

Training & Testing Model on IBM cloud

Team ID - PNT2022TMID31063

Team Leader - AbishekGovindan K B

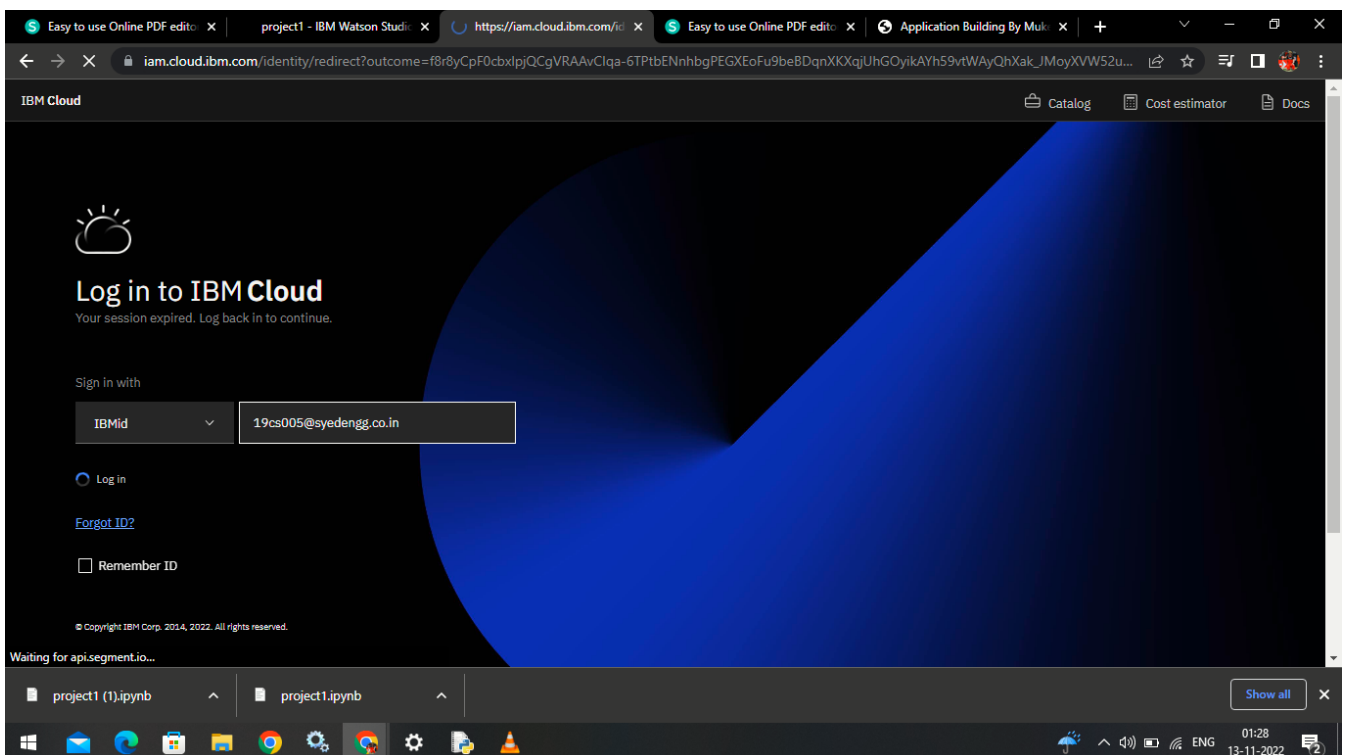
Team Member - MukeshKumar V

Team Member - Muneeswaran I

