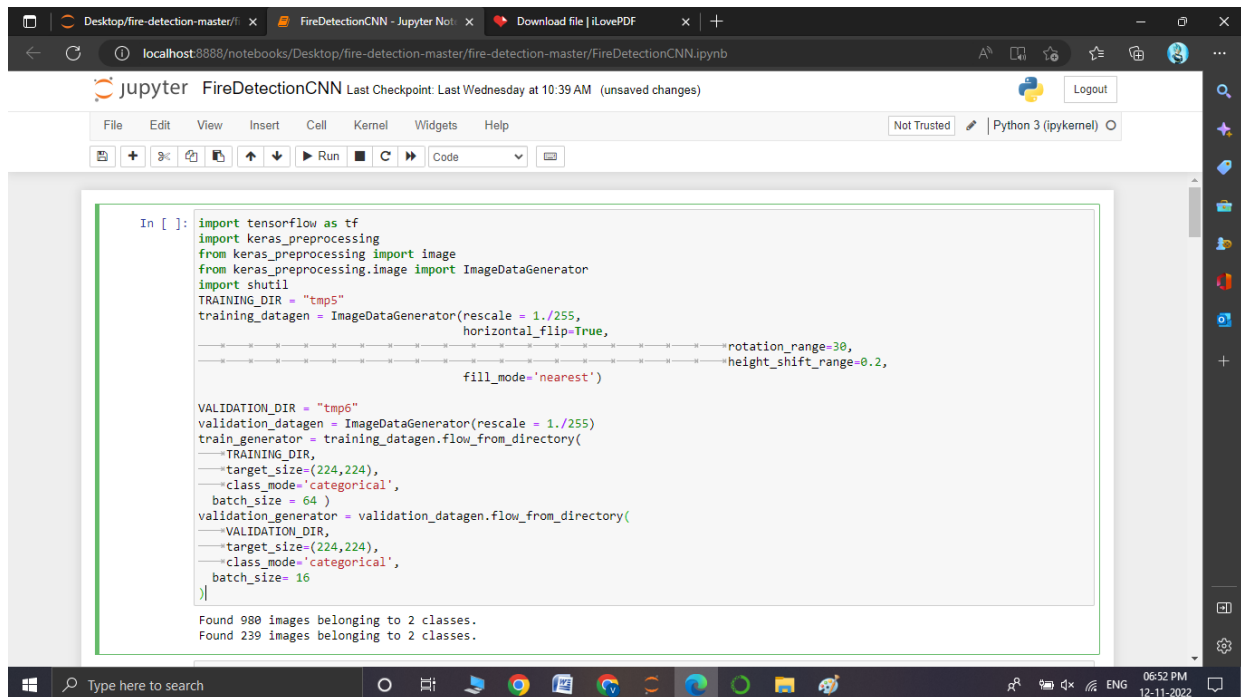


# EMERGING METHODS FOR EARLY DETECTION OF FORESTFIRE

## INITIALIZING THE MODEL

Team ID	PNT2022TMID21968
Project Name	Project-Emerging methods for early detection of forest fire.

