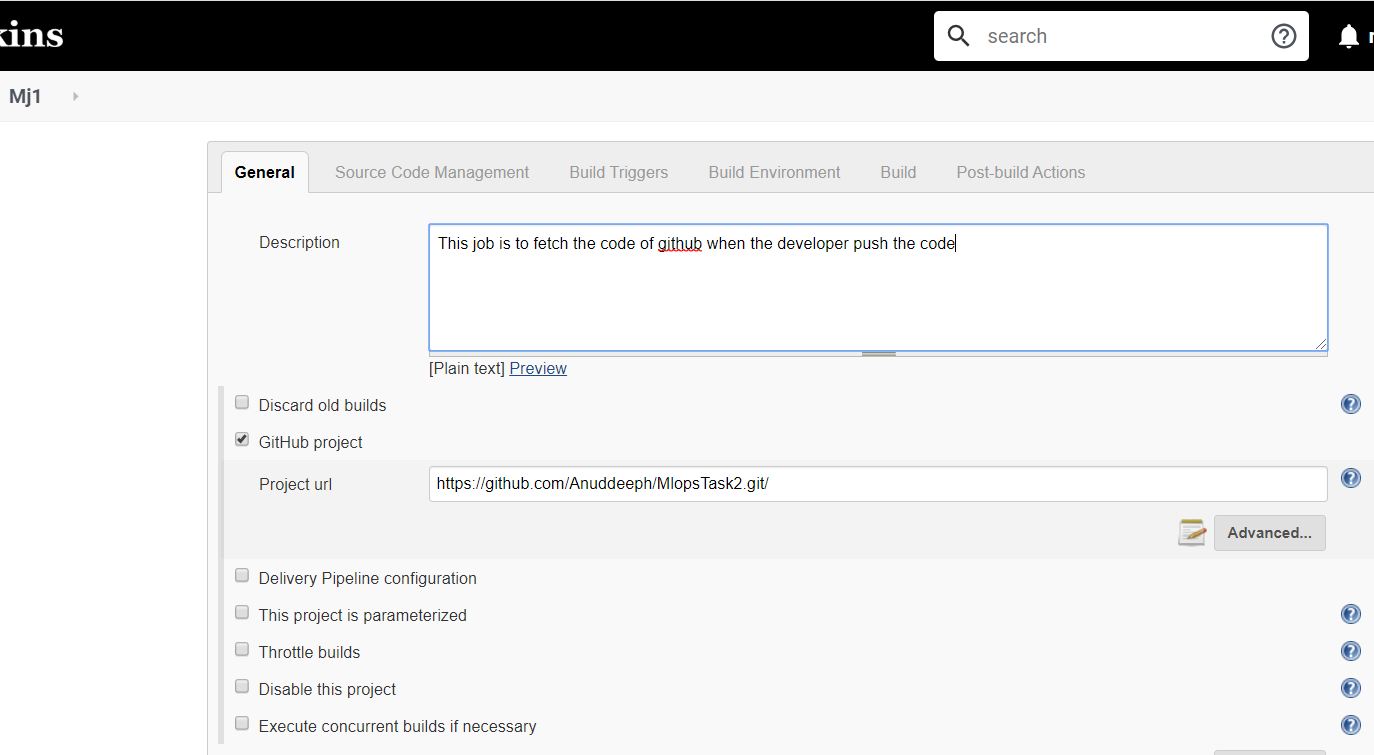
**🡪 Job1(Copy github code):**

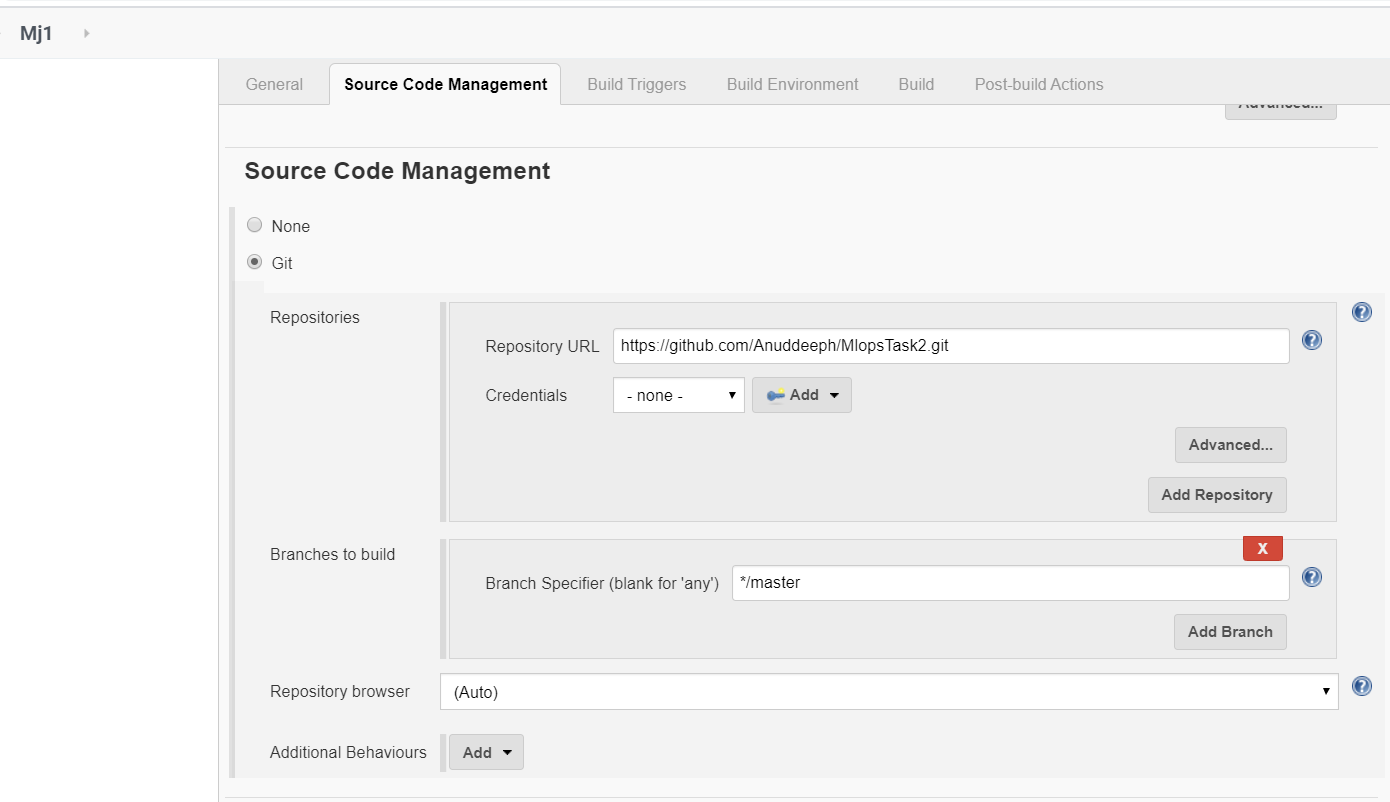
I use MNIST dataset to deploy this model. You can check the code from here...