Team Member - Fazil Ahamed J

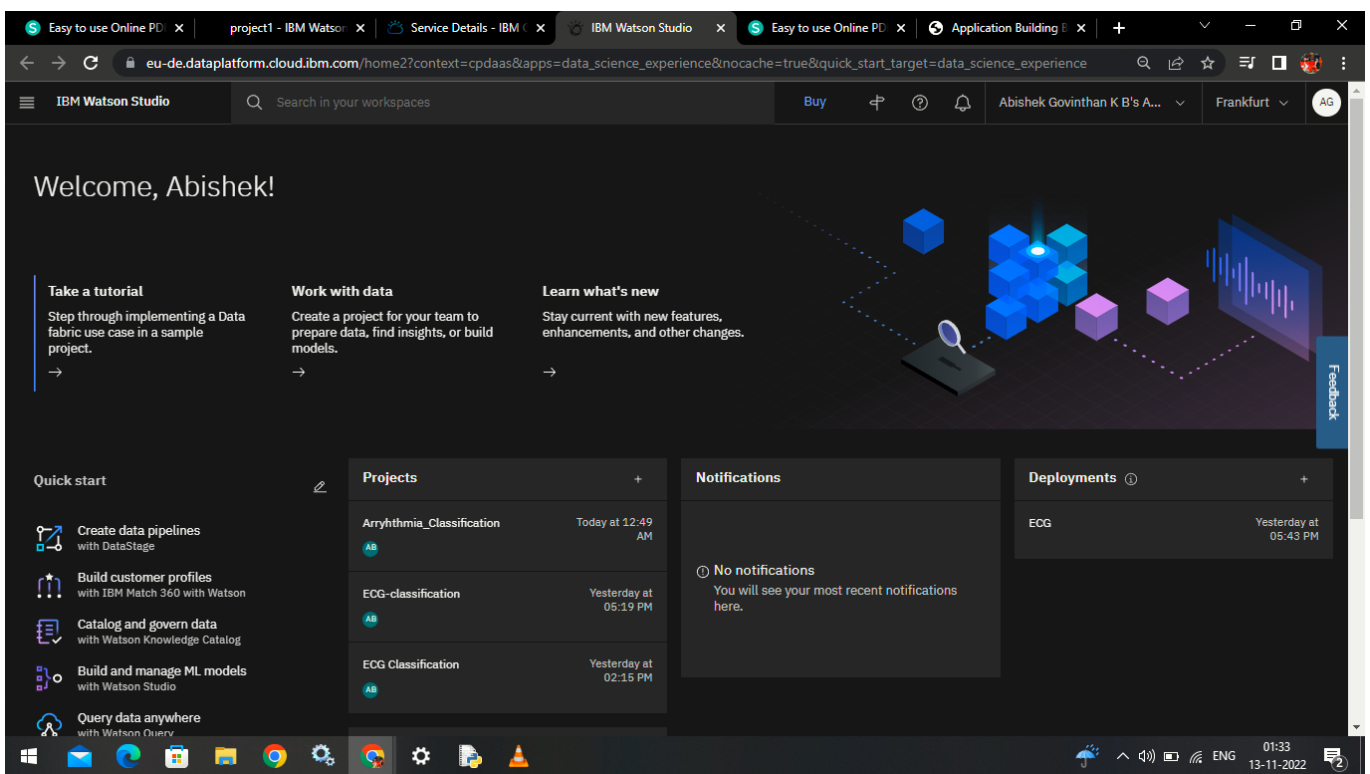
1.Register & Login IBM cloud

From given link we can Register and afterwards login the IBM cloud using credentials.



