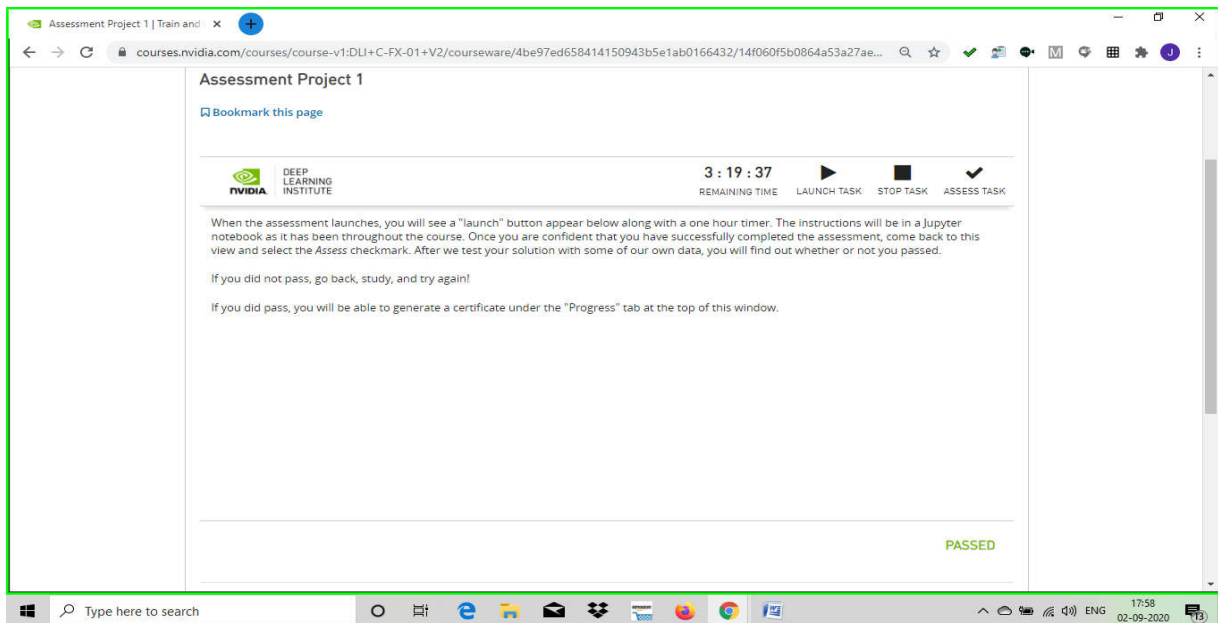
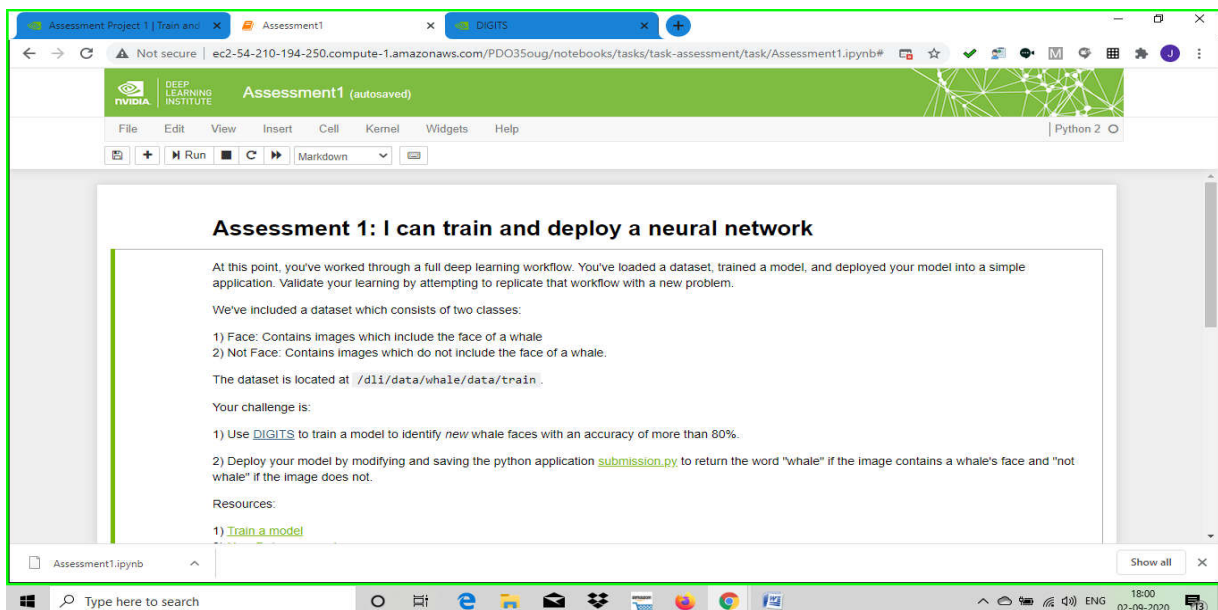


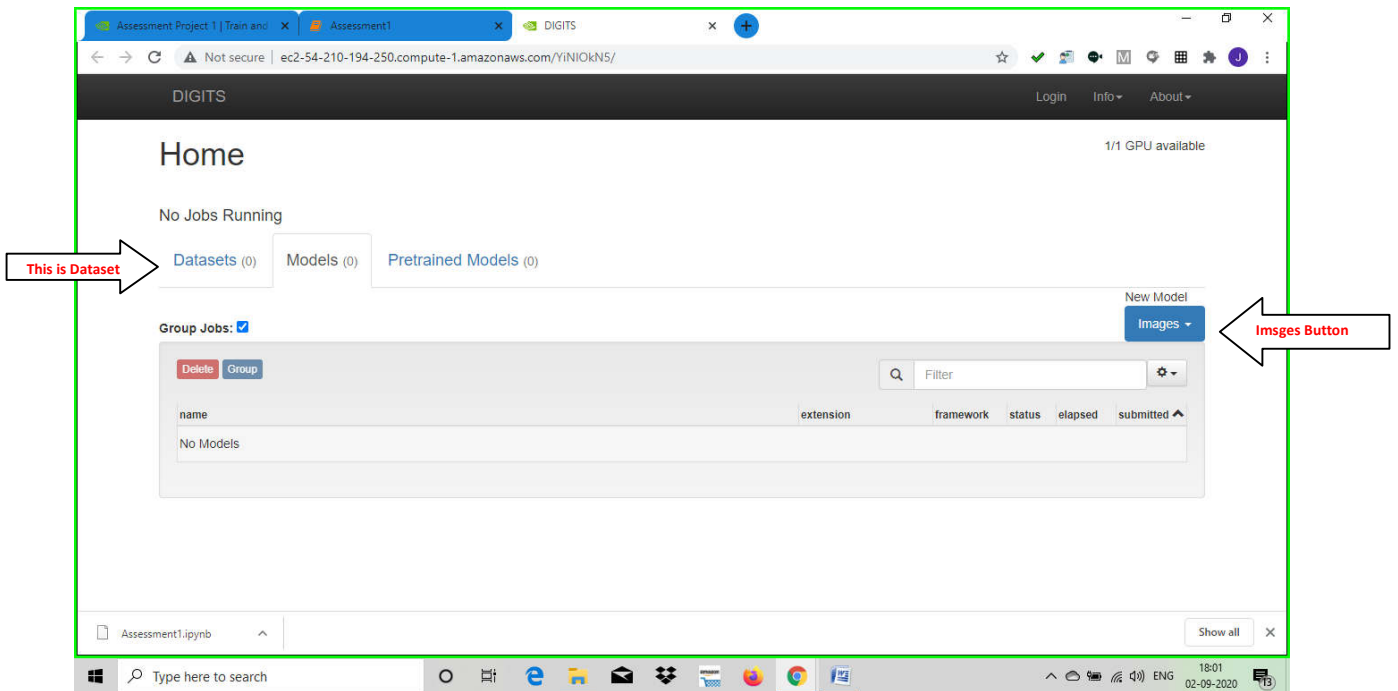
1. login to your nvidia account
2. goto courses tab and click assesment project



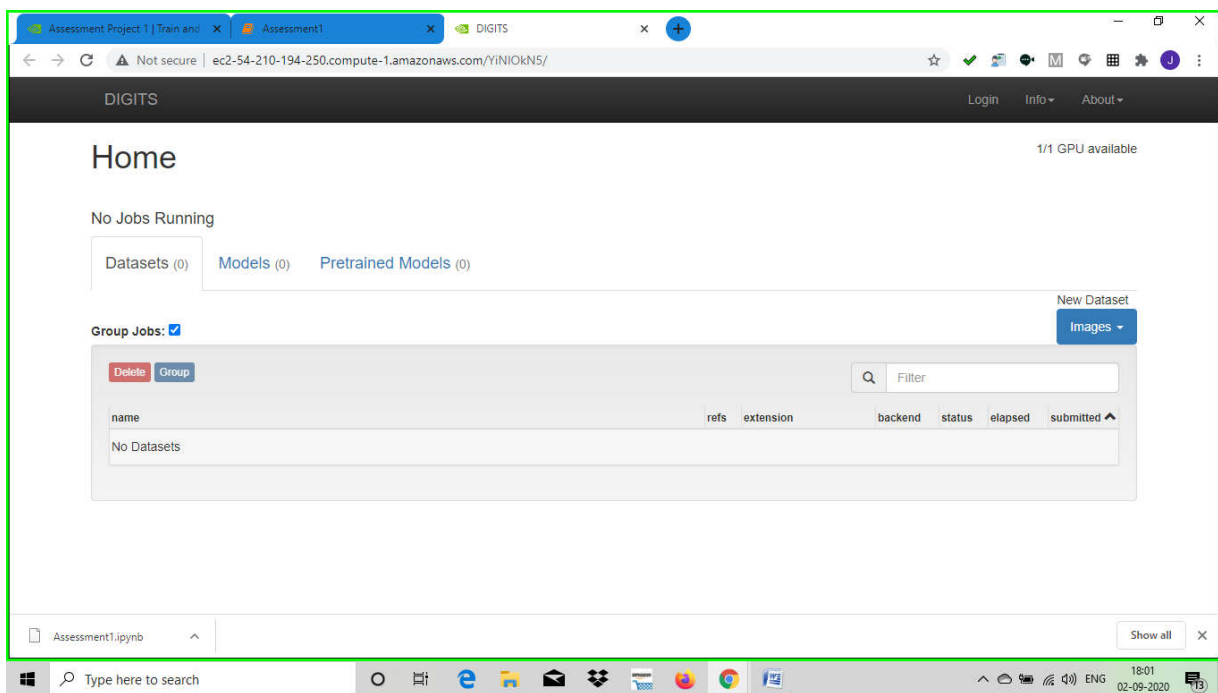
3. start by clicking start button (once the application starts it will begin timer and other things like launch task stop task and asses task)
4. click on launch task