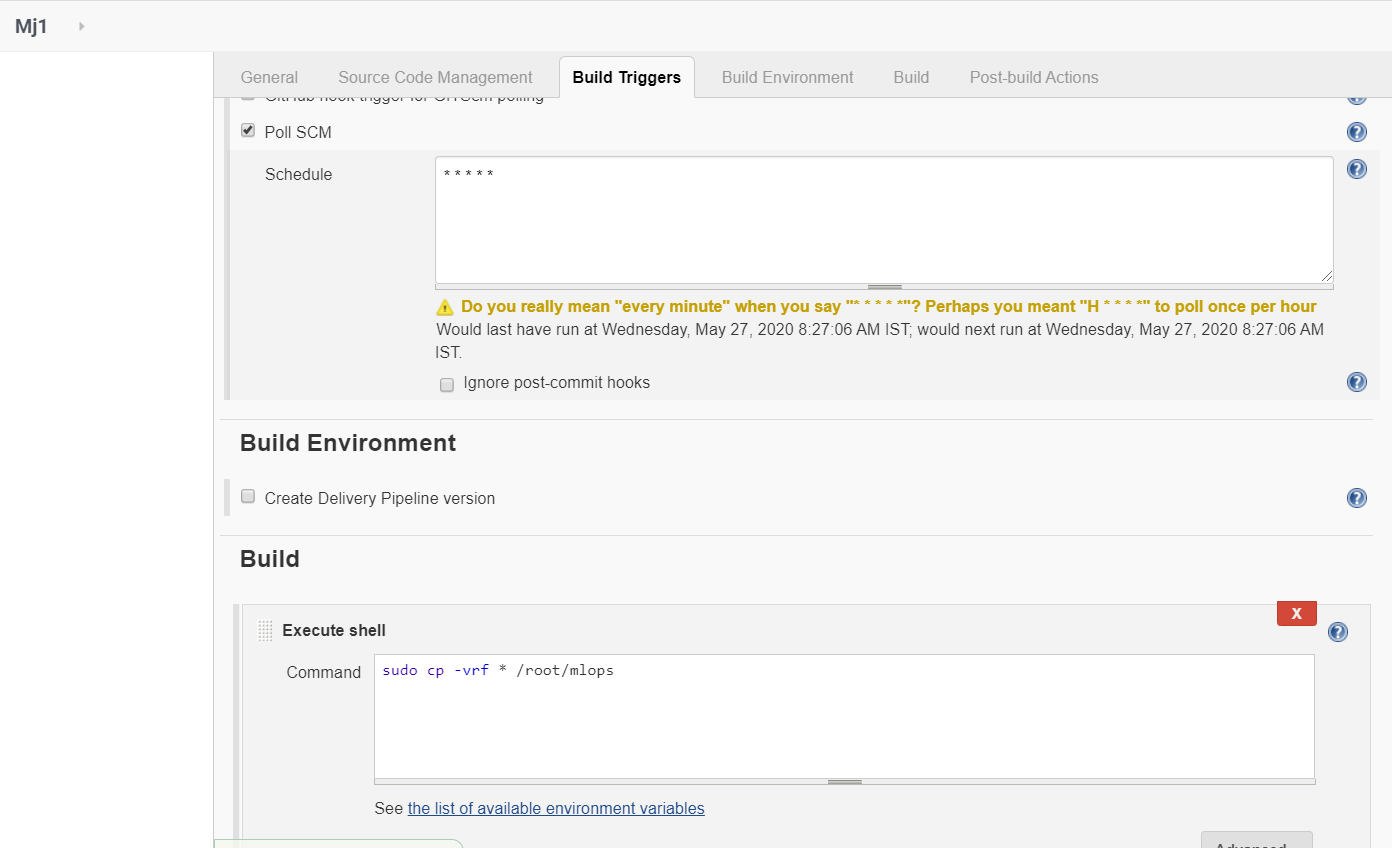
https://github.com/Anuddeeph/MlopsTask2.git



Whenever developer push any code in Github, this job automatically detect and copy in host OS.



