Import the Libraries:

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Convolution2D
from tensorflow.keras.layers import MaxPooling2D
from tensorflow.keras.layers import Flatten
```

Adding CNN Layers:

```
model = Sequential()
```

```
model.add(Convolution2D(32,(3,3),input_shape = (64,64,3),activation = "relu"))
```

```
model.add(MaxPooling2D(pool_size = (2,2)))
```

```
model.add(Convolution2D(32,(3,3),activation='relu'))
```

```
model.add(MaxPooling2D(pool_size=(2,2)))
```

```
model.add(Flatten()) # ANN Input...
```

Adding Dense Layers:

```
model.add(Dense(units = 128,kernel_initializer = "random_uniform",activation = "relu"))
```

```
model.add(Dense(units = 128,kernel_initializer = "random_uniform",activation = "relu"))
```

```
model.add(Dense(units = 128,kernel_initializer = "random_uniform",activation = "relu"))
```

```
model.add(Dense(units = 128,kernel_initializer = "random_uniform",activation = "relu"))
```

```
model.add(Dense(units = 128,kernel_initializer = "random_uniform",activation = "relu"))
```

Adding Output Layer:

```
model.add(Dense(units = 6,kernel_initializer = "random_uniform",activation = "softmax"))
```

```
model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 conv2d (Conv2D)             (None, 62, 62, 32)        896       
                                                                 
 max_pooling2d (MaxPooling2D  (None, 31, 31, 32)       0         
 )                                                               
                                                                 
 conv2d_1 (Conv2D)           (None, 29, 29, 32)        9248      
                                                                 
 max_pooling2d_1 (MaxPooling  (None, 14, 14, 32)       0         
 2D)                                                             
                                                                 
 flatten (Flatten)           (None, 6272)              0         
                                                                 
 dense (Dense)               (None, 128)               802944    
                                                                 
 dense_1 (Dense)             (None, 128)               16512     
                                                                 
 dense_2 (Dense)             (None, 128)               16512     
                                                                 
 dense_3 (Dense)             (None, 128)               16512     
                                                                 
 dense_4 (Dense)             (None, 128)               16512     
                                                                 
 dense_5 (Dense)             (None, 6)                 774       
                                                                 
=================================================================
Total params: 879,910
Trainable params: 879,910
Non-trainable params: 0
_________________________________________________________________
```

```
model.compile(optimizer='adam',loss='categorical_crossentropy',metrics=['accuracy'])
```

Train the model:

```
model.fit_generator(generator=x_train,steps_per_epoch = len(x_train), epochs=9, validation_da
```

```
/usr/local/lib/python3.7/dist-packages/ipykernel_launcher.py:1: UserWarning: `Model.fit_
```

```
  """Entry point for launching an IPython kernel.
Epoch 1/9
480/480 [==============================] - 41s 66ms/step - loss: 1.3631 - accuracy: 0.56
Epoch 2/9
480/480 [==============================] - 31s 65ms/step - loss: 0.7976 - accuracy: 0.69
Epoch 3/9
480/480 [==============================] - 34s 71ms/step - loss: 0.3399 - accuracy: 0.88
Epoch 4/9
480/480 [==============================] - 30s 63ms/step - loss: 0.2286 - accuracy: 0.92
Epoch 5/9
480/480 [==============================] - 30s 63ms/step - loss: 0.1798 - accuracy: 0.94
Epoch 6/9
480/480 [==============================] - 30s 63ms/step - loss: 0.1416 - accuracy: 0.95
Epoch 7/9
480/480 [==============================] - 30s 62ms/step - loss: 0.1068 - accuracy: 0.96
Epoch 8/9
480/480 [==============================] - 30s 63ms/step - loss: 0.0917 - accuracy: 0.97
Epoch 9/9
480/480 [==============================] - 30s 62ms/step - loss: 0.0796 - accuracy: 0.97
<keras.callbacks.History at 0x7f85e00f6410>
```

Save the model:

```
#Saving Model.
model.save('ECG.h5')
```

Testing the model:

```
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
```

```
model=load_model('ECG.h5')
```

```
img=image.load_img("/content/fig_44.png",target_size=(64,64))
```

```
x=image.img_to_array(img)
```

```
img
```

```
import numpy as np
```

```
x=np.expand_dims(x,axis=0)
```

```
pred = model.predict(x)
y_pred=np.argmax(pred)
y_pred
```

```
     1/1 [==============================] - 0s 151ms/step
     4
```

```
index=['left Bundle Branch block',
       'Normal',
       'Premature Atrial Contraction',
       'Premature Ventricular Contraction',
       'Right Bundle Branch Block',
       'Ventricular Fibrillation']
```

```
result = str(index[y_pred])
result
```

```
     'Right Bundle Branch Block'
```