**ECG - IMAGE BASED HEARTBEAT CLASSIFICATION FOR ARRHYTHMIA DETECTION USING IBM WASTON STUDIO**

An Internship Project Report

Submitted By:

SALLA SAKETH(18UK1A05A7)

THALLADA SAI TEJA(18UK1A0B2)

PALLAKURTHI RAHUL(18UK1A05A1)

GAYAPU SINDHU RAJ(18UK1A05D2)

**INDEX**

**TITLE PAGE NO**

**CHAPTER 1: INTRODUCTION**

* 1. OVERVIEW **3**
  2. PURPOSE **3**

**CHAPTER 2: LITERATURE SURVEY**

* 1. EXISTING PROBLEM **4**
  2. PROPOSED SOLUTION **4**

**CHAPTER 3: THEORETICAL ANALYSIS**

3.1 BLOCK DIAGRAM **5**

3.2 HARDWARE/SOFTWARE DESIGNING **6-7**

**CHAPTER 4: EXPERIMENTAL INVESTIGATIONS 8-9**

**CHAPTER 5: FLOW CHART 10**

**CHAPTER 6: RESULTS 11-17**

**CHAPTER 7: ADVANTAGES AND DISADVANTAGES 18**

**CHAPTER 8: APPLICATIONS 19**

**CHAPTER 9: CONCLUSION 19**

**CHAPTER 10: FUTURE SCOPE 19**

**CHAPTER 11: BIBLIOGRAPHY 20**

**CHAPTER 1**

**INTRODUCTION**

* 1. **Overview:**

According to the World Health Organization (WHO), cardiovascular diseases (CVD ’s) are the number one cause of death today. Over 17.7 million people died from CVD ’s in the year 2017 all over the world which is about 31% of all deaths, and over 75% of these deaths occur in low and middle-income countries.

Arrhythmia is a representative type of CVD that refers to any irregular change from the normal heart rhythms. There are several types of arrhythmia including atrial fibrillation, premature contraction, ventricular fibrillation, and tachycardia. Although a single arrhythmia heartbeat may not have a serious impact on life, continuous arrhythmia beats can result in fatal circumstances.

In this project, we build an effective electrocardiogram (ECG) arrhythmia classification method using a convolution neural network (CNN), in which we classify ECG into seven categories, one being normal and the other six being different types of arrhythmia using deep two-dimensional CNN with grayscale ECG images. We are creating a web application where the user selects the image which is to be classified. The image is fed into the model that is trained and the cited class will be displayed on the web page.

**1.2 Purpose:**

The purpose of this project is to predict the arrhythmia detection on ECG-image based heart beat classification using the deep neural network algorithm by this we can be capable to predict the heart beat classification

for arrhythmia detection. we will prepare the data using JUPYTER notebook and we use various models to predict the output.Deep Learning technique are used very useful in predicting outcomes for large amount of image data. We use Convolution Neural Network (CNN) deep learning algorithm to predict the arrhythmia detection based on ECG image based heart beat classification.

**CHAPTER 2**

**LITERATURE SURVEY**

* 1. **Existing Problem:**

The AAMI standard specifies a protocol for tests and evaluation of arrhythmia classification methods. It also stipulates which databases should be used. However, it does not specify which patients/heartbeats should be used to construct the model to be classified (training phase) and which patients/heartbeats should be used for evaluation methods, *i.e.*,the testing phase, which may render biased results.

Demonstrated that the use of heartbeats from the same patient for both the training and the testing makes the evaluation process biased. This is because the models tend to learn the particularities of the patient’s heartbeat during the training, obtaining expressive numbers during the test (very close to 100%). As previously mentioned, this heartbeat division protocol is called in the literature infra-patient scheme or paradigm. However, in a clinical environment, a fully automatic algorithm/method will find heartbeats of patients different from those they used to learning in the training phase.

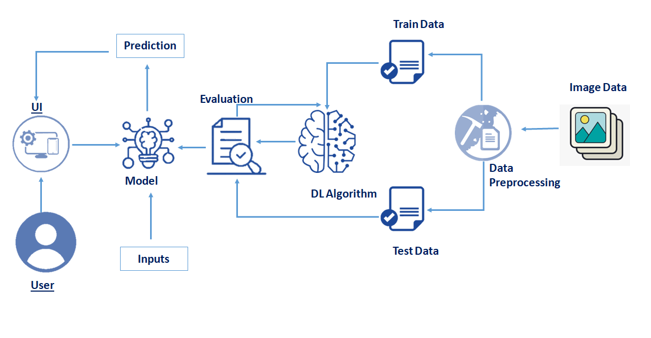
* 1. **Proposed Solution:**

We will prepare the data using a JUPYTER notebook and we use various models to predict the output.Deep Learning techniques are used very useful in predicting outcomes for large amount of image data. We use Conventional Neural Network (CNN) deep learning algorithm to predict the arrhythmia detection based on ECG image based heart beat classification.