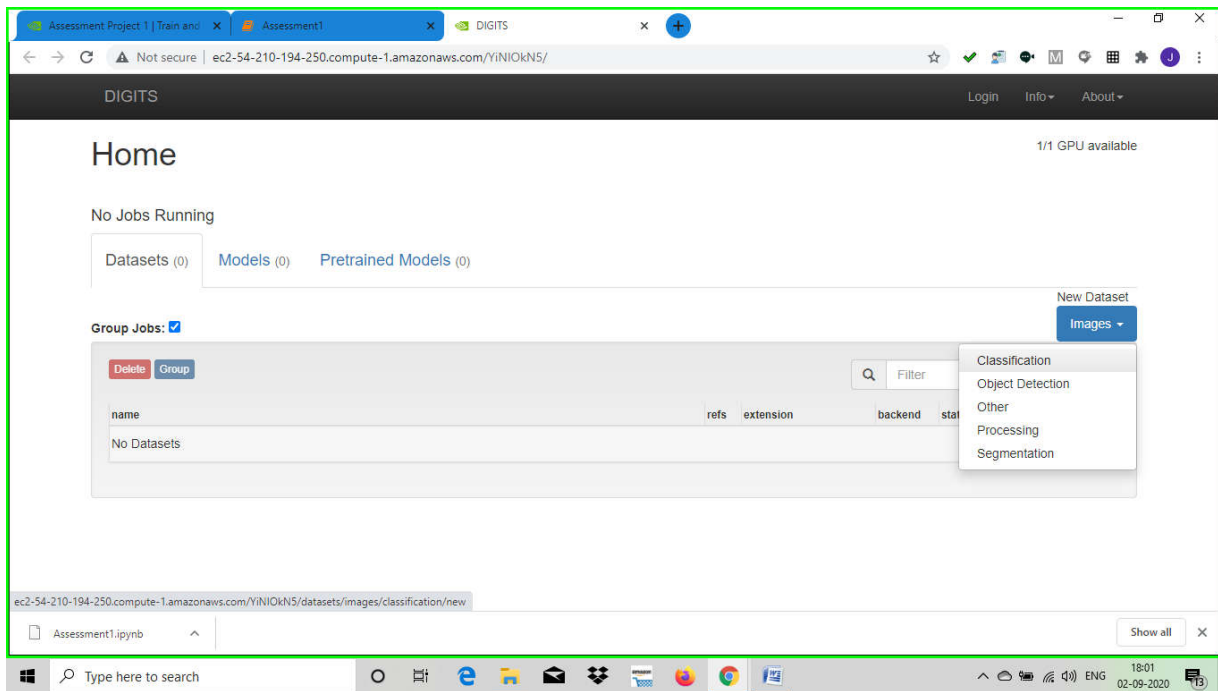


5. it will open assesment (in jupyter)
in that you will find DIGIT and submission.py
open DIGIT



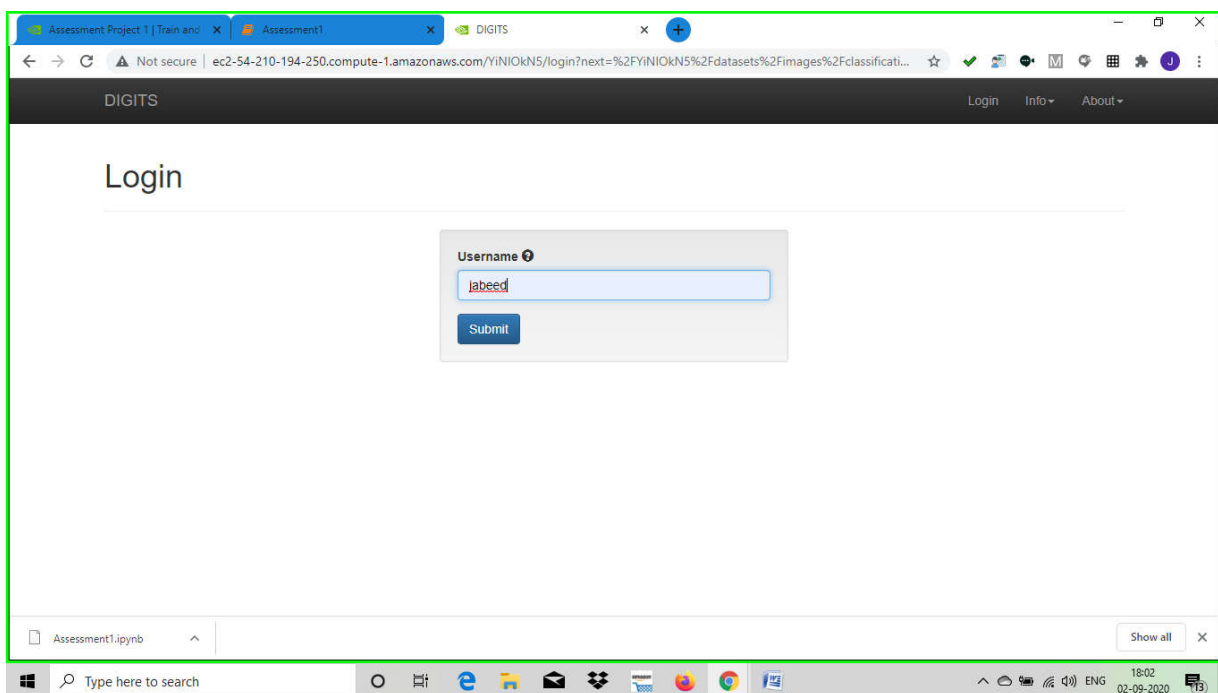
6. click on dataset tab



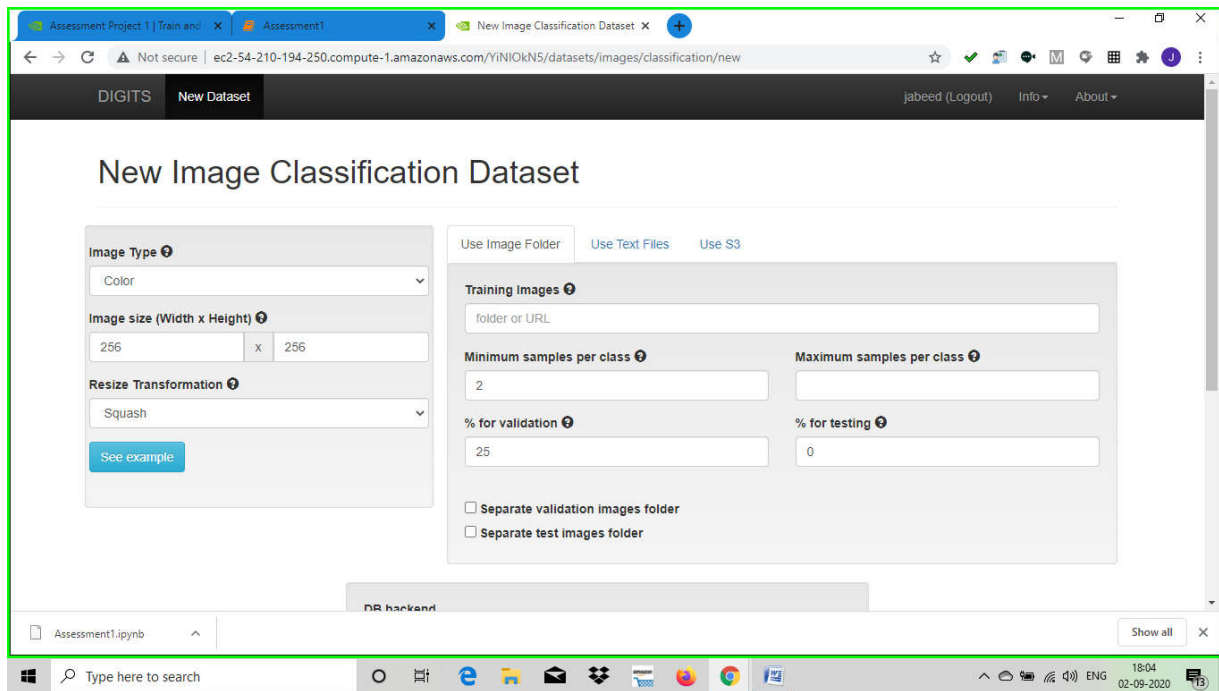


7. click new dataset images button(blue color) and select classification

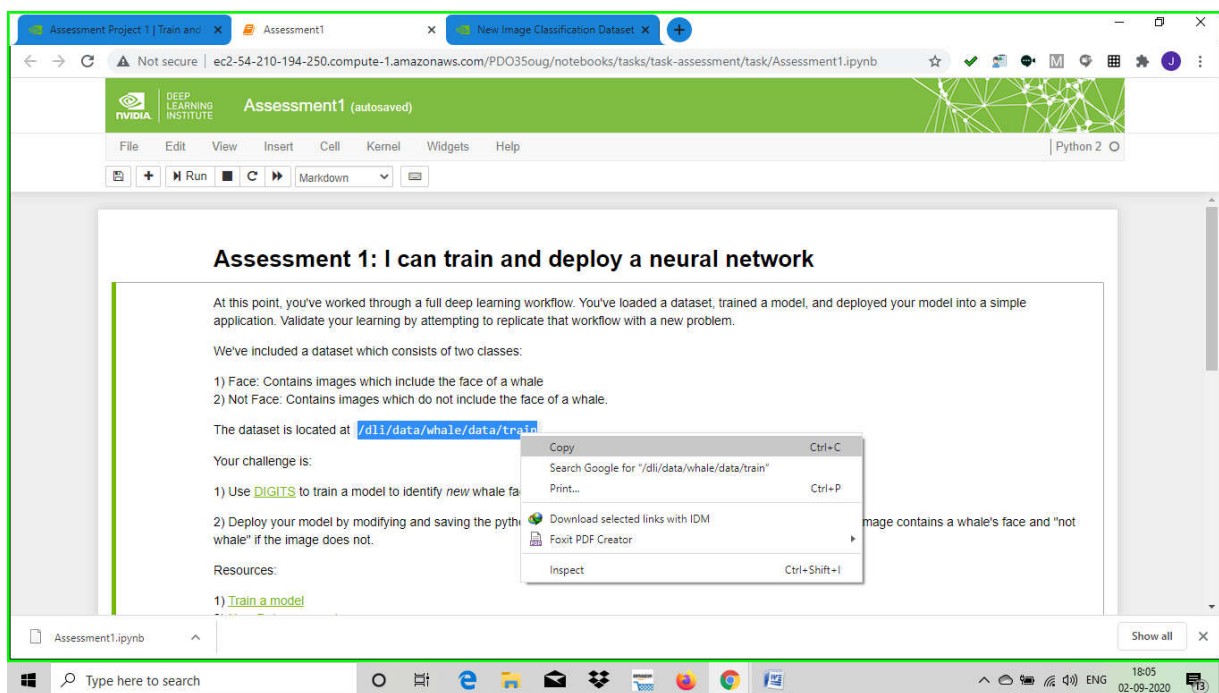
give any user name(ex: Jabeed)



8. click on Submit button



9. When this is opened



10. Goto Assessment Tab(window) and Copy the above text

Assessment Project 1 | Train and x Assessment1 x New Image Classification Dataset x

Not secure | ec2-54-210-194-250.compute-1.amazonaws.com/YiNIokN5/datasets/images/classification/new

DIGITS New Dataset jabeed (Logout) Info About

New Image Classification Dataset

Image Type

Color

Image size (Width x Height)

256 x 256

Resize Transformation

Squash

See example

Use Image Folder Use Text Files Use S3

Training Images

/dlldata/whale/data/train

Minimum samples per class

2

Maximum samples per class

0

% for validation

25

% for testing

0

☐ Separate validation images folder

☐ Separate test images folder

DB backend

Assessment1.ipynb

Show all

11. Come back to Digit tab and paste the above link in Training Images path

☐ Separate validation images folder

☐ Separate test images folder

DB backend

LMDB

Image Encoding

PNG (lossless)