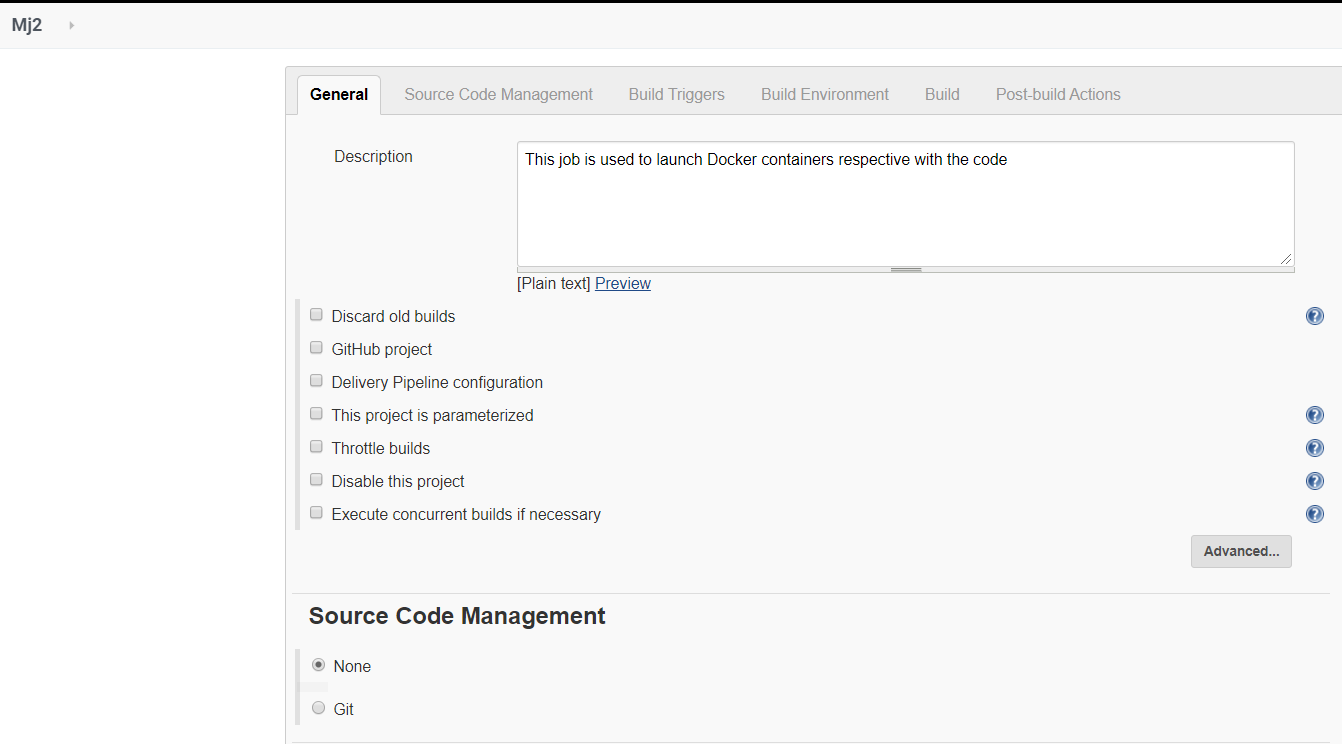


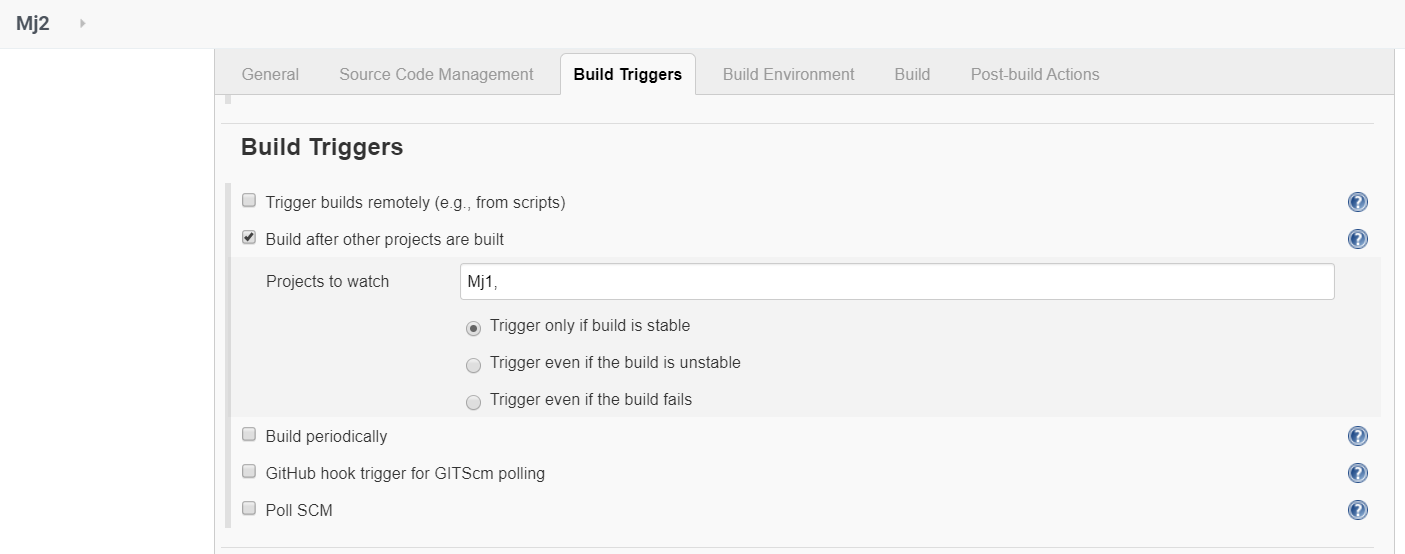


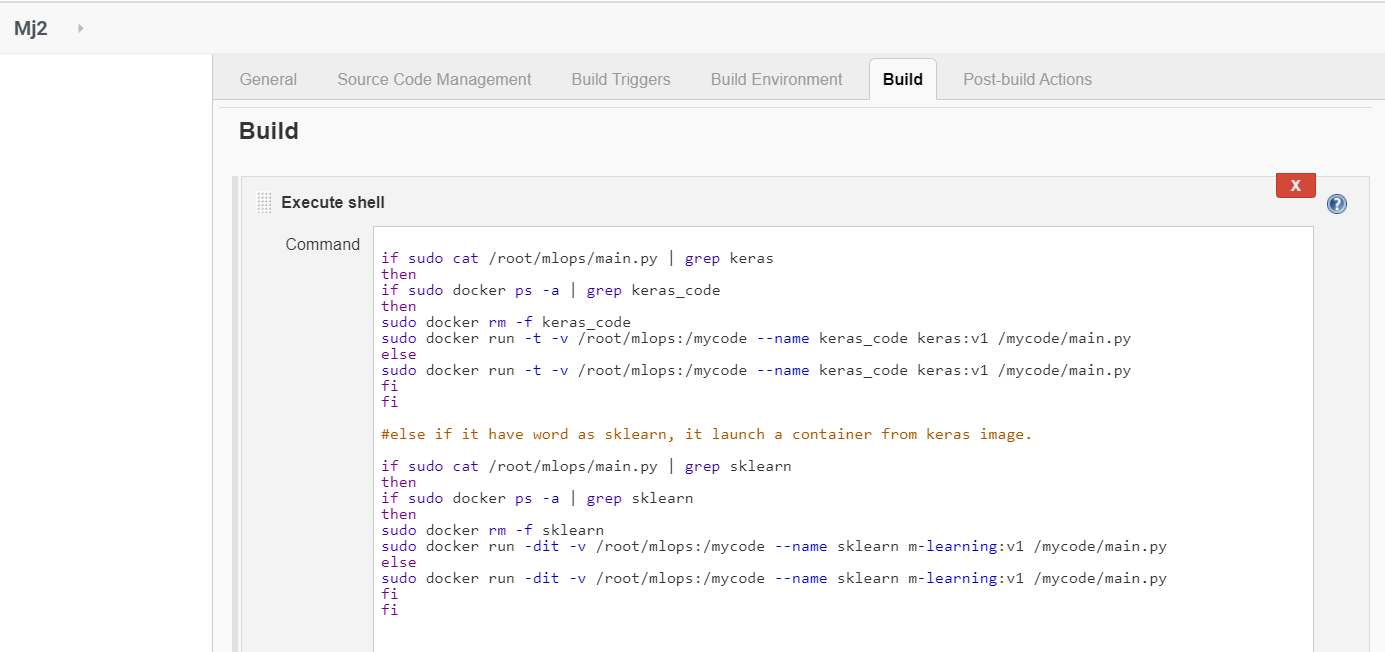
This job creates a directory **mlops** and copy the GitHub code in that directory.

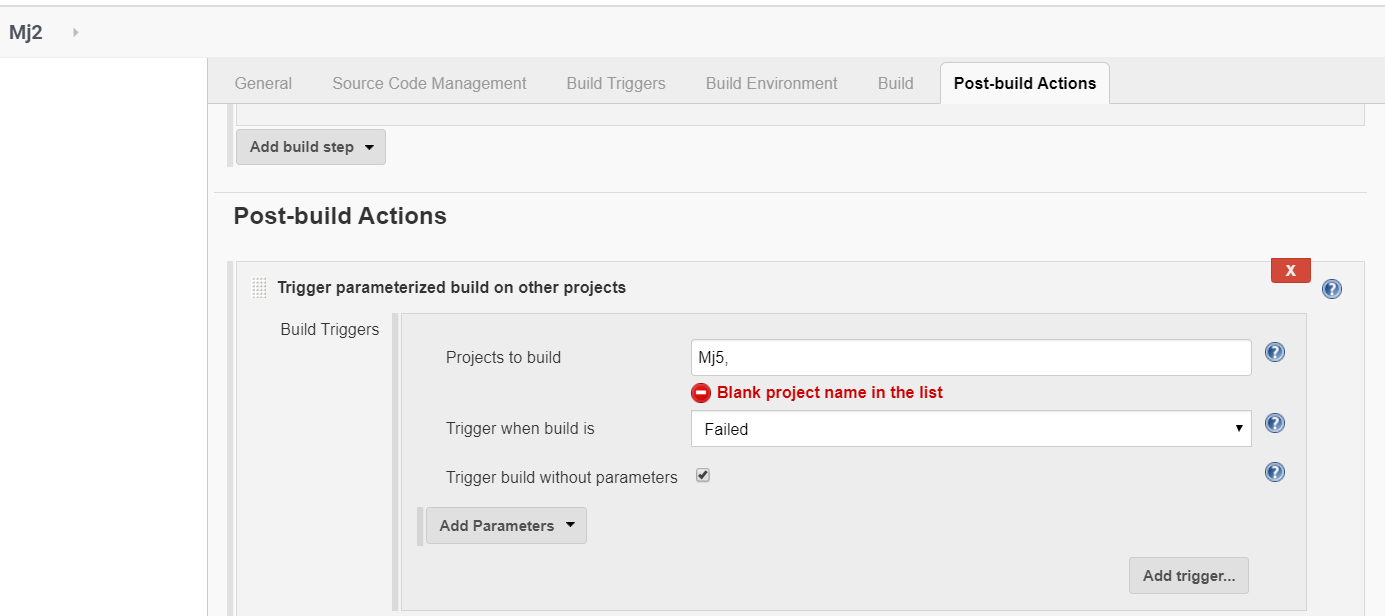
* **Job2(Deploy container for model\_train):**

If my job1 is successfully built, it triggers job2 and launches the container. By looking at the code or program file, this job automatically launch the respective docker container(either Keras os Sklearn).