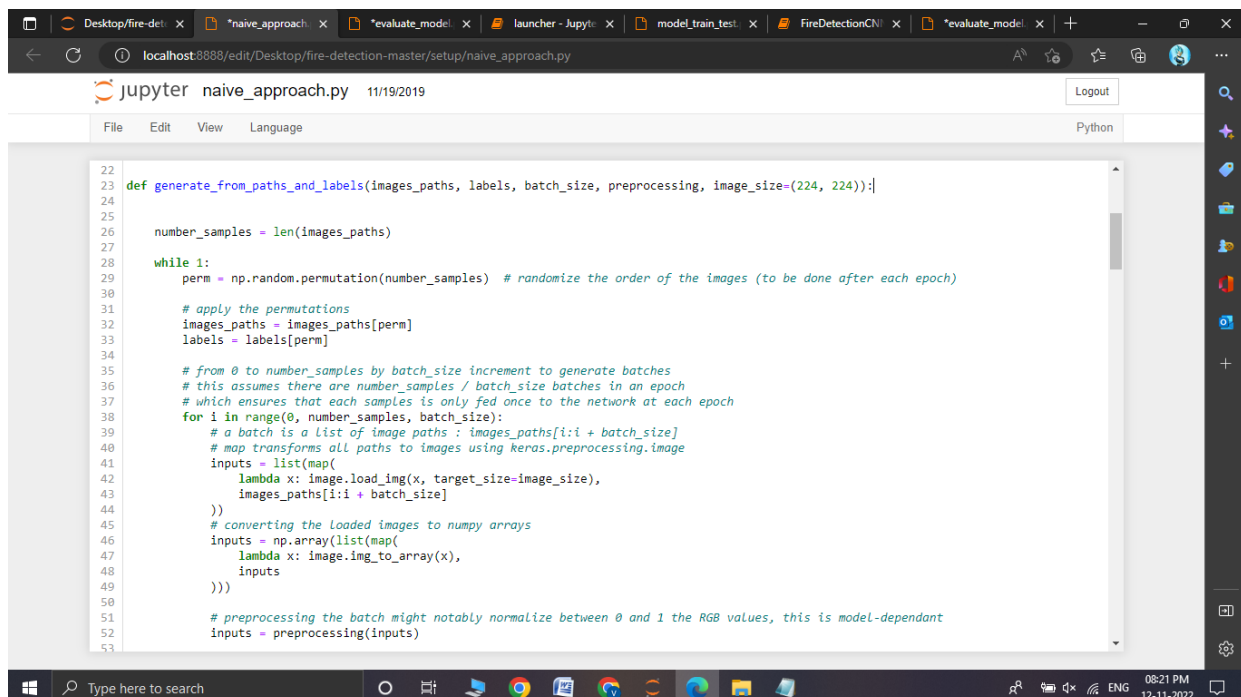


```
In [ ]: import tensorflow as tf
import keras_preprocessing
from keras_preprocessing import image
from keras_preprocessing.image import ImageDataGenerator
import shutil

TRAINING_DIR = "tmp5"
training_datagen = ImageDataGenerator(rescale = 1./255,
                                      horizontal_flip=True,
                                      rotation_range=30,
                                      height_shift_range=0.2,
                                      fill_mode='nearest')

VALIDATION_DIR = "tmp6"
validation_datagen = ImageDataGenerator(rescale = 1./255)
train_generator = training_datagen.flow_from_directory(
    TRAINING_DIR,
    target_size=(224,224),
    class_mode='categorical',
    batch_size = 64 )
validation_generator = validation_datagen.flow_from_directory(
    VALIDATION_DIR,
    target_size=(224,224),
    class_mode='categorical',
    batch_size= 16
)

Found 980 images belonging to 2 classes.
Found 239 images belonging to 2 classes.
```



```
22 def generate_from_paths_and_labels(images_paths, labels, batch_size, preprocessing, image_size=(224, 224)):
23
24
25
26     number_samples = len(images_paths)
27
28     while 1:
29         perm = np.random.permutation(number_samples) # randomize the order of the images (to be done after each epoch)
30
31         # apply the permutations
32         images_paths = images_paths[perm]
33         labels = labels[perm]
34
35         # from 0 to number_samples by batch_size increment to generate batches
36         # this assumes there are number_samples / batch_size batches in an epoch
37         # which ensures that each samples is only fed once to the network at each epoch
38         for i in range(0, number_samples, batch_size):
39             # a batch is a list of image paths : images_paths[i:i + batch_size]
40             # map transforms all paths to images using keras.preprocessing.image
41             inputs = list(map(
42                 lambda x: image.load_img(x, target_size=image_size),
43                 images_paths[i:i + batch_size]
44             ))
45             # converting the loaded images to numpy arrays
46             inputs = np.array(list(map(
47                 lambda x: image.img_to_array(x),
48                 inputs
49             )))
50
51             # preprocessing the batch might notably normalize between 0 and 1 the RGB values, this is model-dependant
52             inputs = preprocessing(inputs)
53
```