Group Name

Dataset Name

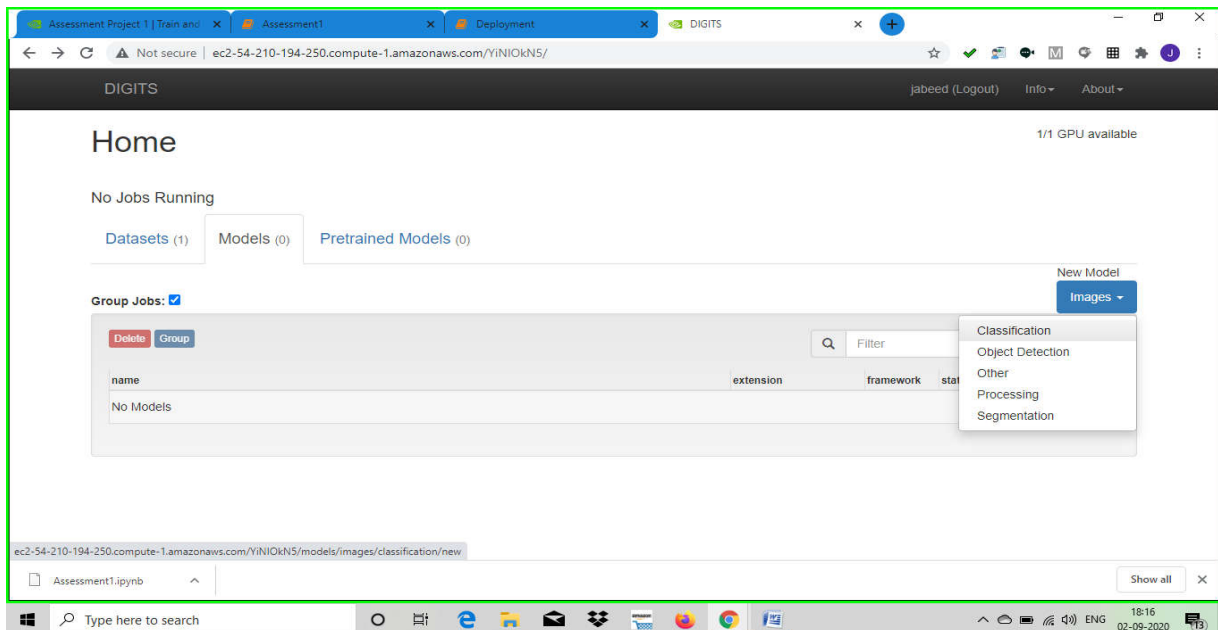
whale

Create

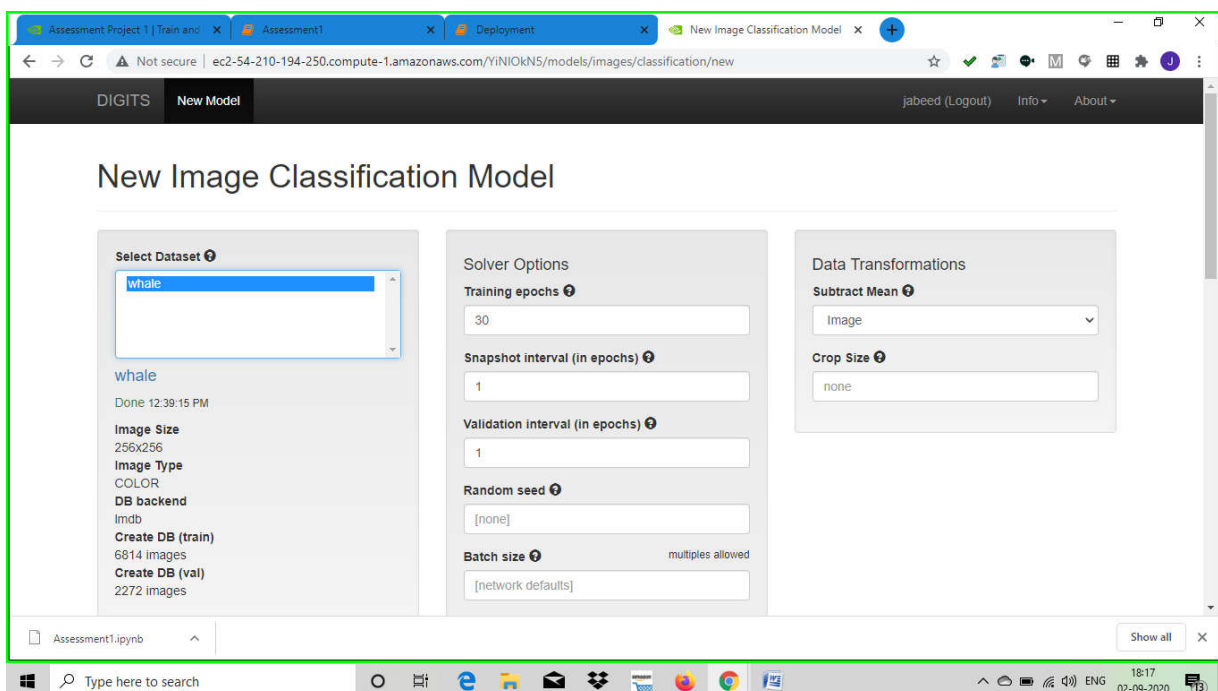
Assessment1.ipynb

Show all

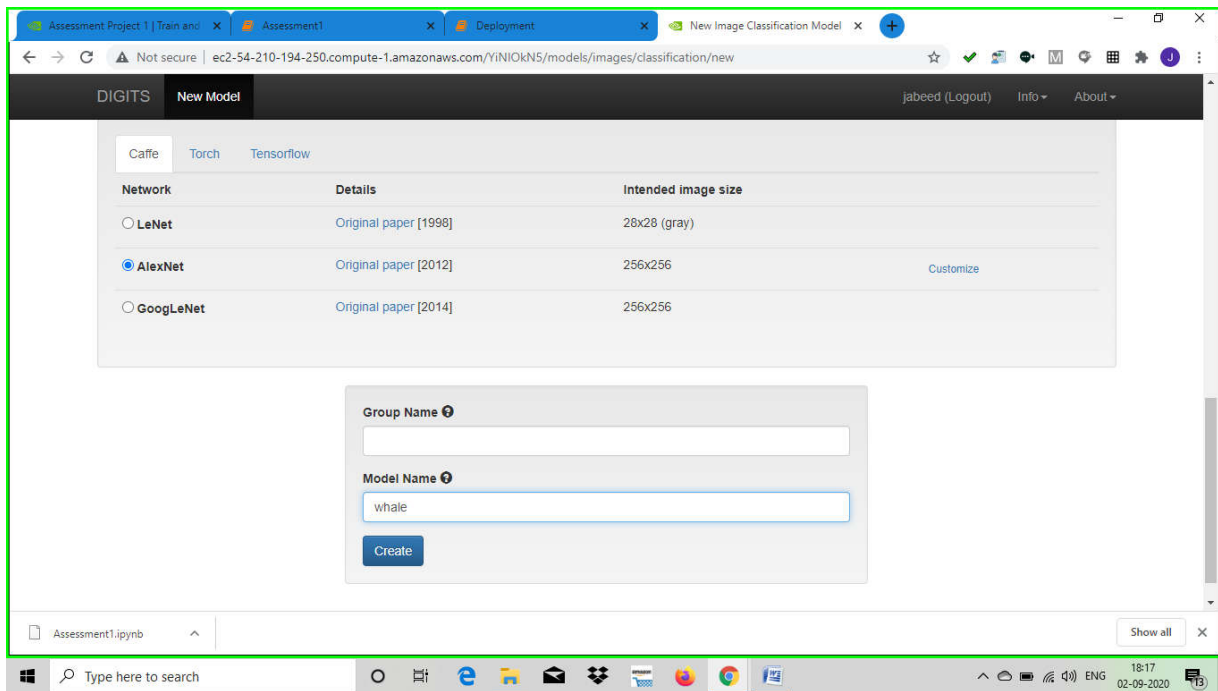
12. Scroll to bottom of the same page (of step 11) and give Dataset name as whale and click on Create.



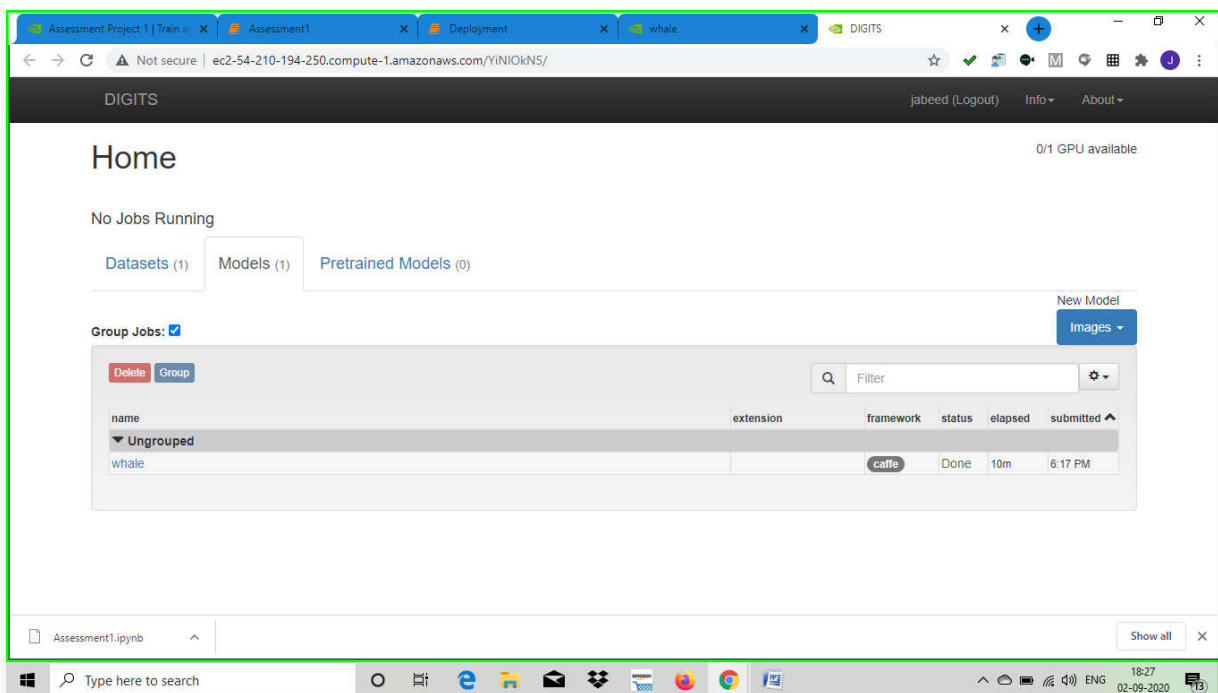
14. Now again open DIGITS(ensure your Dataset is having one dataset whale in that) and select Models tab and create new classification as shown above.