This is code which I written in this job.

#check the code if it have words related to keras, it launch a container from keras image

if sudo cat /root/mlops/main.py | grep keras

then

if sudo docker ps -a | grep keras\_code

then

sudo docker rm -f keras\_code

sudo docker run -t -v /root/mlops:/mycode --name keras\_code keras:v1 /mycode/main.py

else

sudo docker run -t -v /root/mlops:/mycode --name keras\_code keras:v1 /mycode/main.py

fi

fi

#else if it have word as sklearn, it launch a container from keras image.

if sudo cat /root/mlops/main.py | grep sklearn

then

if sudo docker ps -a | grep sklearn

then

sudo docker rm -f sklearn

sudo docker run -dit -v /root/mlops:/mycode --name sklearn m-learning:v1 /mycode/main.py

else

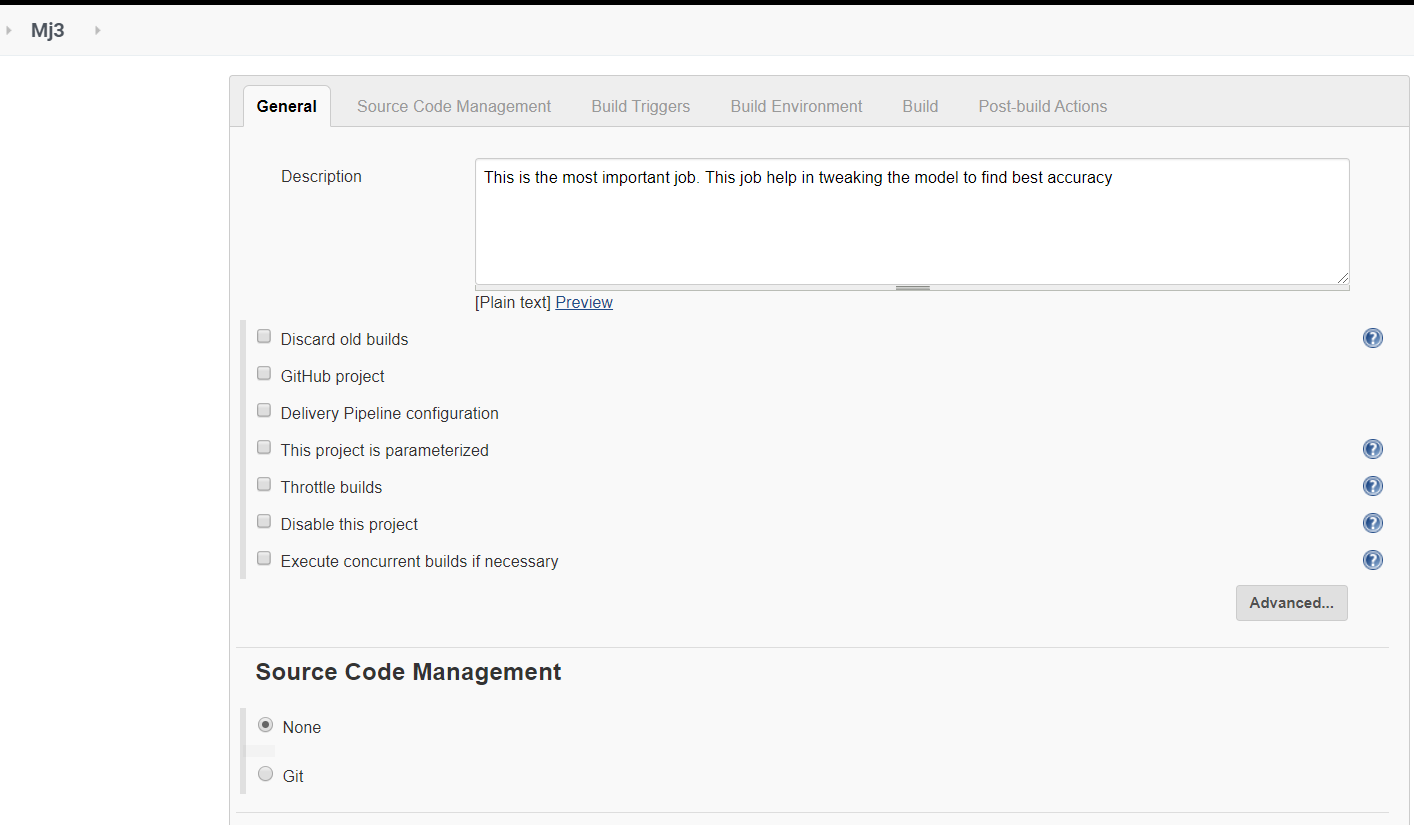
sudo docker run -dit -v /root/mlops:/mycode --name sklearn m-learning:v1 /mycode/main.py