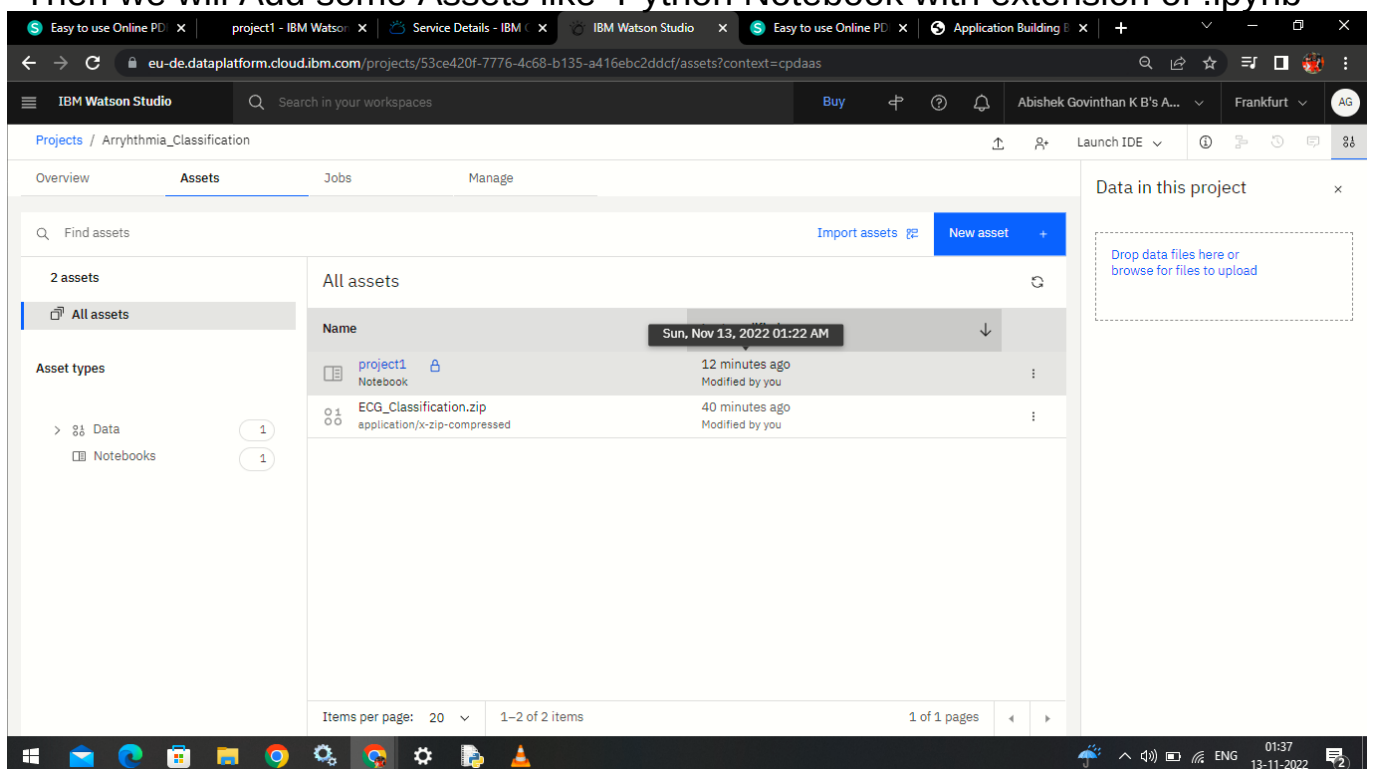
2.Opening IBM Watson Studio

To Run our model we use IBM watson studio inside we will create our own new project.



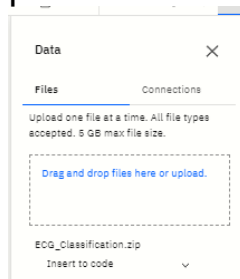
3.Adding Assests

Once we create a new project named Arryhythmia_Classification
Then we will Add some Assets like Python Notebook with extension of .ipynb