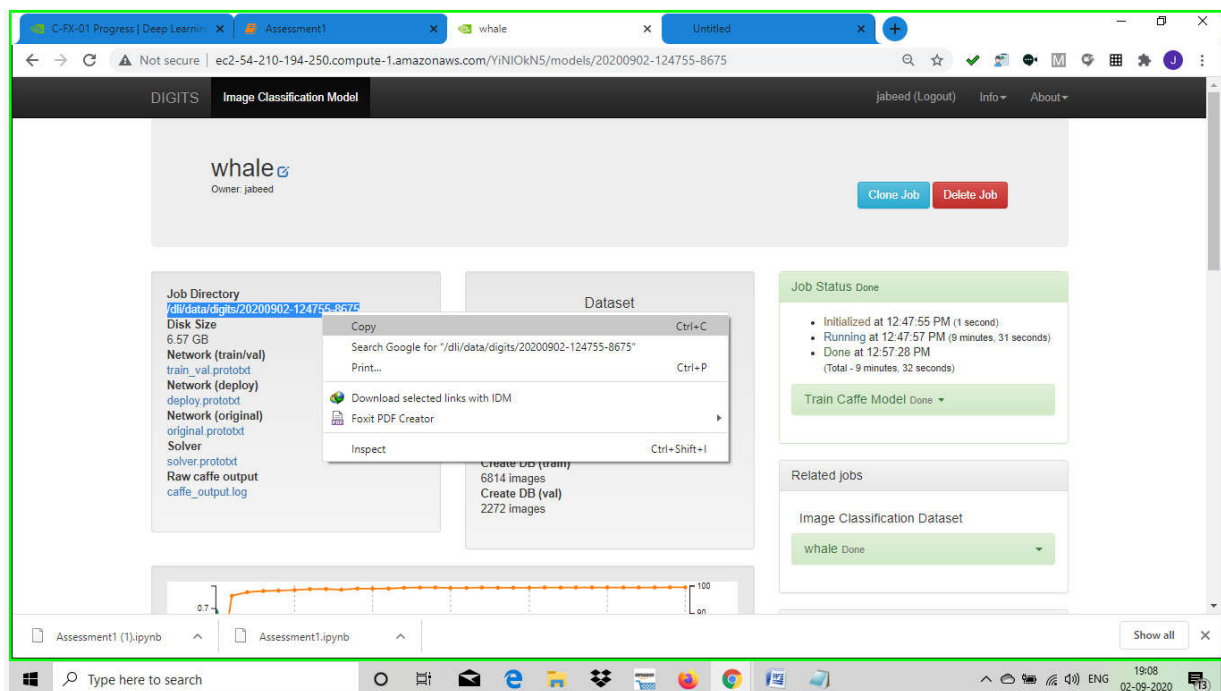
15. Select dataset whale and don't change any default values.



16. Scroll to bottom of (step 15) select AlexNet and give model name whale and click on Create button as shown above.

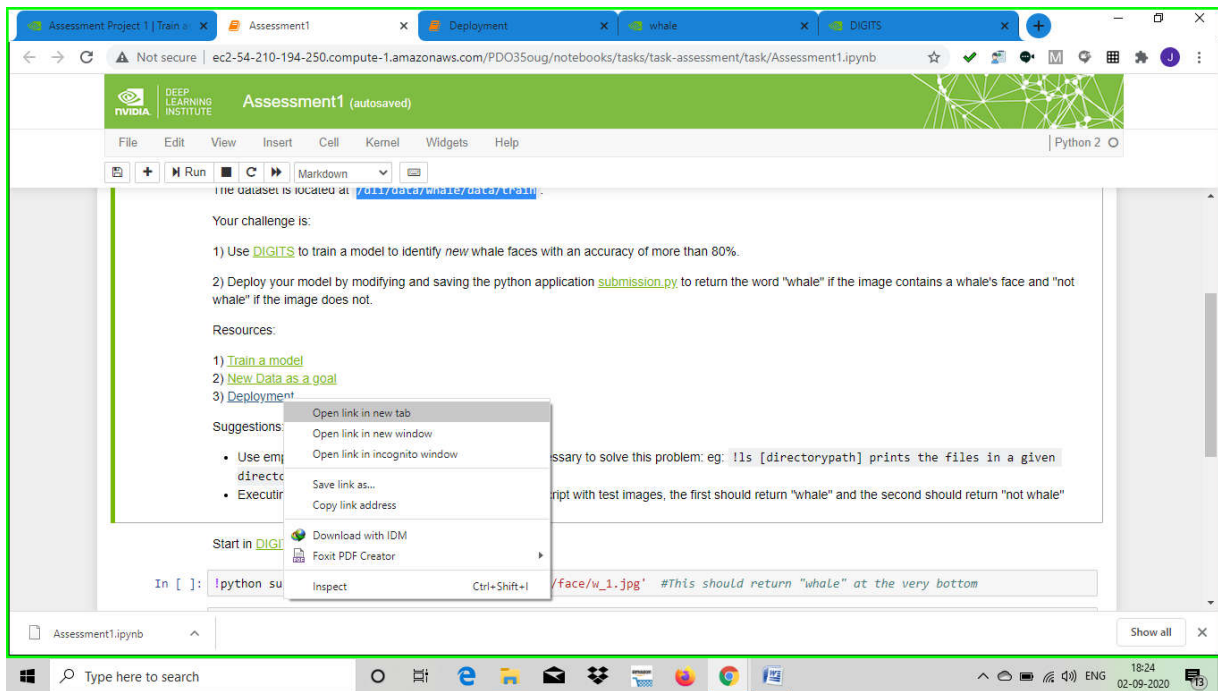


17. If can't see progress in the same page open digits in another tab or another window of Browser Wait till Model Creation

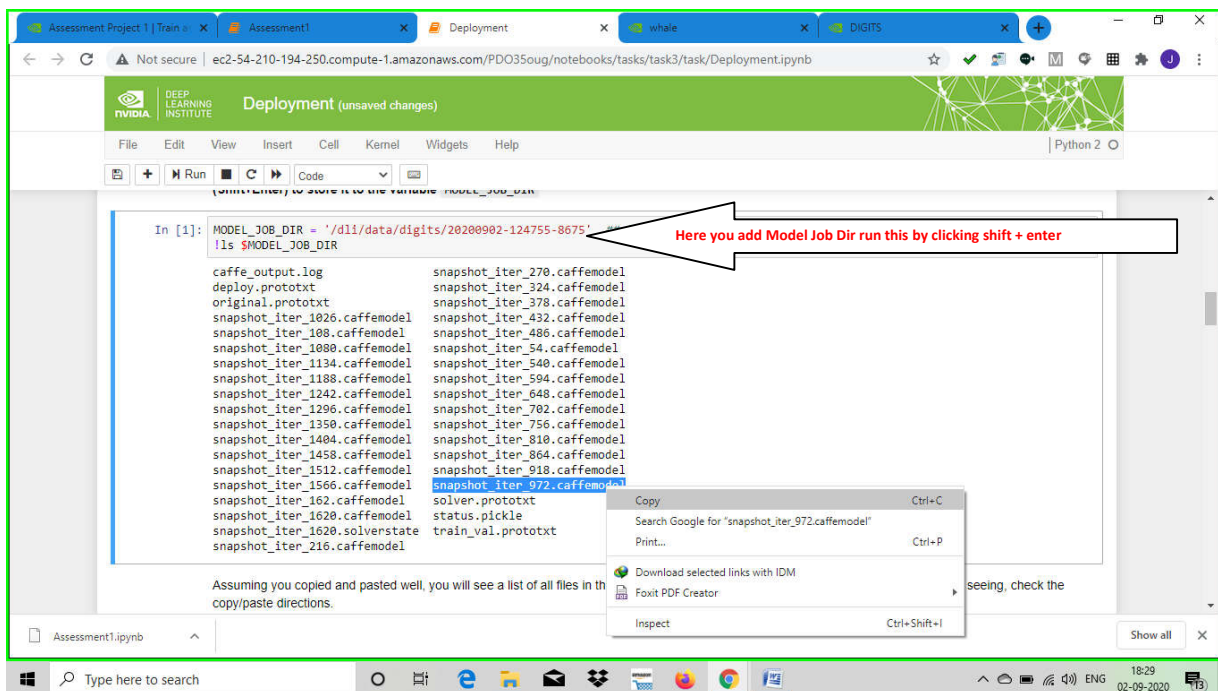


18. Wait till 100% (in the Meanwhile Open Notepad and paste this Model Job Dir: /dli/data/digits/20200902-124755-8675 shown as below which we need in submission.py)





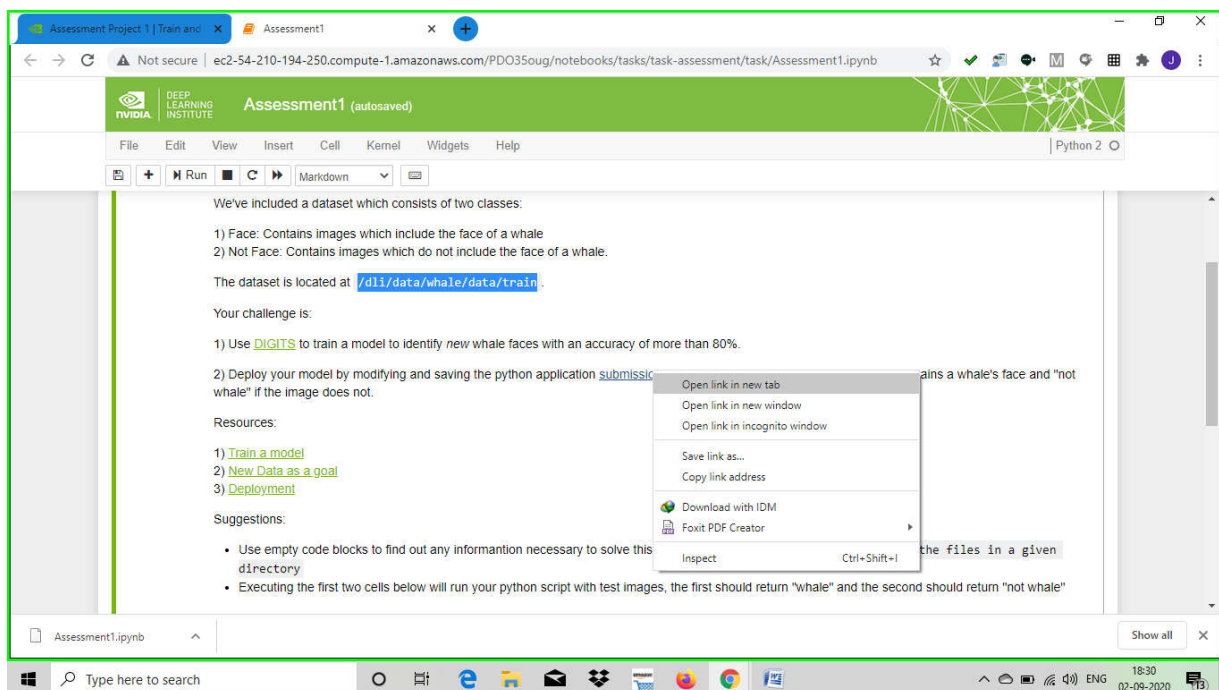
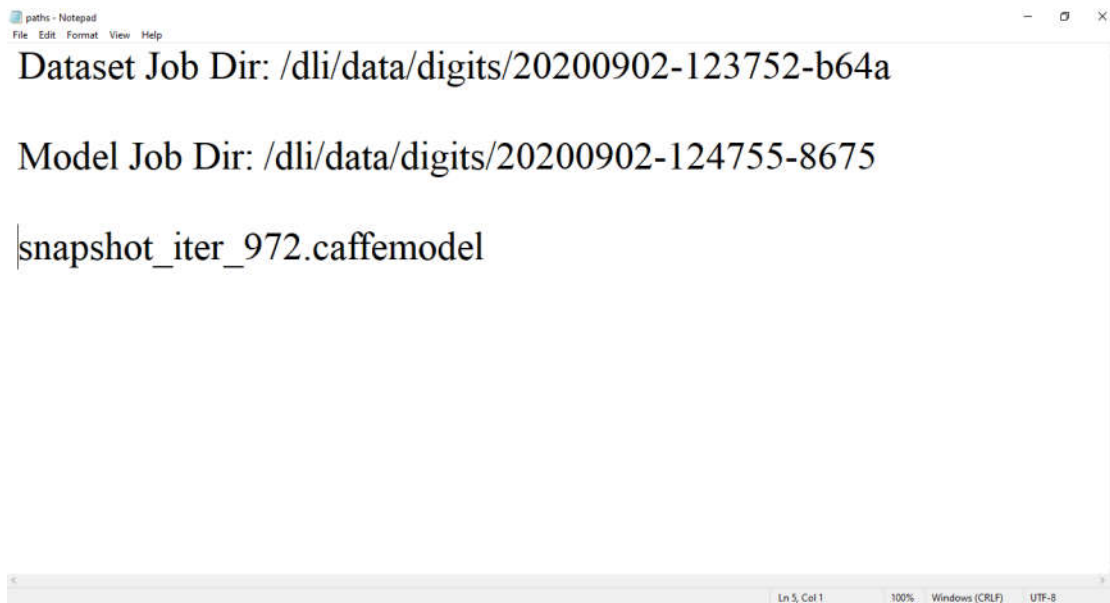
19. After model creation open Deployment as shown above.