**CHAPTER 3**

**THEORTICAL ANALYSIS**

**3.1 Block Diagram:**



**3.2 Hardware/software designing:**

Software specifications:

|  |  |
| --- | --- |
| **REQUIREMENT:** | **SPECIFICATION:** |
| Anaconda Navigator | You must have anaconda installed in your device prior to begin. |
| * SPYDER ,JUPYTER Notebook, Flask * Frame work   . | * One should have SPYDER and JUPYTER notebook. * One should install flask framework through Anaconda prompt for running their web application. * We need to build the mode; using JUPYTER notebook with all the imported packages. |
| Web browser | For all Web browsers, the following must be enabled:   * Cookies * Java script |

Hardware Specifications:

|  |  |
| --- | --- |
| **REQUIREMENT** | **SPECIFICATIONS** |
| Operating system | * Microsoft windows * Unix * Linux |
| Processing | Minimum: 4 CPU cores for one user. For each deployment, a sizing exercise is highly recommended. |
| RAM | Minimum 8 GB. |
| Operating system specifications | File descriptor limit set to 8192 on UNIX and Linux |
| Disk space | A minimum of 7 GB of free space is required to install the software. |

**CHAPTER 4**

**EXPERIMENTAL INVESTIGATIONS**

Analysis or the investigation made while working on the solution:

While working on the solution we investigated on what is Arrhythmia Detection,IBM cloud, IBM Watson studio, Machine Learning service, Cloud Object Storage. The key role on investigation is collection of data set.

**IBM Cloud Account**:

IBM Acquired soft layer, a public cloud platform, to serve as the foundation for its I a a S offering. In October 2016, IBM rolled the soft layer brand under its Blue mix brand of P a a S offerings, giving users to access both I a a S and P a a S resources from a single console. IBM cloud provides a full-stack, public cloud platform with various products in the catalog, including options for compute, storage, networking, end to end developer solutions for app development, testing and deployment, security databases, and cloud native services.

Creating the IBM cloud account by going to the IBM cloud login page and click create on IBM cloud account. Enter our IBM id and an ID is created based on the email that we enter. Completing the remaining fields with our information and click create account by this the account is created.

**Data set collection**:

* kaggle.com
* data.gov
* UCI machine learning repository.

The data set contains six classes:

1. Left Bundle Branch Block
2. Normal
3. Premature Atrial Contraction
4. Premature Ventricular Contractions
5. Right Bundle Branch Block
6. Ventricular Fibrillation

**CHAPTER 5**

**FLOW CHART:**

IBM CLOUD ACCOUNT

IBM WASTON STUDIO

LOADING THE DATASRT INTO IBM WASTON STUDIO

JUPYTER NOTEBOOK ON IBM

READING THE DATA AND DECIDING THE ALGORITHM AND TRAINING THE MODEL

TESTING THE MODEL

**Chapter 6**

**Results**

The code used in this project:

# ECG arrhythmia classification using CNN

### Importing Neccessary Libraries

import numpy as np#used for numerical analysis

import tensorflow #open source used for both ML and DL for computation

from tensorflow.keras.models import Sequential #it is a plain stack of layers

from tensorflow.keras import layers #A layer consists of a tensor-in tensor-out computation function

#Dense layer is the regular deeply connected neural network layer

from tensorflow.keras.layers import Dense,Flatten

#Faltten-used fot flattening the input or change the dimension

from tensorflow.keras.layers import Conv2D,MaxPooling2D #Convolutional layer

#MaxPooling2D-for downsampling the image

from keras.preprocessing.image import ImageDataGenerator

​

### Image Data Agumentation

#setting parameter for Image Data agumentation to the traing data

train\_datagen=ImageDataGenerator(rescale=1./255,shear\_range=0.2,zoom\_range=0.2,horizontal\_flip=True)

#Image Data agumentation to the testing data

test\_datagen=ImageDataGenerator(rescale=1./255)

### Loading our data and performing data agumentation

#performing data agumentation to train data

x\_train=train\_datagen.flow\_from\_directory(directory=r'C:\Users\saket\OneDrive\Desktop\project dataset\data\train'

,target\_size=(64,64),batch\_size=32,class\_mode='categorical')

#performing data agumentation to test data

x\_test=test\_datagen.flow\_from\_directory(directory=r'C:\Users\saket\OneDrive\Desktop\project dataset\data\test'

,target\_size=(64,64),batch\_size=32,class\_mode='categorical')

Found 15341 images belonging to 6 classes.