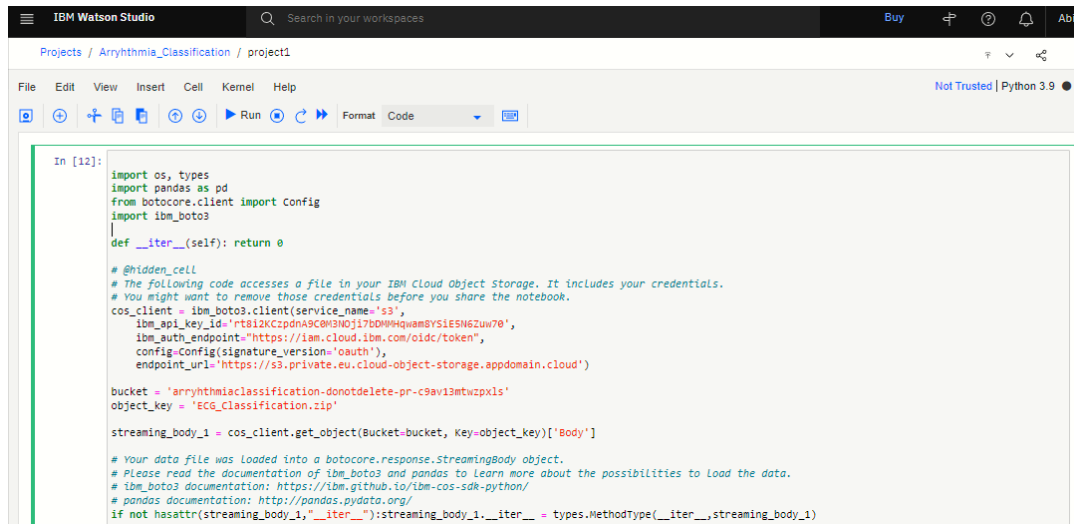
4 .Opening notebook file

At the end we will open the note book file and after we download the dataset
once we download it into zip file we will add the file into data assests.



5. Insert File code

Then we will create a new cell and insert the code into it.



```
In [12]:
import os, types
import pandas as pd
from botocore.client import Config
import boto3

def __iter__(self): return 0

#@hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.
cos_client = boto3.client(service_name='s3',
    iam_api_key_id='rt812K2CpdnA5CM3NOj17BDMMHqWamsYSIESN6Zuw70',
    iam_auth_endpoint='https://iam.cloud.ibm.com/oidc/token',
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.eu.cloud-object-storage.appdomain.cloud')

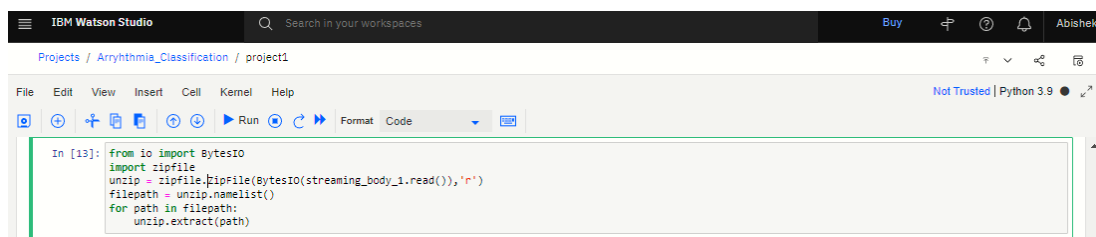
bucket = 'arrythmiaclassification-donotdelete-pr-c9av13mtwzpx1s'
object_key = 'ECG_Classification.zip'

streaming_body_1 = cos_client.get_object(Bucket=bucket, Key=object_key)['Body']

# Your data file was loaded into a botocore.response.StreamingBody object.
# Please read the documentation of boto3 and pandas to learn more about the possibilities to load the data.
# boto3 documentation: https://boto3.amazonaws.com/v1/documentation/api/latest/guide/quickstart.html#authentication
# pandas documentation: http://pandas.pydata.org/
if not hasattr(streaming_body_1, '__iter__'): streaming_body_1.__iter__ = types.MethodType(__iter__, streaming_body_1)
```