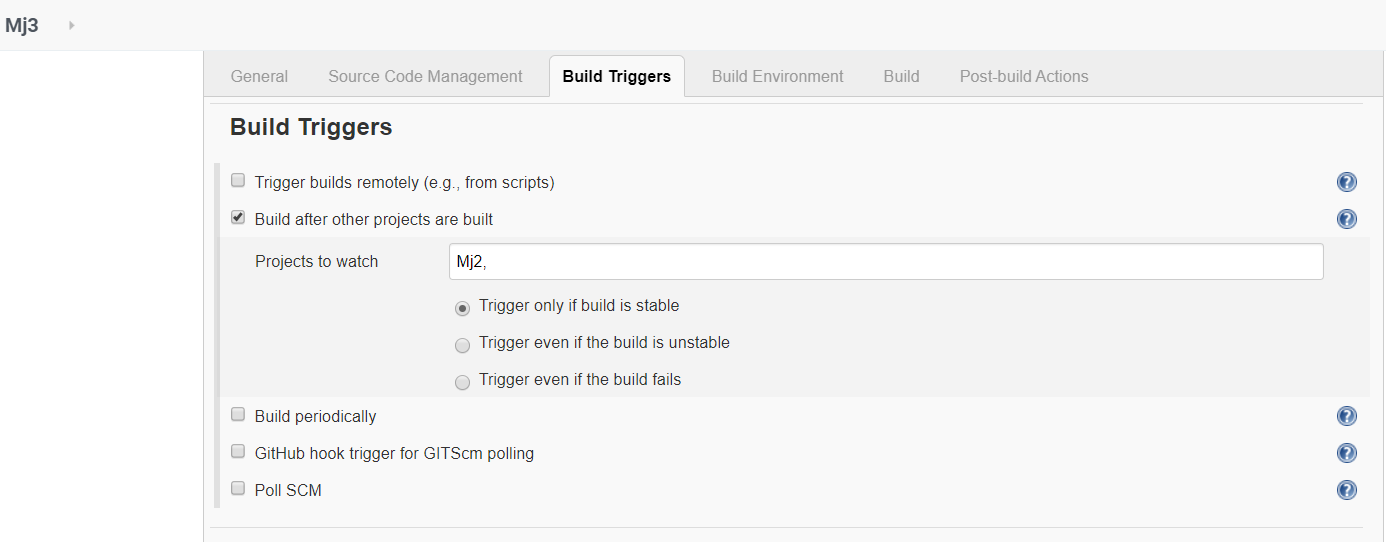
fi

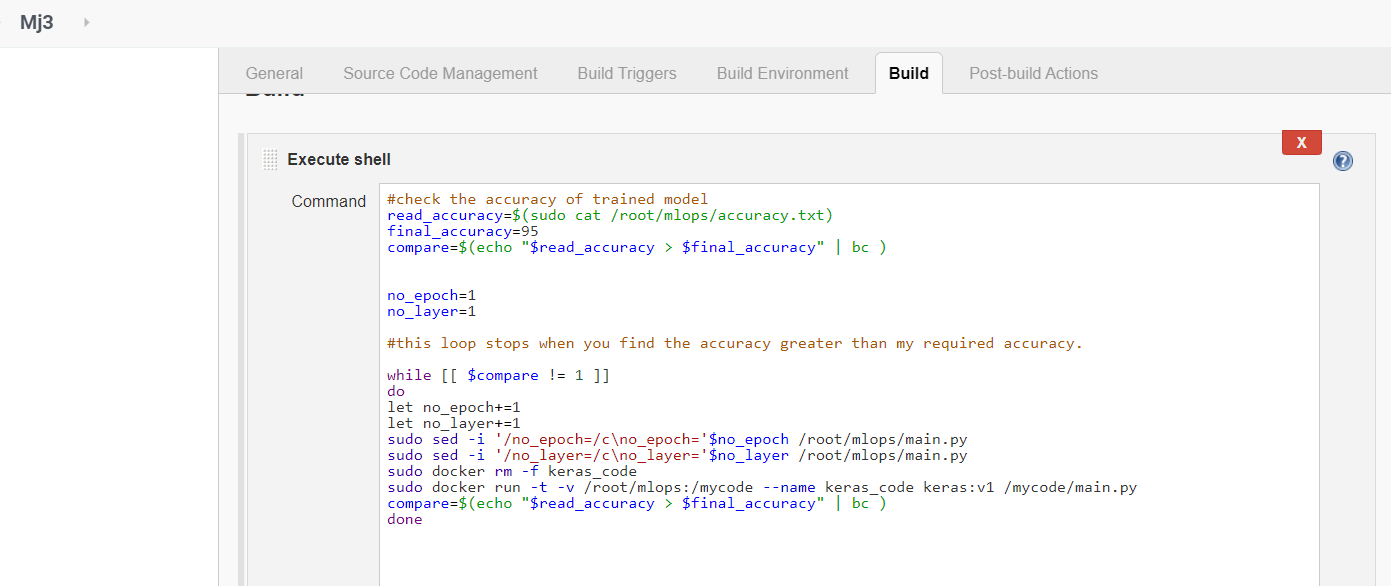
fi

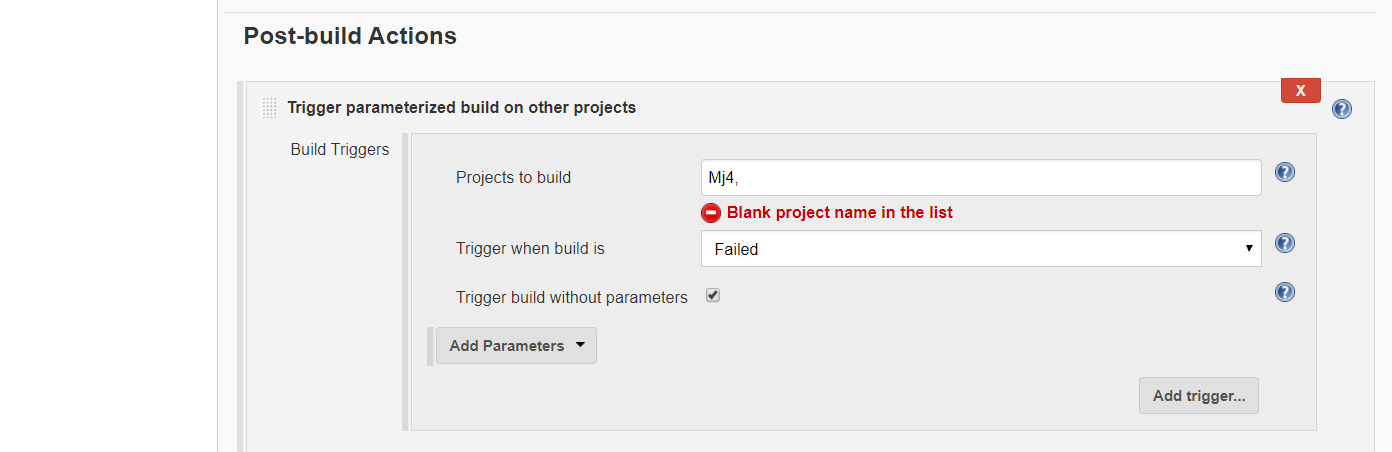
* **Job3(Check the accuracy and tweak the code and again run until it found the required accuracy):**

This is the most important job of whole project. This job check the accuracy of model which I trained in job2 and if accuracy is below from required, this do some changes in code and run again the container to find until required accuracy.









I take number of epoch and number of convolve layer as a parameter increase them by one each time after train the dataset.

#check the accuracy of trained model

read\_accuracy=$(sudo cat /root/mlops/accuracy.txt)

final\_accuracy=95

compare=$(echo "$read\_accuracy > $final\_accuracy" | bc )

no\_epoch=1

no\_layer=1

#this loop stops when you find the accuracy greater than my required accuracy.

while [[ $compare != 1 ]]

do

let no\_epoch+=1

let no\_layer+=1

sudo sed -i '/no\_epoch=/c\no\_epoch='$no\_epoch /root/mlops/main.py

sudo sed -i '/no\_layer=/c\no\_layer='$no\_layer /root/mlops/main.py

sudo docker rm -f keras\_code

sudo docker run -t -v /root/mlops:/mycode --name keras\_code keras:v1 /mycode/main.py