Found 6825 images belonging to 6 classes.

print(x\_train.class\_indices)#checking the number of classes

{'Left Bundle Branch Block': 0, 'Normal': 1, 'Premature Atrial Contraction': 2, 'Premature Ventricular Contractions': 3, 'Right Bundle Branch Block': 4, 'Ventricular Fibrillation': 5}

from collections import Counter as c

c(x\_train.labels)

Counter({0: 504, 1: 7346, 2: 2054, 3: 2759, 4: 2239, 5: 439})

### Creating the model

# create model

model=Sequential()

# adding model layer

model.add(Conv2D(32,(3,3),input\_shape=(64,64,3),activation='relu'))#convolutional layer

model.add(MaxPooling2D(pool\_size=(2,2))) #MaxPooling2D-for downsampling the input

​

model.add(Conv2D(32,(3,3),activation='relu'))

model.add(MaxPooling2D(pool\_size=(2,2)))

​

model.add(Flatten())#flatten the dimension of the image

model.add(Dense(32))#deeply connected neural network layers.

model.add(Dense(6,activation='soft max'))#output layer with 6 neurons

​

model.summary()#summary of our model

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 (MaxPooling2 (None, 14, 14, 32) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

flatten (Flatten) (None, 6272) 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dense (Dense) (None, 32) 200736

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

dense\_1 (Dense) (None, 6) 198

=================================================================

Total params: 211,078

Trainable params: 211,078

Non-trainable params: 0

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

### Compiling the model

# Compile model

model.compile(optimizer='adam',loss='categorical\_cross entropy',metrics=['accuracy'])

### Fitting the model

# Fit the model

model.fit\_generator(generator=x\_train,steps\_per\_epoch = len(x\_train),

epochs=10, validation\_data=x\_test,validation\_steps = len(x\_test))

C:\Users\saket\.vscode\anakonda.s\lib\site-packages\keras\engine\training.py:1972: UserWarning: `Model.fit\_generator` is deprecated and will be removed in a future version. Please use `Model.fit`, which supports generators.

warnings.warn('`Model.fit\_generator` is deprecated and '

Epoch 1/10

480/480 [==============================] - 404s 839ms/step - loss: 0.8690 - accuracy: 0.7028 - val\_loss: 0.7367 - val\_accuracy: 0.7451

Epoch 2/10

480/480 [==============================] - 107s 222ms/step - loss: 0.3397 - accuracy: 0.9003 - val\_loss: 0.5277 - val\_accuracy: 0.8442

Epoch 3/10

480/480 [==============================] - 94s 196ms/step - loss: 0.2809 - accuracy: 0.9195 - val\_loss: 0.4684 - val\_accuracy: 0.8658

Epoch 4/10

480/480 [==============================] - 95s 198ms/step - loss: 0.2435 - accuracy: 0.9296 - val\_loss: 0.3869 - val\_accuracy: 0.8693

Epoch 5/10

480/480 [==============================] - 92s 193ms/step - loss: 0.2142 - accuracy: 0.9381 - val\_loss: 0.2769 - val\_accuracy: 0.9162

Epoch 6/10

480/480 [==============================] - 97s 203ms/step - loss: 0.1894 - accuracy: 0.9457 - val\_loss: 0.2617 - val\_accuracy: 0.9118

Epoch 7/10

480/480 [==============================] - 93s 193ms/step - loss: 0.1711 - accuracy: 0.9497 - val\_loss: 0.3073 - val\_accuracy: 0.9053

Epoch 8/10

480/480 [==============================] - 94s 195ms/step - loss: 0.1563 - accuracy: 0.9538 - val\_loss: 0.2654 - val\_accuracy: 0.9111

Epoch 9/10

480/480 [==============================] - 102s 213ms/step - loss: 0.1506 - accuracy: 0.9549 - val\_loss: 0.2917 - val\_accuracy: 0.9093

Epoch 10/10

480/480 [==============================] - 95s 199ms/step - loss: 0.1325 - accuracy: 0.9582 - val\_loss: 0.2734 - val\_accuracy: 0.9096

<keras.callbacks.History at 0x27d1f92ad00>

#model.fit\_generator(x\_train,epochs=10,validation\_data=x\_test)

### Saving our model

# Save the model

model.save('ECG.h5')