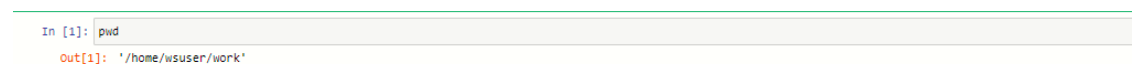
6. Unzip the file

Once we store the file we will deprecate or unzip the file from the beginning.



```
In [13]:
from io import BytesIO
import zipfile
unzip = zipfile.ZipFile(BytesIO(streaming_body_1.read()), 'r')
filepath = unzip.namelist()
for path in filepath:
    unzip.extract(path)
```

Know About the directory by using pwd command.



```
In [1]: pwd

Out[1]: '/home/wsuser/work'
```

7. Train the Model in IBM Watson Studio

After Completion of these stuffs we will move forward to start Train the model.



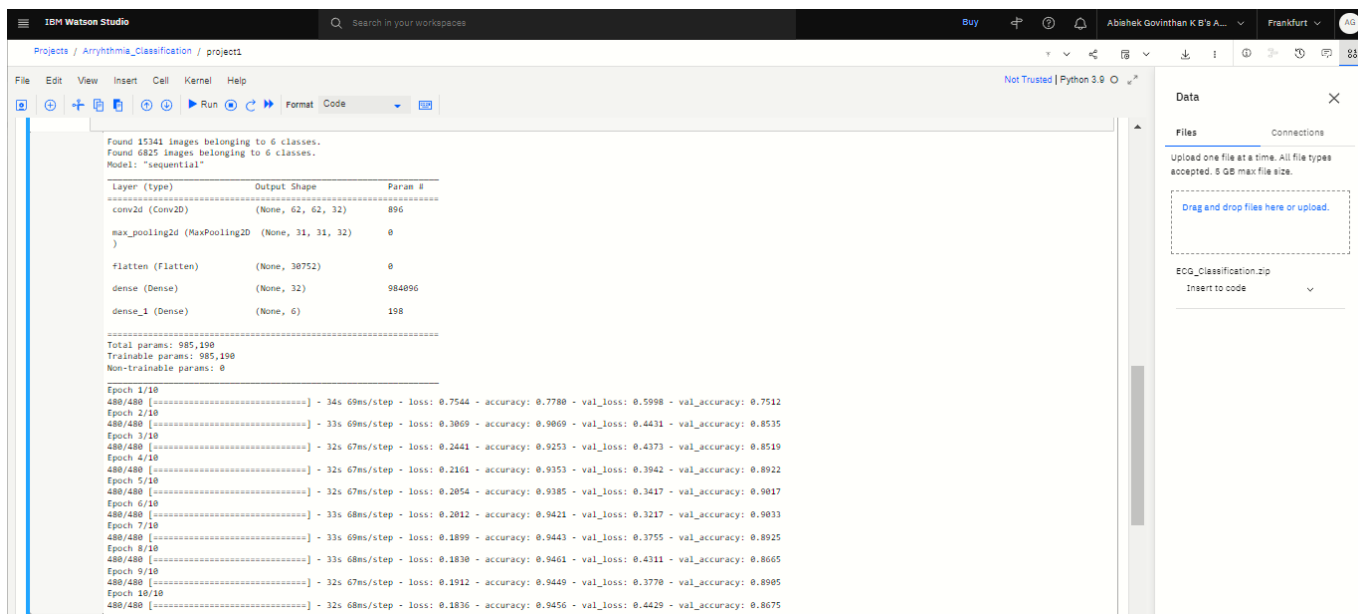
```
In [*]:
from keras.preprocessing.image import ImageDataGenerator
train_datagen=ImageDataGenerator(rescale=1./255, shear_range=0.2, zoom_range=0.2, horizontal_flip=True)
test_datagen=ImageDataGenerator(rescale=1./255)
x_train=train_datagen.flow_from_directory(directory=r'/home/wsuser/work/data/train', target_size=(64,64), batch_size=32, class_mode='categorical')
x_test=test_datagen.flow_from_directory(directory=r'/home/wsuser/work/data/test', target_size=(64,64), batch_size=32, class_mode='categorical')

import numpy as np
import tensorflow
from tensorflow.keras.models import Sequential
from tensorflow.keras import layers
from tensorflow.keras.layers import Dense, Flatten
from tensorflow.keras.layers import Conv2D, MaxPooling2D
model=Sequential()
model.add(Conv2D(32,(3,3),input_shape=(64,64,3),activation='relu'))
model.add(MaxPooling2D(pool_size=(2,2)))
model.add(Flatten())
model.add(Dense(32))
model.add(Dense(6,activation='softmax'))
model.summary()
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])

model.fit(x_train, steps_per_epoch=len(x_train), epochs=10, validation_data=x_test, validation_steps=len(x_test))

model.save('ECG.h5')
```