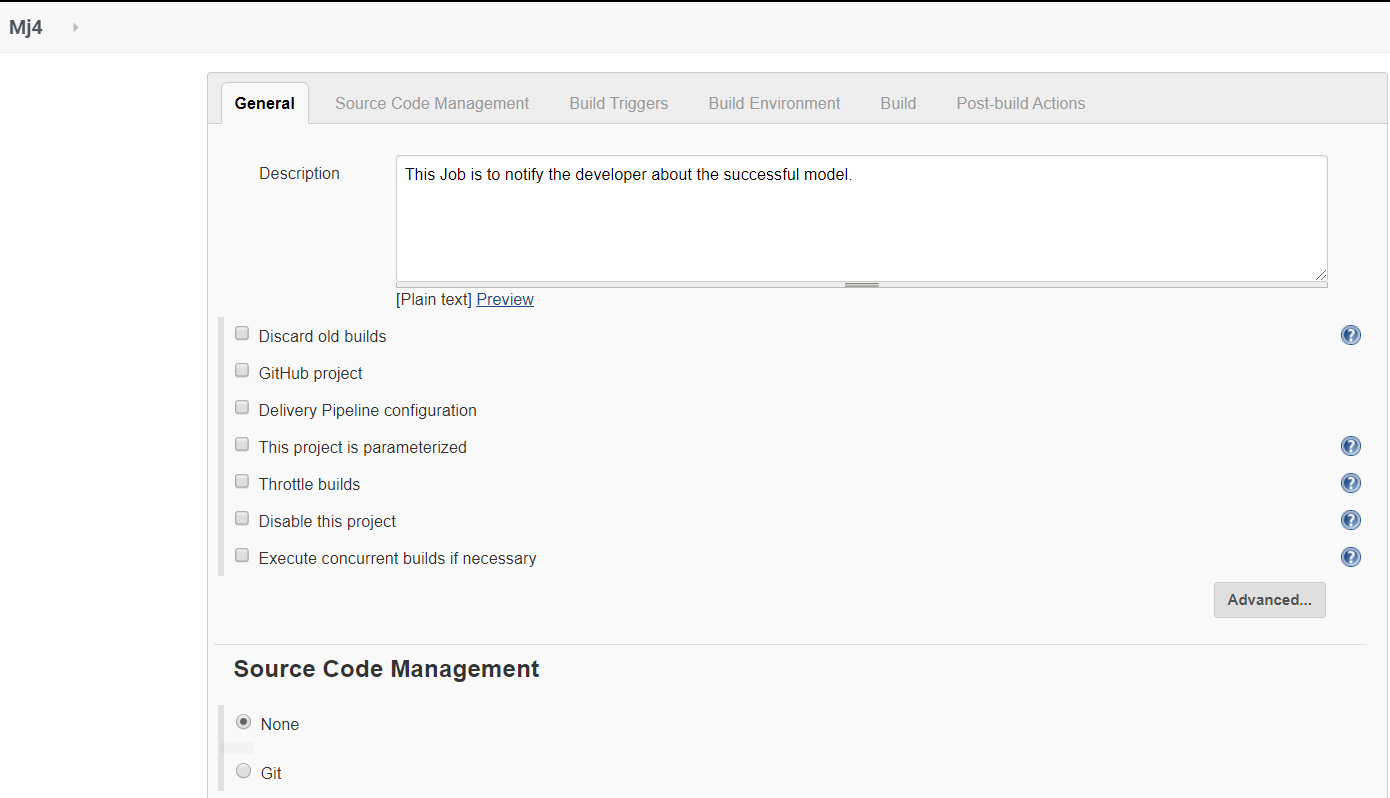
compare=$(echo "$read\_accuracy > $final\_accuracy" | bc )

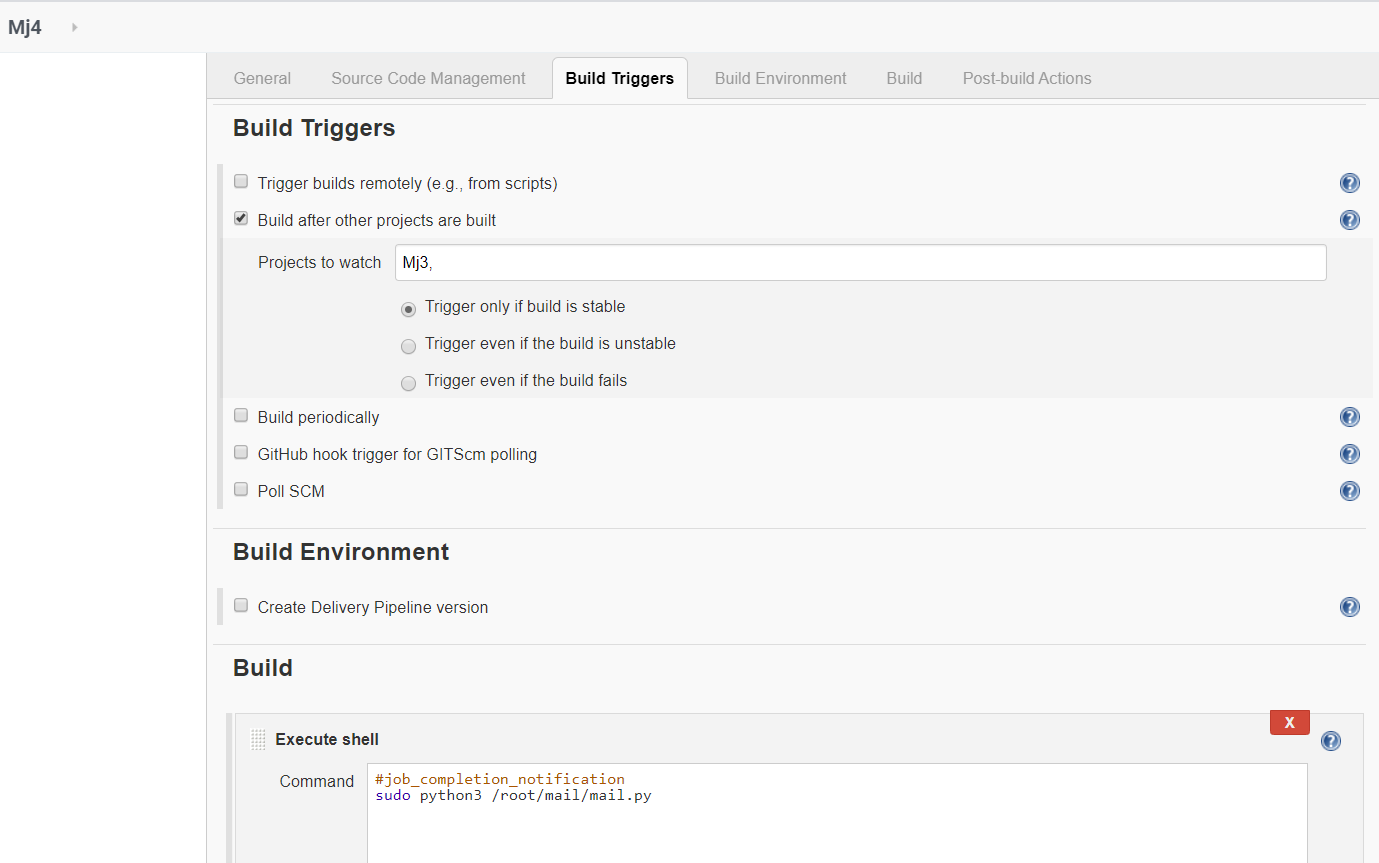
done

If anything goes wrong, it triggers to job4.

* **Job4 (This job sent a mail to the developer for successful train of model):**

After getting the required accuracy, it sent a mail to developer.