### Predicting our results

from tensorflow.keras.models import load\_model

from keras.preprocessing import image

model = load\_model("ECG.h5") #loading the model for testing

img = image.load\_img(r"C:\Users\saket\OneDrive\Desktop\SAKETH\_ECG\New folder\ECG-main\Flask\uploads\PAC.png",target\_size= (64,64))#loading of the image

x = image.img\_to\_array(img)#image to array

x = np.expand\_dims(x,axis = 0)#changing the shape

#pred = model.predict\_classes(x)#predicting the classes

#pred

preds=model.predict(x)

pred=np.argmax(preds,axis=1)

preds

array([[0., 0., 1., 0., 0., 0.]], dtype=float32)

index=['Left Bundle Branch Block','Normal','Premature Atrial Contraction',

'Premature Ventricular Contractions', 'Right Bundle Branch Block','Ventricular Fibrillation']

result=str(index[pred[0]])

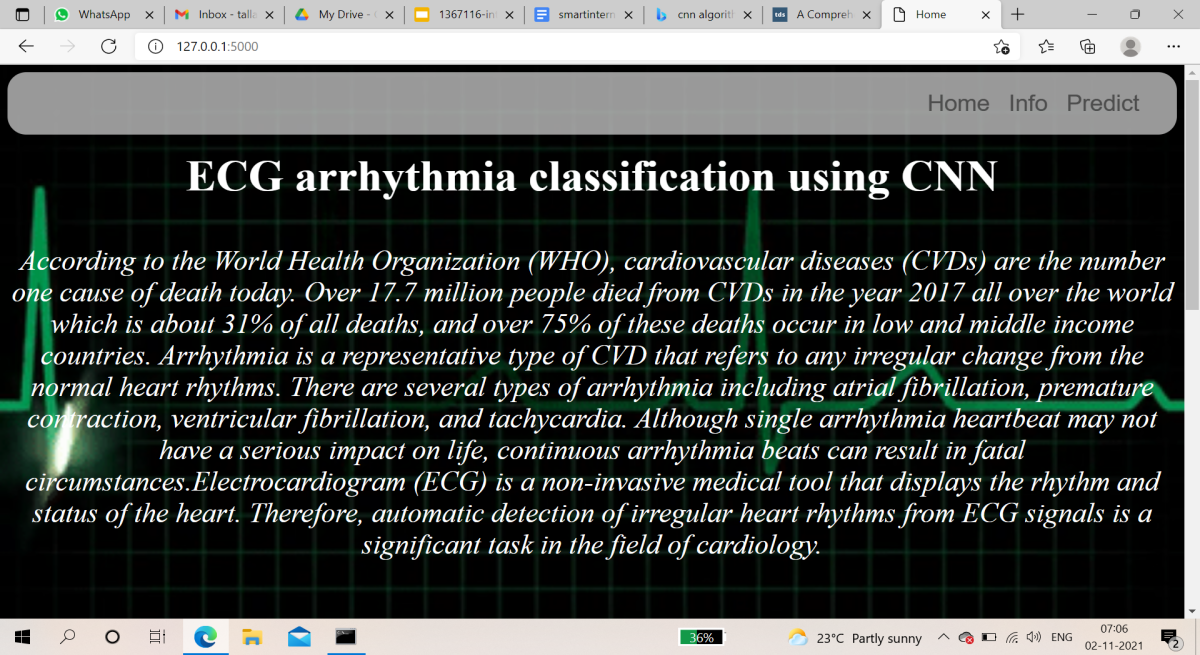
result

'Premature Atrial Contraction'