Outcome of the Trained Model will be shown after compiling and summarizing it.



```
Found 15341 images belonging to 6 classes.
Found 6825 images belonging to 6 classes.
Model: "sequential"

Layer (type)                 Output Shape              Param #
-----
conv2d (Conv2D)              (None, 62, 62, 32)       896
max_pooling2d (MaxPooling2D) (None, 31, 31, 32)       0
flatten (Flatten)            (None, 30752)             0
dense (Dense)                (None, 32)               984896
dense_1 (Dense)              (None, 6)                198

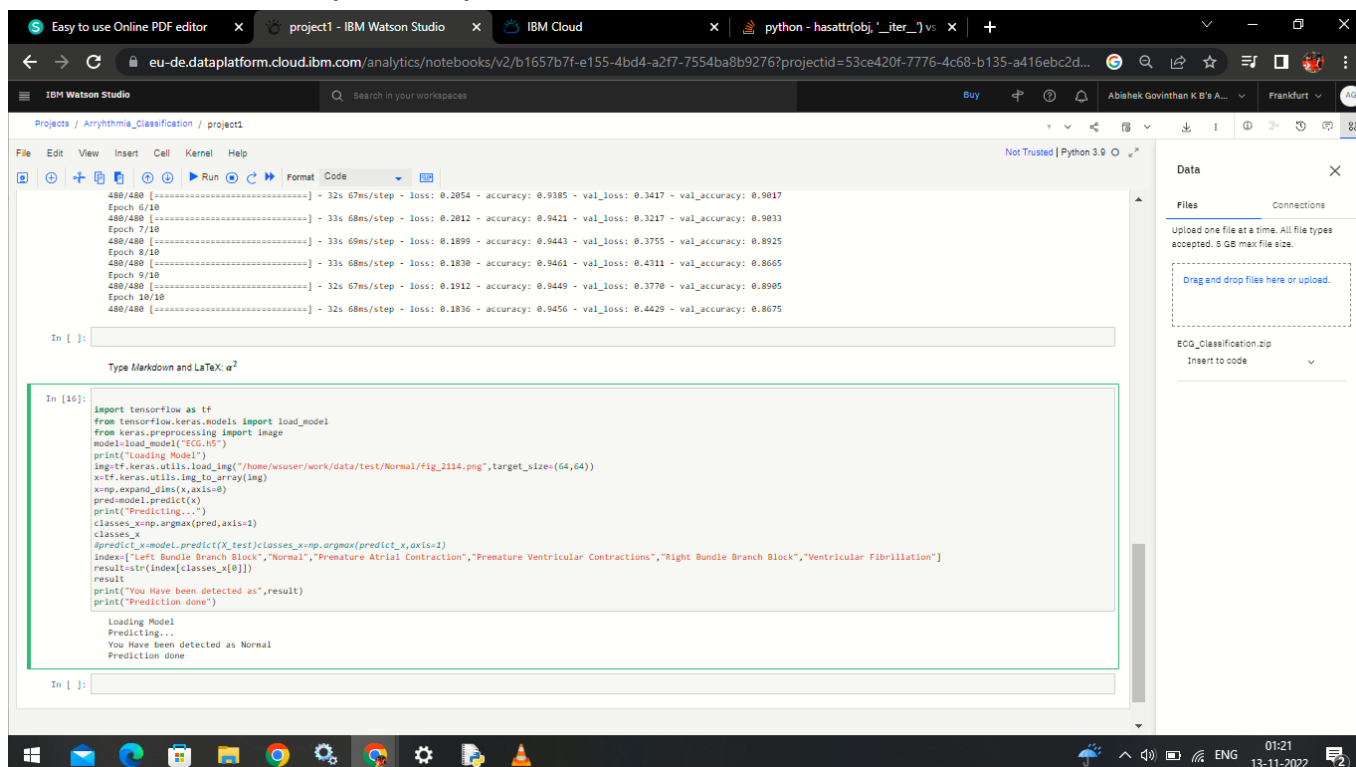
Total params: 985,190
Trainable params: 985,190
Non-trainable params: 0

Epoch 1/10
480/480 [=====] - 34s 69ms/step - loss: 0.7544 - accuracy: 0.7780 - val_loss: 0.5998 - val_accuracy: 0.7512
Epoch 2/10
480/480 [=====] - 33s 69ms/step - loss: 0.3069 - accuracy: 0.9069 - val_loss: 0.4431 - val_accuracy: 0.8535
Epoch 3/10
480/480 [=====] - 32s 67ms/step - loss: 0.2441 - accuracy: 0.9253 - val_loss: 0.4373 - val_accuracy: 0.8519
Epoch 4/10
480/480 [=====] - 32s 67ms/step - loss: 0.2161 - accuracy: 0.9353 - val_loss: 0.3942 - val_accuracy: 0.8922
Epoch 5/10
480/480 [=====] - 32s 67ms/step - loss: 0.2054 - accuracy: 0.9385 - val_loss: 0.3417 - val_accuracy: 0.9017
Epoch 6/10
480/480 [=====] - 33s 68ms/step - loss: 0.2012 - accuracy: 0.9421 - val_loss: 0.3217 - val_accuracy: 0.9033
Epoch 7/10