#mail.py

import smtplib

# creates SMTP session

s = smtplib.SMTP('smtp.gmail.com', 587)

# start TLS for security

s.starttls()

# Authentication

s.login("sender\_email", "password")

# message to be sent

message = "Hey Developer, Finally we got the model trained. "

# sending the mail

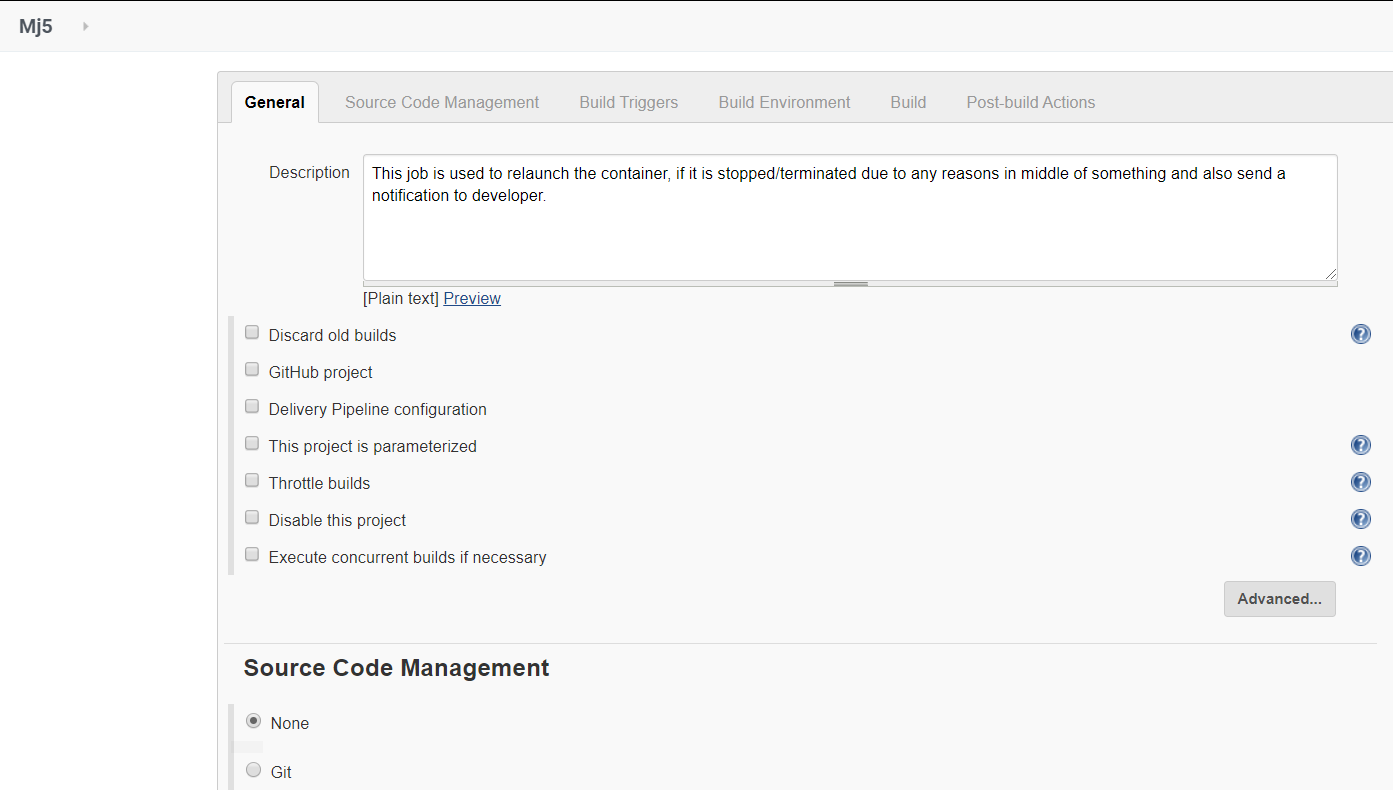
s.sendmail("sender\_mail", "developr\_mail", message)

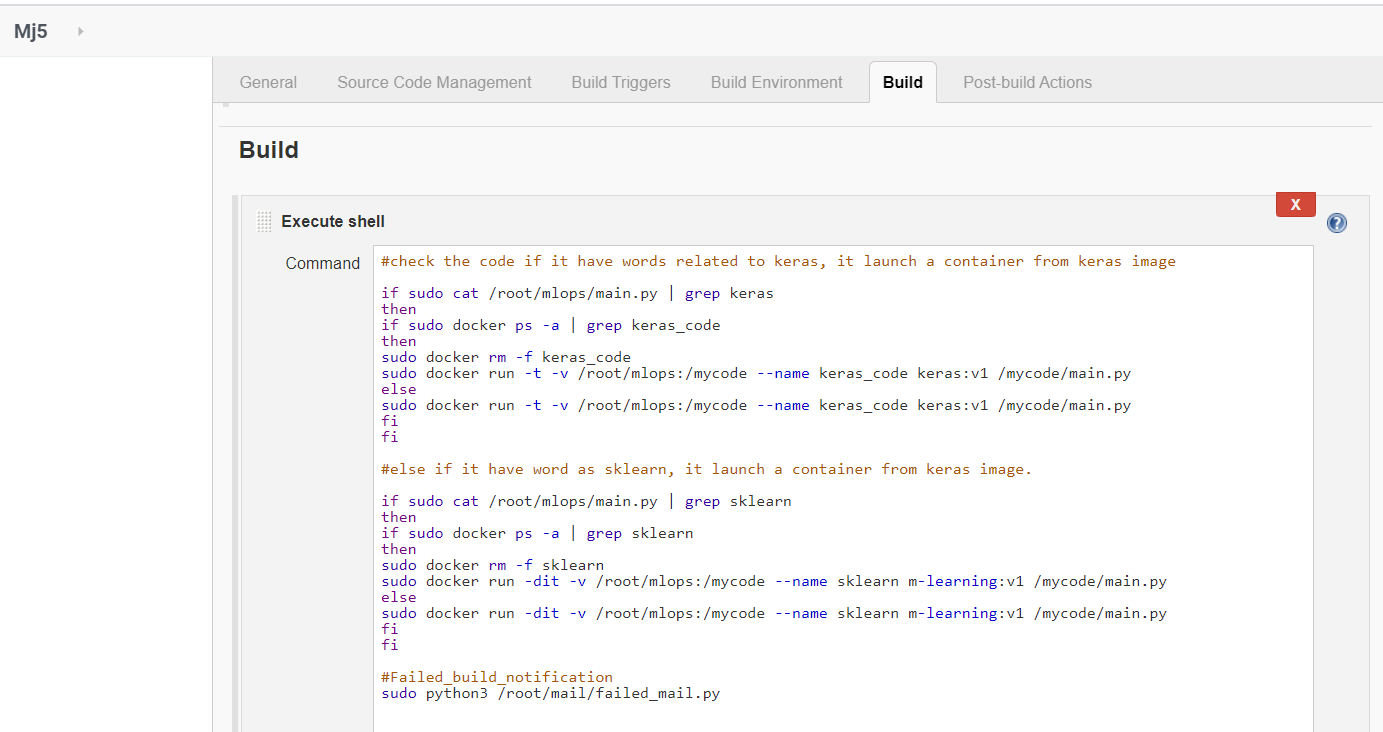
# terminating the session

s.quit()