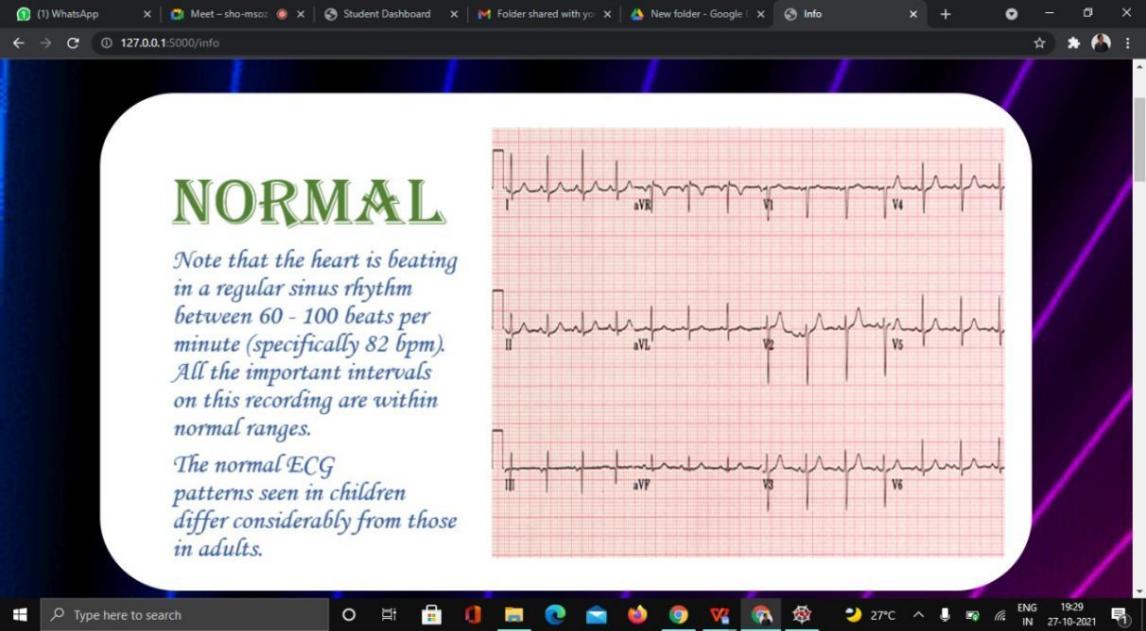
**Final output of the project:** In this we used Convolution Neural Network Deep Learning algorithm to predict arrhythmia detection .

the output can be seen as:

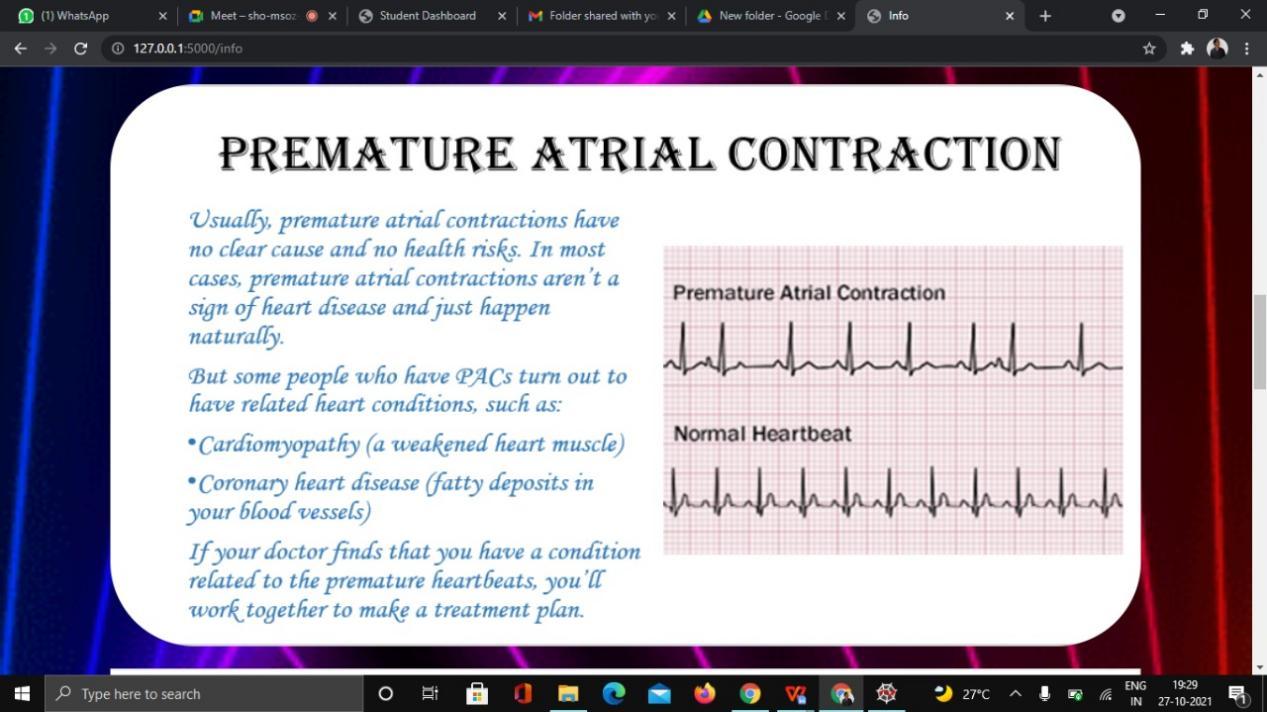
The home page can be shown as:



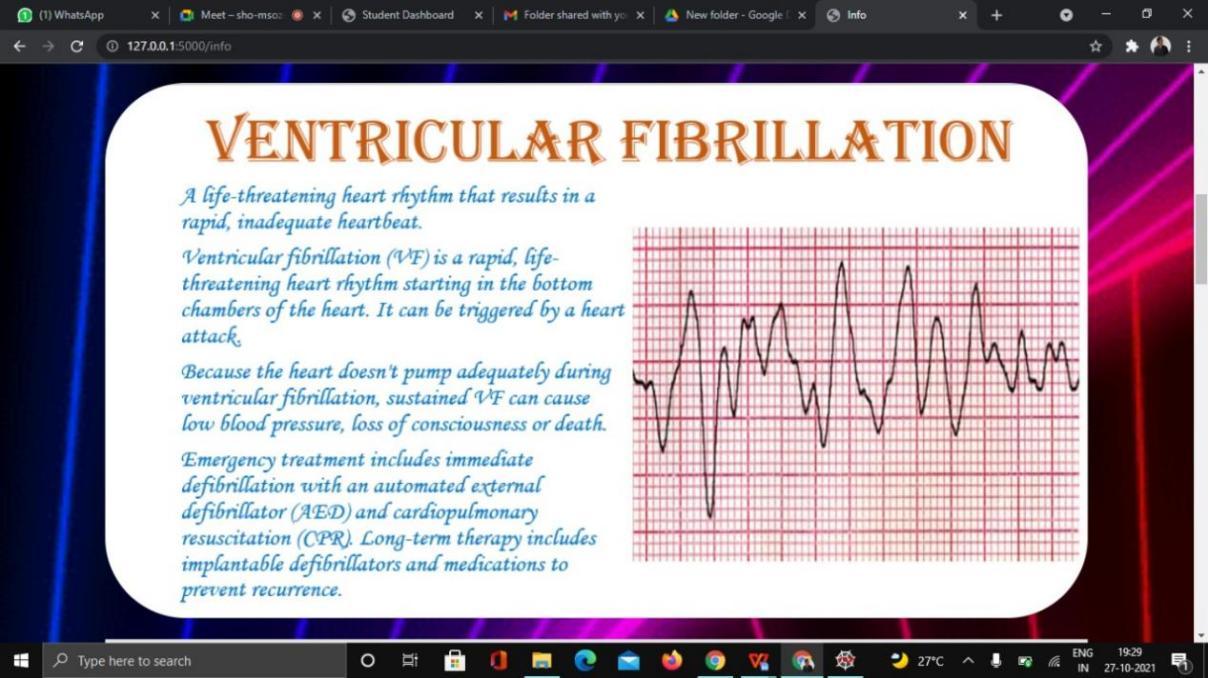
The info page about Normal:



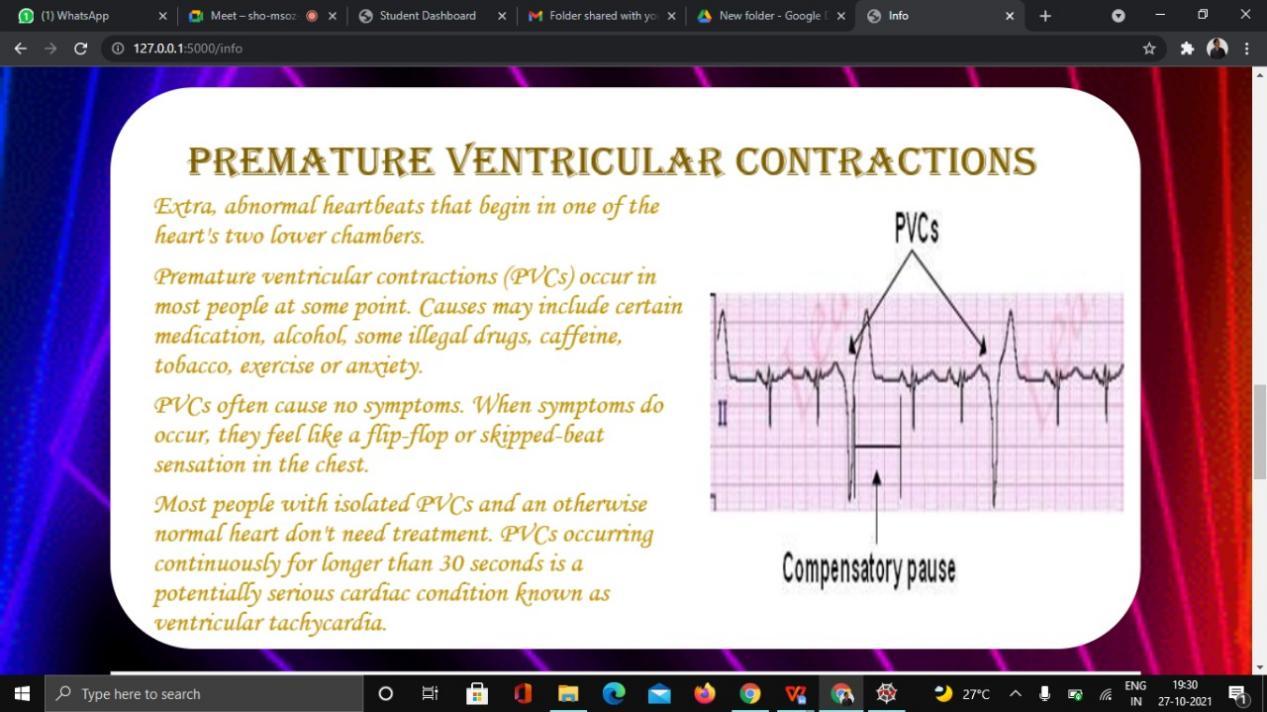
The info page about Premature Atrial Contraction:



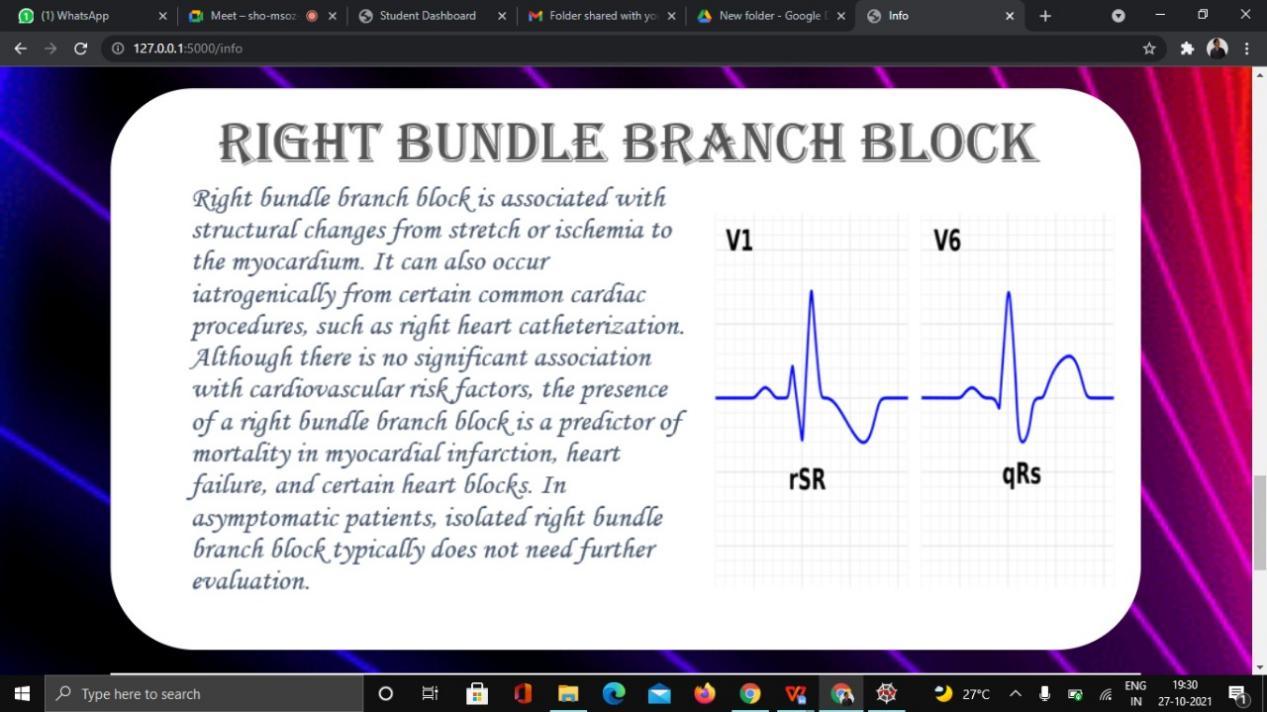
The info page about Ventricular Fibrillation:



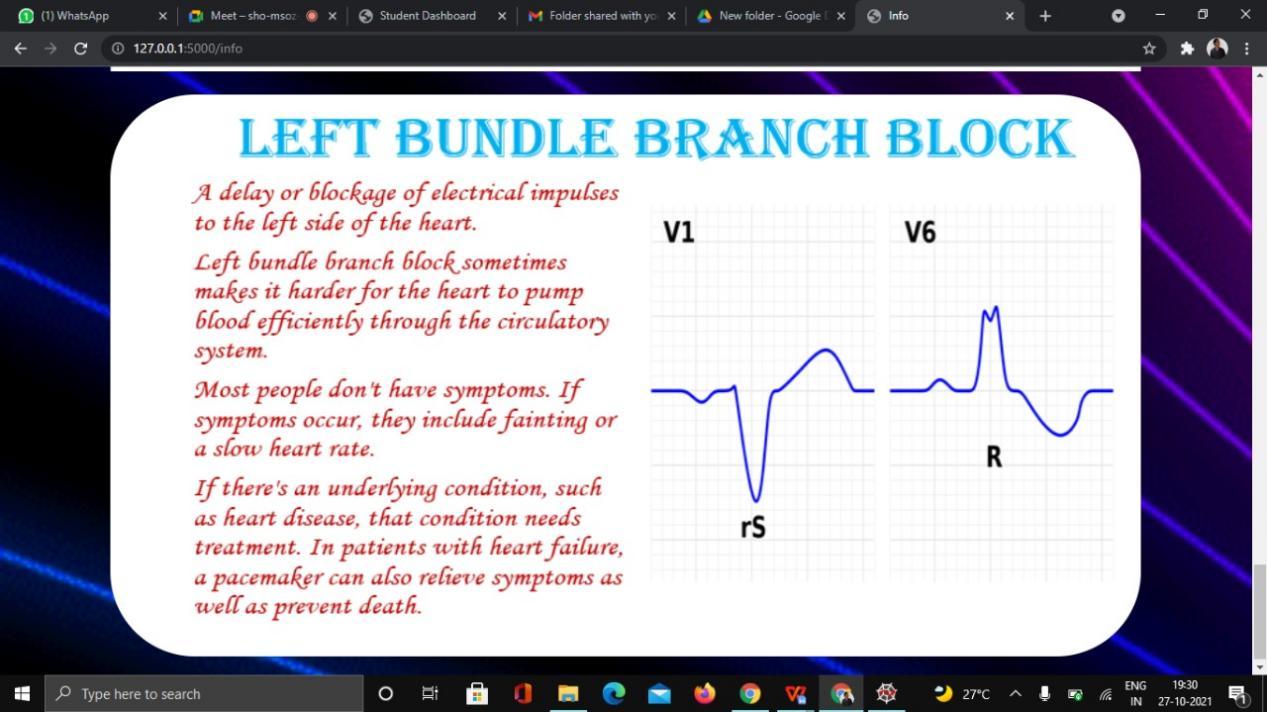
The info page about Premature Ventricular Contractions:



The info page about Right Bundle Branch Block:



The info page about Left Bundle Branch Block:



The result page is shown as : First we should choose a image as input from choose button and it will predict and classify the type of the image and shows the result:



Here we given RBBB image as input it predicted Right Bundle Branch Block as output.

**CHAPTER 7**

**ADVANTAGES AND DISADVANTAGES**

**Advantages:**

* ECG is helpful to measure three basic parameters of clinical interest viz. rhythm and heart rate, axis of the heart and state of myocardial muscle.
* ECG represents data in the topographic form which provides higher diagnostic information.
* ECG helps to prevent heart attacks by analyzing heart parameters at the initial stage.
* ECG is used to detect the cardiac conditions of the patients after surgical or any other operation and after application of anesthesia.
* ECG test is quick, painless and safe.
* ECG test is cheap in cost.

**Disadvantages:**

* It does not provide underlying heart problems for patients not having any symptoms.
* It does not always provide help in accurate diagnosis. More tests are needed to trace serious heart problems undetected by normal ECG curve.

**CHAPTER 8**

**APPLICATIONS**

**The areas where this solution can be applied:**

* In hospital and operation theaters.
* In laboratories for testing.

**CHAPTER 9**

**CONCLUSION**

**From this entire findings we know fundamental concepts and can work on IBM Watson and machine learning.**

* Gain a board understanding of Deep learning classification algorithms.
* Learn to build stunning models on IBM cloud.
* To create data visualizations to understand.

**CHAPTER 10**

**FUTURE SCOPE**

**Enhancements that can be made in the future:**

* This model can be further developed to suggest material among all the possible materials based on the input parameters.
* We can scope the better job in future with easy experience.

**CHAPTER 11**

**BIBILOGRAPHY**

References of previous works or websites visited/books referred for analysis about the project, previous solution findings and previous ECG documents.