20. Add the copied Job directory in the Jupiter editor as shown above and run it

After getting the above screen copy the last instance of snapshot in this case it is snapshot_iter_972.caffemodel

21. And paste the same in the previously open Notepad



22. Now come to Assessment 1 window and click like above submission.py in new tab

```
1 import caffe
2 import cv2
3 import sys
4
5 def deploy(img_path):
6
7     caffe.set_mode_gpu()
8
9     # Initialize the Caffe model using the model trained in DIGITS. Which two files constitute your trained model?
10    net = caffe.Classifier('##REPLACE WITH ARCHITECTURE FILE##', '##REPLACE WITH WEIGHTS FILE##',
11                          channel_swap=(2,1,0),
12                          raw_scale=255,
13                          image_dims=(256, 256))
14
15    # Create an input that the network expects. This is different for each project, so don't worry about the exact steps, but find the
16    # dataset job directory to show you know that whatever preprocessing is done during training must also be done during deployment.
17    input_image = caffe.io.load_image(img_path)
18    input_image = cv2.resize(input_image, (256,256))
19    mean_image = caffe.io.load_image('##REPLACE WITH DATASET JOB DIRECTORY##/mean.jpg')
20    input_image = input_image - mean_image
21
22    # Make prediction. What is the function and the input to the function needed to make a prediction?
23    prediction = net.##REPLACE WITH THE FUNCTION THAT RETURNS THE OUTPUT OF THE NETWORK##(##REPLACE WITH THE INPUT TO THE FUNCTION##)
24
25    # Create an output that is useful to a user. What is the condition that should return "whale" vs. "not whale"?
26    if ##REPLACE WITH THE CONDITION THAT WOULD MAKE THE FUNCTION RETURN WHALE##:
27        return "whale"
```

23. In the above file there are **5 changes**(**this is default file if you wish you can even use submission.py file from the parent folder also in such case perform 1st and 3rd change only[just 2 changes at the required line in this file]**)

1st change :

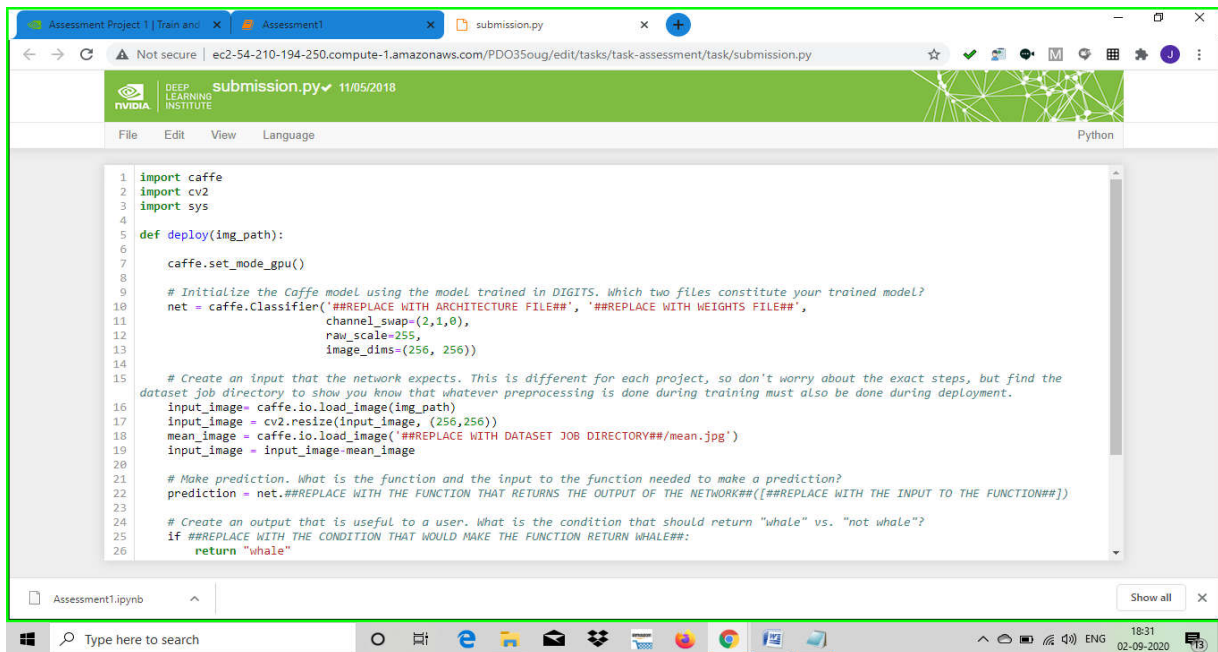
add these lines of code after 7th line(caffe.set_mode_gpu()) add the following

#the line below this has to be corrected with your code at this instance of time when I was creating Model job dir is /dli/data/digits/20200902-124755-8675' and snapshot_iter from NOTEPAD file

```
MODEL_JOB_DIR='/dli/data/digits/20200902-124755-8675'
```

```
ARCH = MODEL_JOB_DIR + '/' + 'deploy.prototxt'
```

```
WEIGHTS = MODEL_JOB_DIR + '/' + 'snapshot_iter_972.caffemodel'
```

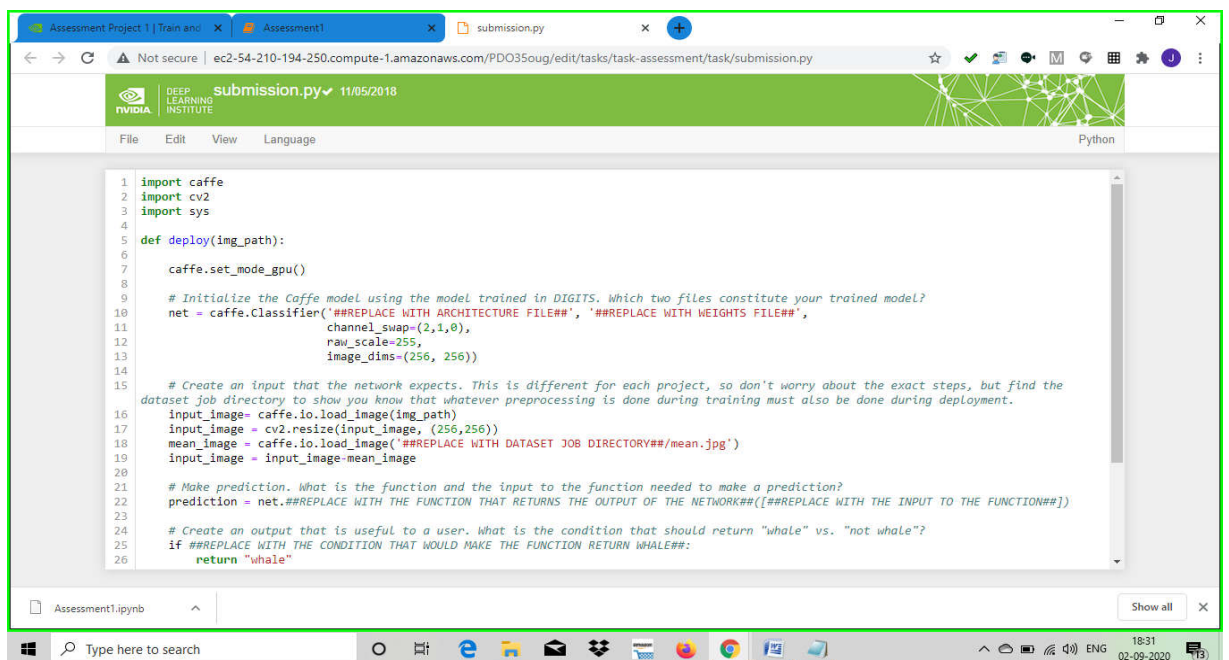


```
1 import caffe
2 import cv2
3 import sys
4
5 def deploy(img_path):
6
7     caffe.set_mode_gpu()
8
9     # Initialize the Caffe model using the model trained in DIGITS. Which two files constitute your trained model?
10    net = caffe.Classifier('##REPLACE WITH ARCHITECTURE FILE##', '##REPLACE WITH WEIGHTS FILE##',
11                          channel_swap=(2,1,0),
12                          raw_scale=255,
13                          image_dims=(256, 256))
14
15    # Create an input that the network expects. This is different for each project, so don't worry about the exact steps, but find the
16    # dataset job directory to show you know that whatever preprocessing is done during training must also be done during deployment.
17    input_image = caffe.io.load_image(img_path)
18    input_image = cv2.resize(input_image, (256,256))
19    mean_image = caffe.io.load_image('##REPLACE WITH DATASET JOB DIRECTORY##/mean.jpg')
20    input_image = input_image - mean_image
21
22    # Make prediction. What is the function and the input to the function needed to make a prediction?
23    prediction = net.##REPLACE WITH THE FUNCTION THAT RETURNS THE OUTPUT OF THE NETWORK##([##REPLACE WITH THE INPUT TO THE FUNCTION##])
24
25    # Create an output that is useful to a user. What is the condition that should return "whale" vs. "not whale"?
26    if ##REPLACE WITH THE CONDITION THAT WOULD MAKE THE FUNCTION RETURN WHALE##:
27        return "whale"
```

2nd Change:

In the above file at line no: 10 replace it with this code

net = caffe.Classifier(ARCH, WEIGHTS,



```
1 import caffe
2 import cv2
3 import sys
4
5 def deploy(img_path):
6
7     caffe.set_mode_gpu()
8
9     # Initialize the Caffe model using the model trained in DIGITS. Which two files constitute your trained model?
10    net = caffe.Classifier(ARCH, WEIGHTS,
11                          channel_swap=(2,1,0),
12                          raw_scale=255,
13                          image_dims=(256, 256))
14
15    # Create an input that the network expects. This is different for each project, so don't worry about the exact steps, but find the
16    # dataset job directory to show you know that whatever preprocessing is done during training must also be done during deployment.
17    input_image = caffe.io.load_image(img_path)
18    input_image = cv2.resize(input_image, (256,256))
19    mean_image = caffe.io.load_image('##REPLACE WITH DATASET JOB DIRECTORY##/mean.jpg')
20    input_image = input_image - mean_image
21
22    # Make prediction. What is the function and the input to the function needed to make a prediction?
23    prediction = net.##REPLACE WITH THE FUNCTION THAT RETURNS THE OUTPUT OF THE NETWORK##([##REPLACE WITH THE INPUT TO THE FUNCTION##])
24
25    # Create an output that is useful to a user. What is the condition that should return "whale" vs. "not whale"?
26    if ##REPLACE WITH THE CONDITION THAT WOULD MAKE THE FUNCTION RETURN WHALE##:
27        return "whale"
```