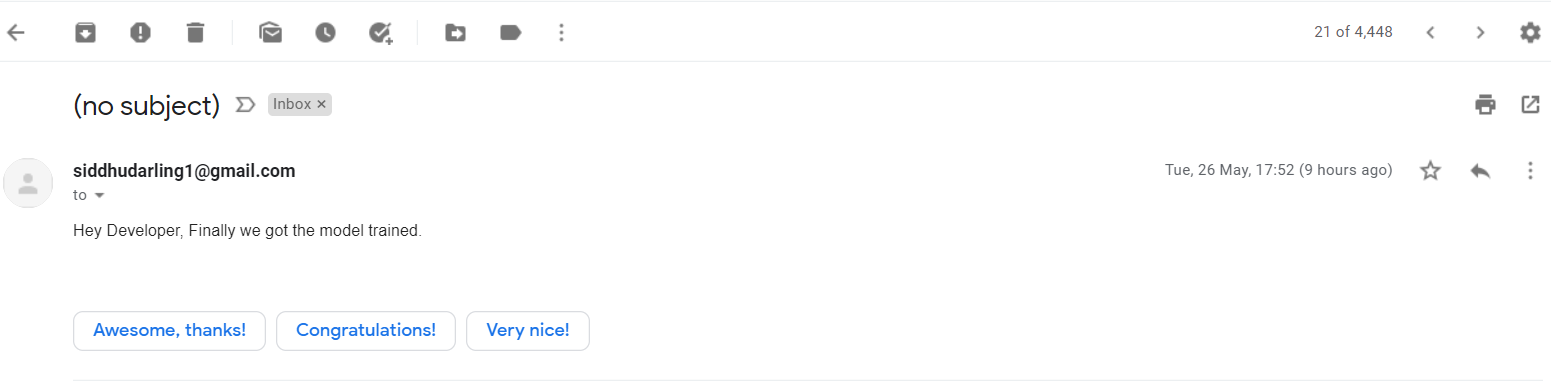
* **Job5(This job is for monitoring the job2 and job3, if any container failed, it rebuild the container):**

This job relaunch the container, if any of container get failed and also sent a email for failure the job.

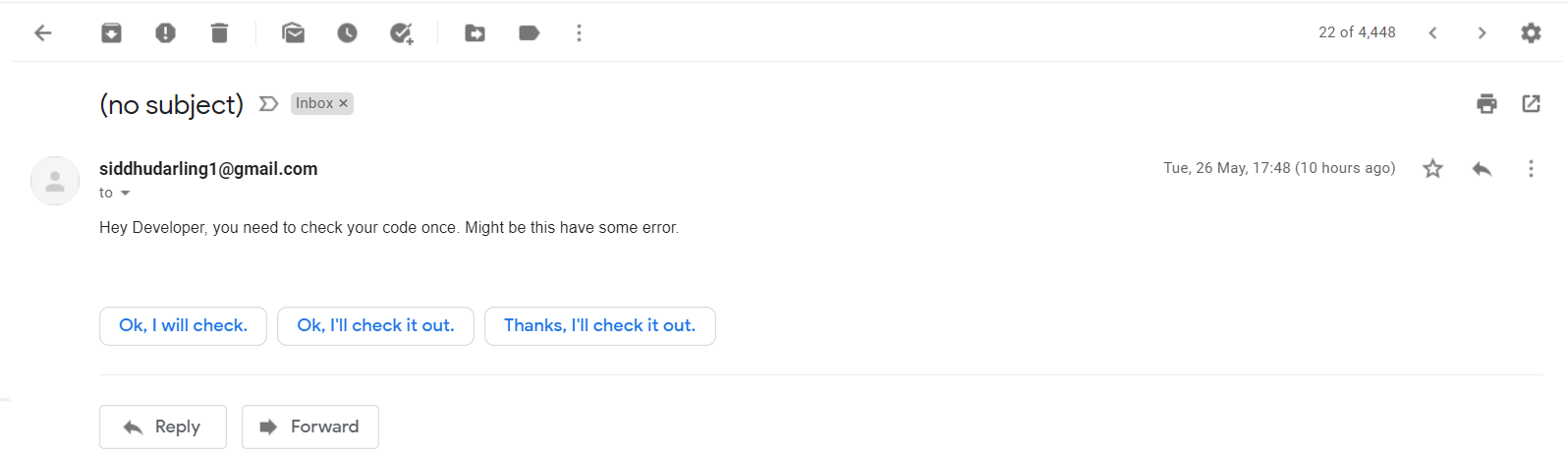




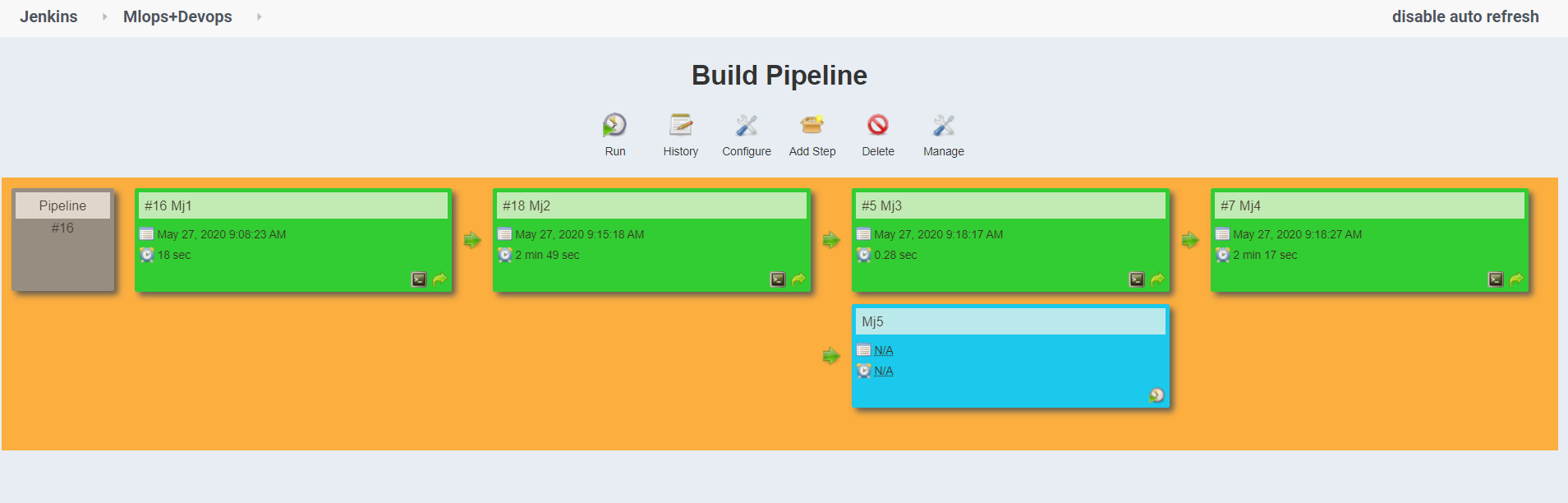
#when code is successful and achieved good accuracy, then the developer get the mail as



##when code is unsuccessful and error in code,then the developer get the mail as



#The Build pipe line is



Here when ever the code fails, then the Mj5 will execute otherwise it stays idle.

For code you can go to the GitHub repo, https://github.com/Anuddeeph/MlopsTask2.git

You can also ping me or comment below if you have any problem in code...

Thank You…