480/480 [=====] - 33s 69ms/step - loss: 0.1899 - accuracy: 0.9443 - val_loss: 0.3755 - val_accuracy: 0.8925
Epoch 8/10
480/480 [=====] - 33s 68ms/step - loss: 0.1830 - accuracy: 0.9461 - val_loss: 0.4311 - val_accuracy: 0.8665
Epoch 9/10
480/480 [=====] - 32s 67ms/step - loss: 0.1912 - accuracy: 0.9449 - val_loss: 0.3778 - val_accuracy: 0.8985
Epoch 10/10
480/480 [=====] - 32s 68ms/step - loss: 0.1836 - accuracy: 0.9456 - val_loss: 0.4429 - val_accuracy: 0.8675
```

This how the Model was Trained in IBM watson Studio.

8. Testing the Model

At the end of the day we will try to test the model which can give the desired and precise prediction based on the trained model.



```
In [16]:
import tensorflow as tf
from tensorflow.keras.models import load_model
from keras.preprocessing import image
model=load_model('ECG.h5')
print('Loading Model')
img=tf.keras.utils.load_img('/home/wuser/work/data/test/Normal/fig_2114.png',target_size=(64,64))
x=tf.keras.utils.img_to_array(img)
x=np.expand_dims(x,axis=0)
pred=model.predict(x)
print('Predicting...')
classes_x=np.argmax(pred,axis=1)
classes_x
#predict_x=model.predict(X_test)classes_x=np.argmax(predict_x,axis=1)
index=[0,1,2,3,4,5]
result=[str(index[classes_x[i]])]
print('You Have been detected as',result)
print('Prediction done')

Loading Model
Predicting...
You Have been detected as Normal
Prediction done
```

This How the Model was tested in IBM Watson Studio.

Regards
Mukesh & Team