3rd Change:

In the above file at line no 18 add this

#the line below this has to be corrected with your code at this instance of time when I was creating
Dataset job dir is /dli/data/digits/20200902-123752-b64a

mean_image = caffe.io.load_image('/dli/data/digits/20200902-123752-b64a/mean.jpg')


```
1 import caffe
2 import cv2
3 import sys
4
5 def deploy(img_path):
6
7     caffe.set_mode_gpu()
8
9     # Initialize the Caffe model using the model trained in DIGITS. Which two files constitute your trained model?
10    net = caffe.Classifier('##REPLACE WITH ARCHITECTURE FILE##', '##REPLACE WITH WEIGHTS FILE##',
11                          channel_swap=(2,1,0),
12                          raw_scale=255,
13                          image_dims=(256, 256))
14
15    # Create an input that the network expects. This is different for each project, so don't worry about the exact steps, but find the
16    # dataset job directory to show you know that whatever preprocessing is done during training must also be done during deployment.
17    input_image = caffe.io.load_image(img_path)
18    input_image = cv2.resize(input_image, (256,256))
19    mean_image = caffe.io.load_image('##REPLACE WITH DATASET JOB DIRECTORY##/mean.jpg')
20    input_image = input_image - mean_image
21
22    # Make prediction. What is the function and the input to the function needed to make a prediction?
23    prediction = net.##REPLACE WITH THE FUNCTION THAT RETURNS THE OUTPUT OF THE NETWORK##(##REPLACE WITH THE INPUT TO THE FUNCTION##)
24
25    # Create an output that is useful to a user. What is the condition that should return "whale" vs. "not whale"?
26    if ##REPLACE WITH THE CONDITION THAT WOULD MAKE THE FUNCTION RETURN WHALE##:
27        return "whale"
```

4th Change:

In the above at line no 22 add this

Make prediction. What is the function and the input to the function needed to make a prediction?

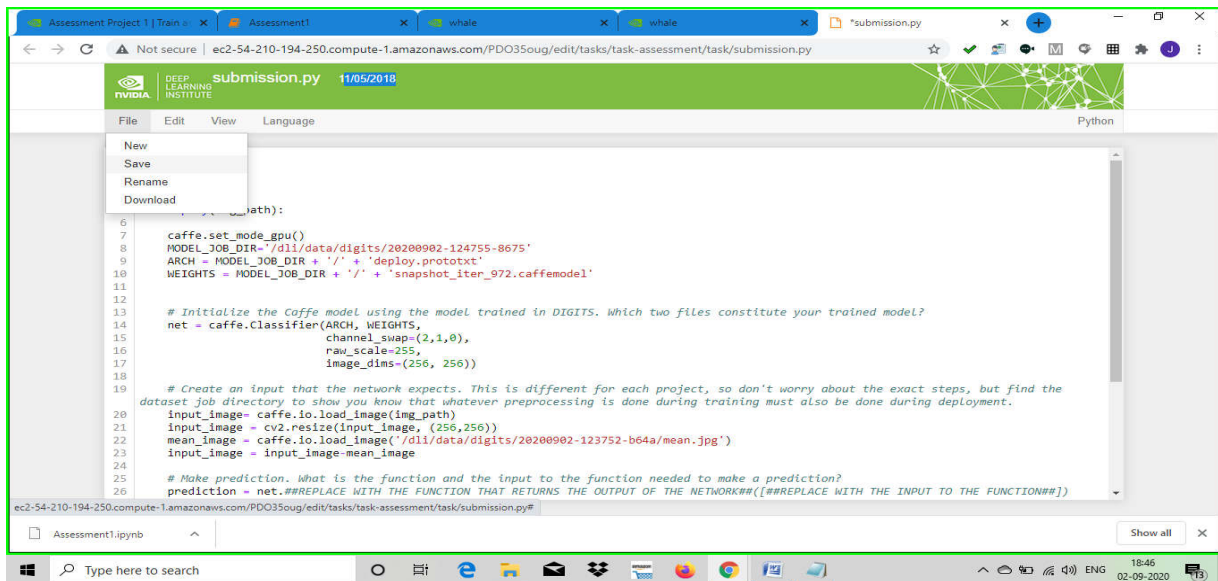
prediction = net.predict([input_image])

```
1 import caffe
2 import cv2
3 import sys
4
5 def deploy(img_path):
6
7     caffe.set_mode_gpu()
8     MODEL_JOB_DIR = '/dl1/data/digits/20200902-124755-8675'
9     ARCH = MODEL_JOB_DIR + '/' + 'deploy.prototxt'
10    WEIGHTS = MODEL_JOB_DIR + '/' + 'snapshot_iter_972.caffemodel'
11
12
13    # Initialize the Caffe model using the model trained in DIGITS. Which two files constitute your trained model?
14    net = caffe.Classifier(ARCH, WEIGHTS,
15                          channel_swap=(2,1,0),
16                          raw_scale=255,
17                          image_dims=(256, 256))
18
19    # Create an input that the network expects. This is different for each project, so don't worry about the exact steps, but find the
20    # dataset job directory to show you know that whatever preprocessing is done during training must also be done during deployment.
21    input_image = caffe.io.load_image(img_path)
22    input_image = cv2.resize(input_image, (256,256))
23    mean_image = caffe.io.load_image('/dl1/data/digits/20200902-123752-b64a/mean.jpg')
24    input_image = input_image - mean_image
25
26    # Make prediction. What is the function and the input to the function needed to make a prediction?
27    prediction = net.predict([input_image])
```

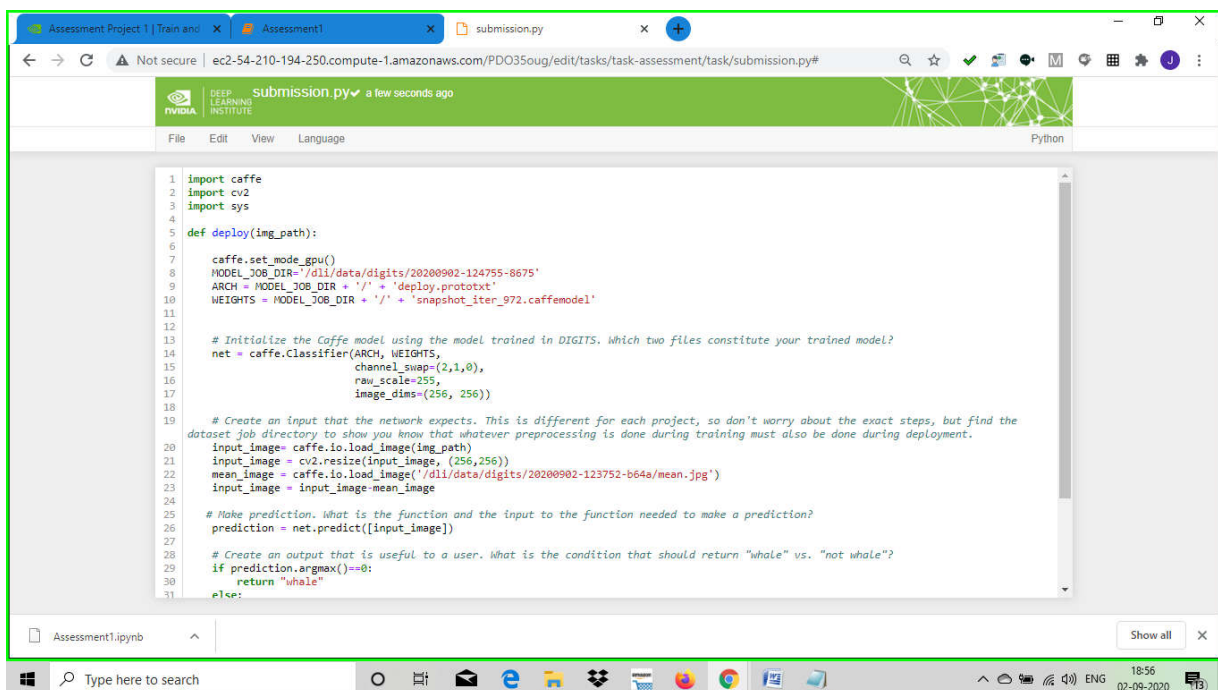
5th Change:

In the above at line no At line no 25

if prediction.argmax()==0:



24. Till here date was 11/05/2018 once when we save we get the below one



25. All five (5) changes are shown in the above image

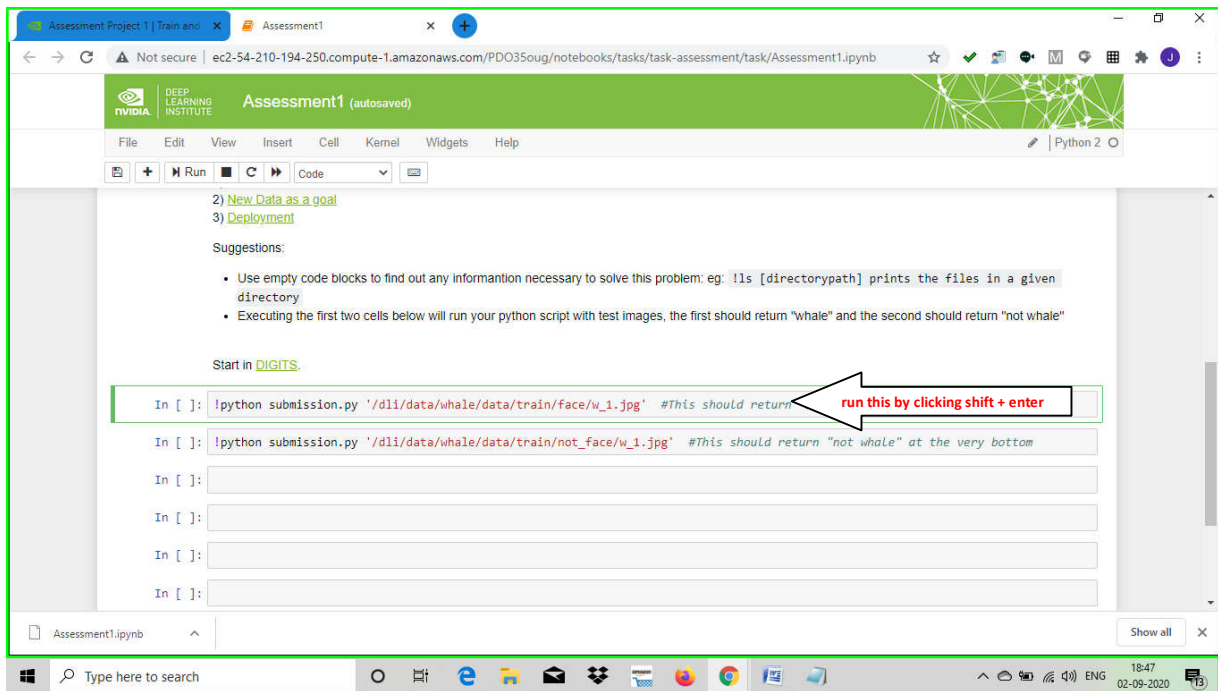
1st change at line no 7 to 10

2nd change at line no 14

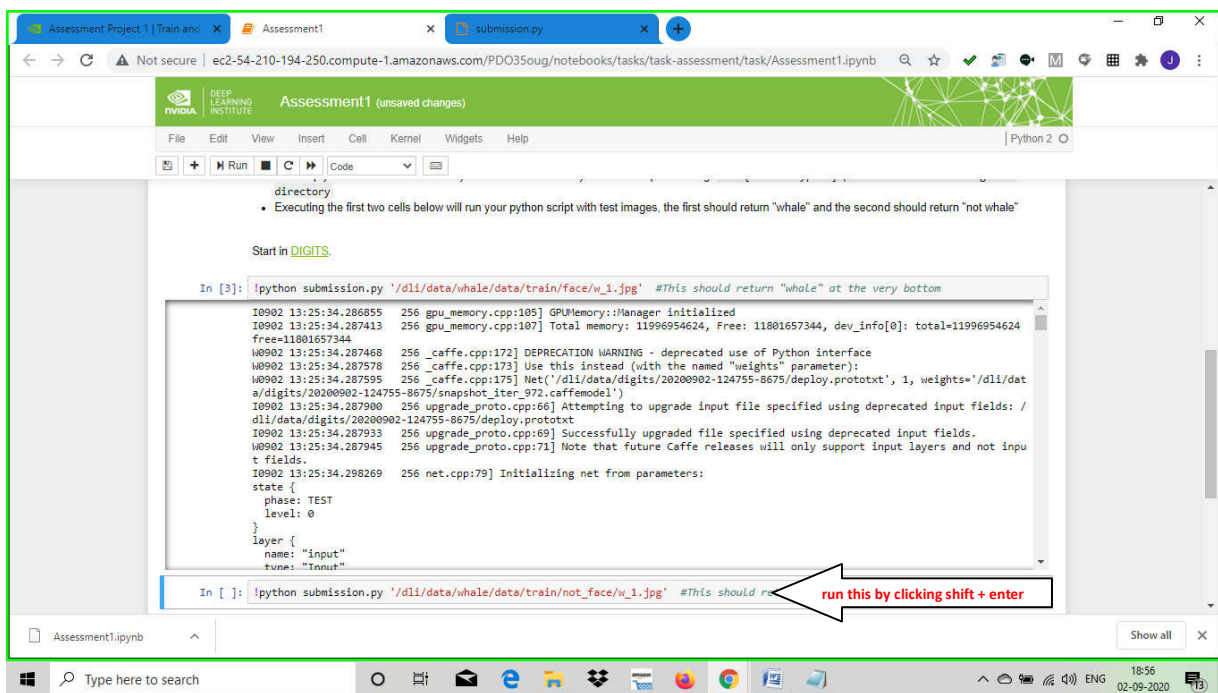
3rd change at line no 22

4th change at line no 26

5th change at line no 29



26. Now come back to Assessment tab(window) and run the above code



27. run the above code


```

I0902 13:25:36.005311 256 net.cpp:1137] Copying source layer relu6 Type:ReLU #blobs=0
I0902 13:25:36.005340 256 net.cpp:1137] Copying source layer drop6 Type:Dropout #blobs=0
I0902 13:25:36.013444 256 net.cpp:1137] Copying source layer relu7 Type:ReLU #blobs=0
I0902 13:25:36.013476 256 net.cpp:1137] Copying source layer drop7 Type:Dropout #blobs=0
I0902 13:25:36.013481 256 net.cpp:1137] Copying source layer fc8 Type:InnerProduct #blobs=2
I0902 13:25:36.013481 256 net.cpp:1137] Copying source layer loss Type:Loss #blobs=0
I0902 13:25:36.013481 256 net.cpp:1137] Ignoring source layer loss
whale

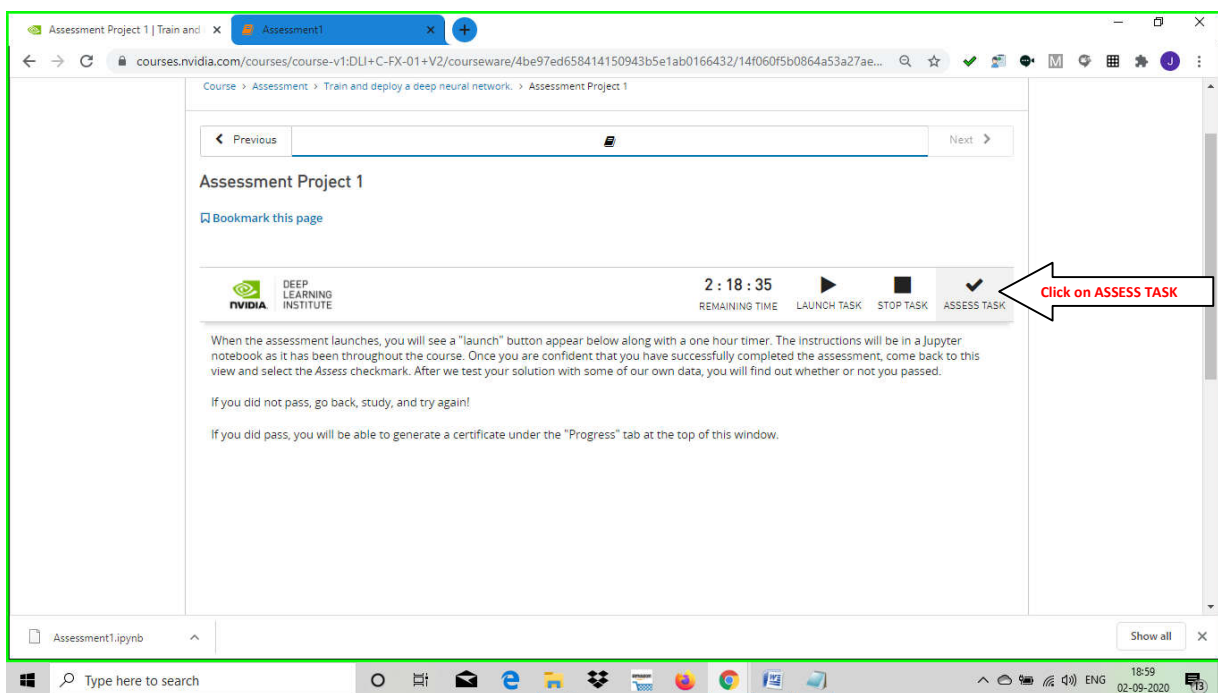
In [4]: !python submission.py "/d1/data/whale/data/train/not_face/wc.1.jpg" #This should return "not whale" at the very bottom
I0902 13:27:12.942335 271 net.cpp:1137] Copying source layer relu2 Type:ReLU #blobs=0
I0902 13:27:12.942349 271 net.cpp:1137] Copying source layer norm2 Type:LRN #blobs=0
I0902 13:27:12.942361 271 net.cpp:1137] Copying source layer pool2 Type:Pooling #blobs=0
I0902 13:27:12.942368 271 net.cpp:1137] Copying source layer conv3 Type:Convolution #blobs=2
I0902 13:27:12.942814 271 net.cpp:1137] Copying source layer relu3 Type:ReLU #blobs=0
I0902 13:27:12.942828 271 net.cpp:1137] Copying source layer conv4 Type:Convolution #blobs=2
I0902 13:27:12.943171 271 net.cpp:1137] Copying source layer relu4 Type:ReLU #blobs=0
I0902 13:27:12.943186 271 net.cpp:1137] Copying source layer conv5 Type:Convolution #blobs=2
I0902 13:27:12.943421 271 net.cpp:1137] Copying source layer relu5 Type:ReLU #blobs=0
I0902 13:27:12.943433 271 net.cpp:1137] Copying source layer pool5 Type:Pooling #blobs=0
I0902 13:27:12.943439 271 net.cpp:1137] Copying source layer fc6 Type:InnerProduct #blobs=2
I0902 13:27:12.960903 271 net.cpp:1137] Copying source layer relu6 Type:ReLU #blobs=0
I0902 13:27:12.961017 271 net.cpp:1137] Copying source layer drop6 Type:Dropout #blobs=0
I0902 13:27:12.961022 271 net.cpp:1137] Copying source layer fc7 Type:InnerProduct #blobs=2
I0902 13:27:12.969089 271 net.cpp:1137] Copying source layer relu7 Type:ReLU #blobs=0
I0902 13:27:12.969135 271 net.cpp:1137] Copying source layer drop7 Type:Dropout #blobs=0
I0902 13:27:12.969147 271 net.cpp:1137] Copying source layer fc8 Type:InnerProduct #blobs=2
I0902 13:27:12.969197 271 net.cpp:1129] Ignoring source layer loss
not whale

```

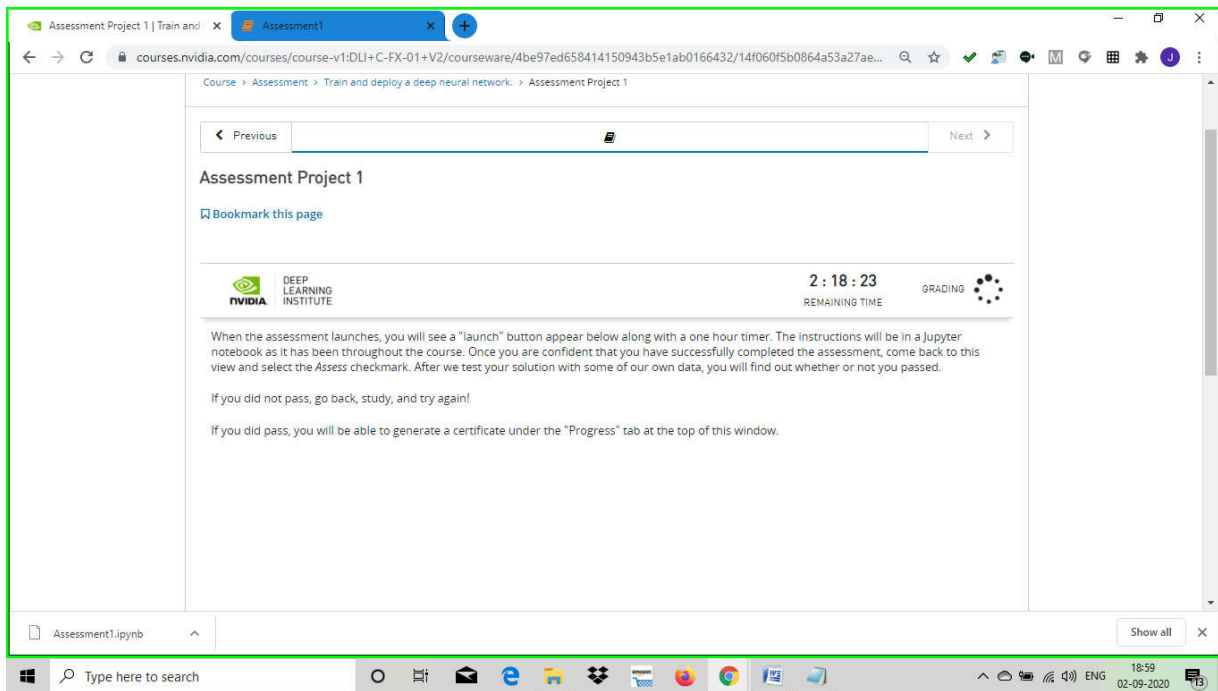
Look here by scrolling this segment

Look here by scrolling this segment

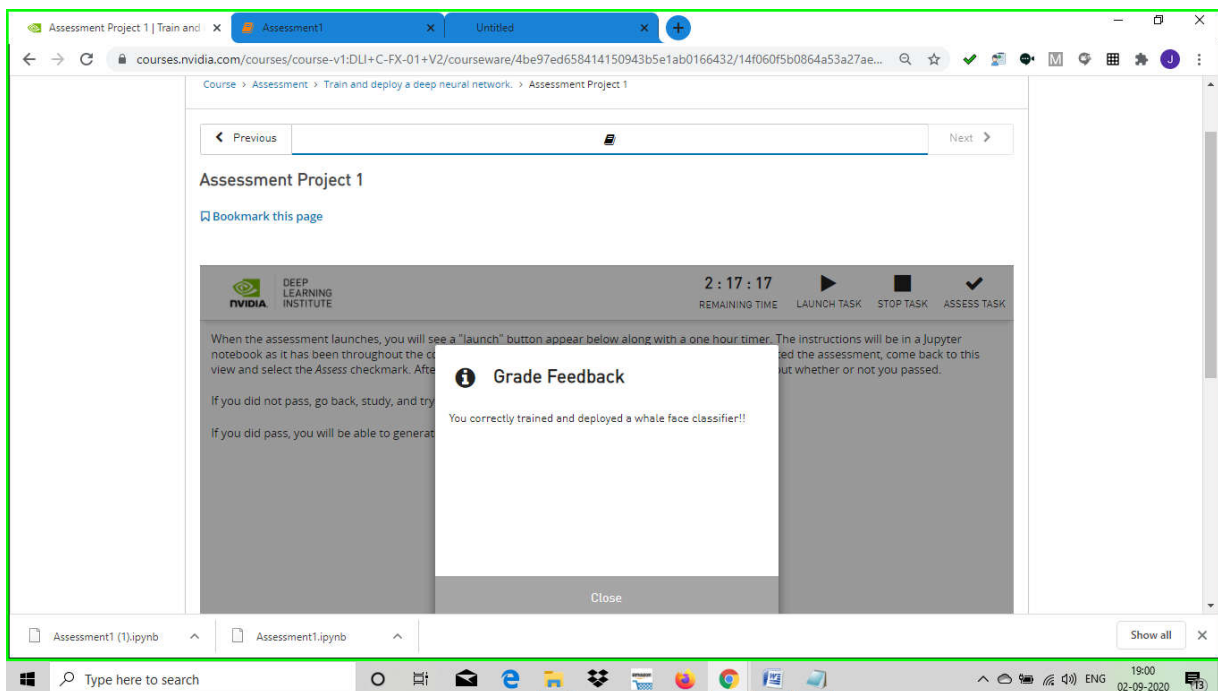
28. After successful execution you get first code as whale and second as not whale



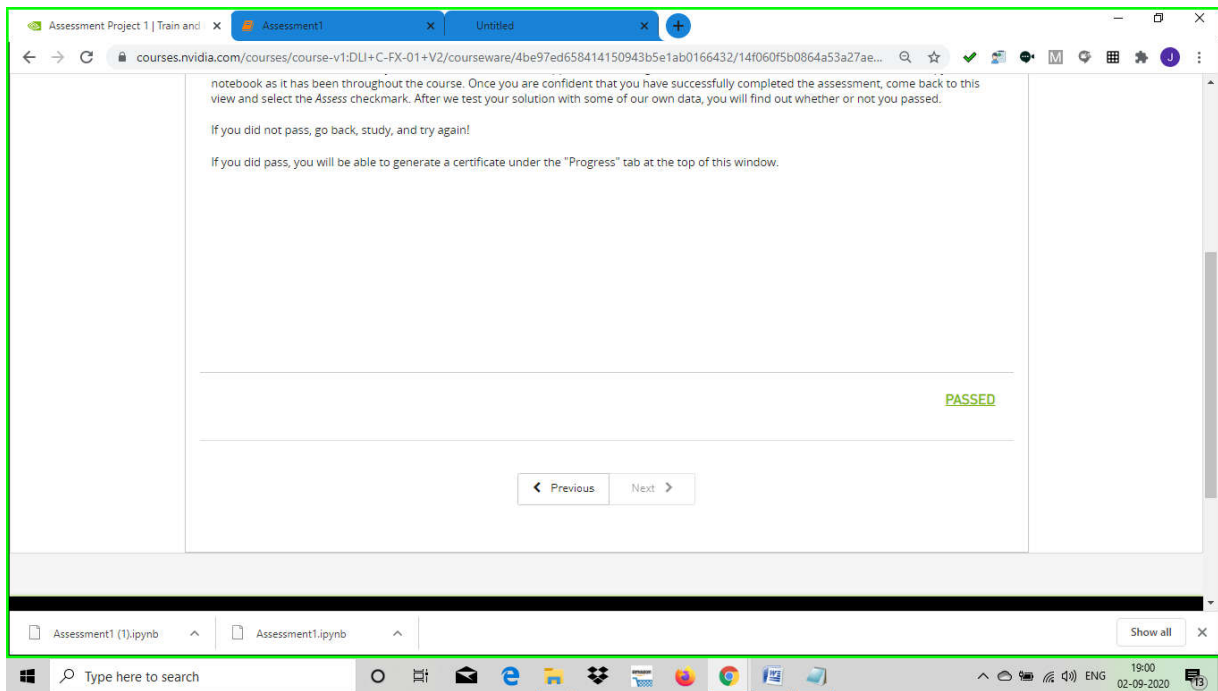
29. Now come back to assessment Project tab(window) click on assess task button as shown above.



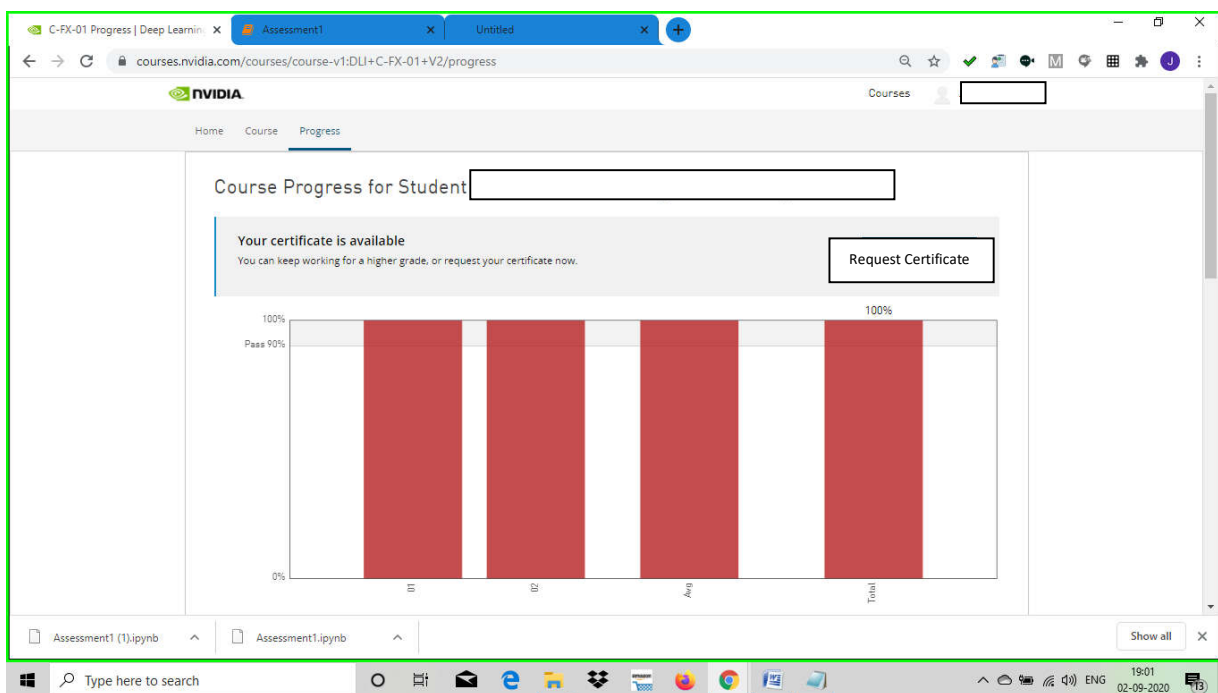
30. After clicking assess task your model will be assessed as shown above.



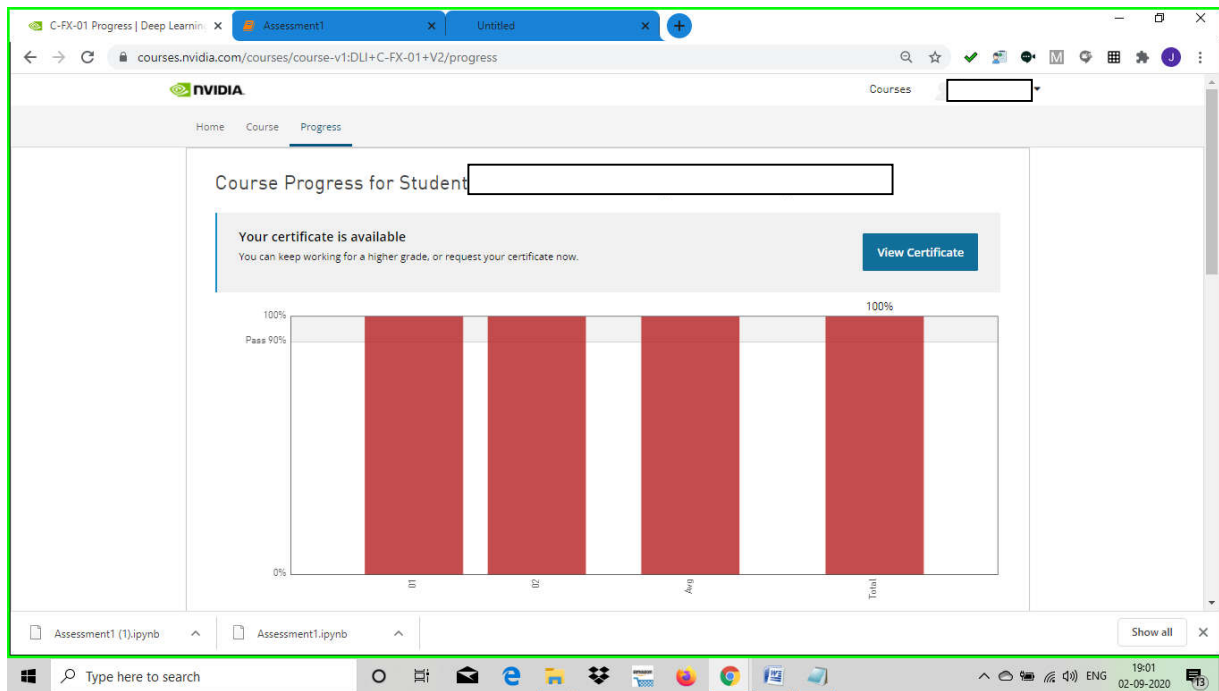
31. If you have followed the steps as said so far you will get the same screen as shown above click on close button.



32.if You scroll you will see the same on your screen as PASSED



33.Go to Progress Tab and Click on Request Certificate(but ensure you have answered all multiple choice questions in all the sections of this certification module or at least met passing criteria in this case it is 90%.



34. Click on view certificate button to get your certificate.

35. Share with all for easy access.

Thank you all....