Classification of Arrhythmia by Using Deep Learning with 2-D ECG Spectral Image Representation

Team ID: PNT2022TMID17913

Team Details

R.Harihara sudhan.

R. B. Ragul.

N. R. Somnath Babu.

P. Santosh.

S. Srivatsan.

1. INTRODUCTION

1.1 Project Overview

According to the World Health Organization (WHO), cardiovascular diseases (CVDs) is today's number one cause of death. Over a 17.7million people died from CVDs 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 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 severe 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 convolutional 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 webpage

1.2 Purpose

In the past few decades, Deep Learning has proved to be a compelling tool because of its ability to handle large amounts of data. The interest to use hidden layers has surpassed traditional techniques, especially in pattern recognition. One of the most popular deep neural networks is Convolutional Neural Networks. In this project, we develop a convolutional neural network (CNN) based method for electrocardiogram (ECG) arrhythmia classification. Using deep two-dimensional CNN and grayscale ECG images, we divide the ECG into seven categories, one of which is normal and the other six of which are various types of arrhythmia.

2. LITERATURE SURVEY

2.1 Existing problem

Cardiovascular diseases (CVDs) are the number one cause of death today. Over 17.7 million people died from CVDs 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 normal heart rhythms. There are several types of arrhythmia including atrial fibrillation, premature contraction, ventricular fibrillation, and tachycardia.

2.2 References

Title of the work	Abstract
Towards Interpretable Arrhythmia Classification with Human-machine Collaborative Knowledge Representation	The first unique interpretable arrhythmia classification strategy based on a human-machine collaborative knowledge representation is proposed in this research as a solution to this flaw. This method uses an AutoEncoder to split the expertise of hand-encoding and machine-encoding ECG data into two portions. The encoded information is then placed into a classifier to categories arrhythmia heartbeats with or without a person in the loop (HIL).
Arrhythmia Classification Using Biased Dropout and Morphology-Rhythm Feature With Incremental Broad Learning	In the proposed strategy, authors develop an incremental broad learning (IBL) classification model for arrhythmia-type detection based on the biassed dropout method. In the ECG signal preprocessing, they extract the morphological-rhythm properties of the denoised signal as the input data of the IBL model. By utilising better features, the IBL model improves the node optimization model's classification impact.
Multi-Label Classification of Arrhythmia for Long-Term Electrocardiogram Signals With Feature Learning	This paper aims: 1) to propose a multi-label feature selection method based on ECG (MS-ECG) and design an evaluation criterion of ECG features based on kernelized fuzzy rough sets so as to choose the optimal feature subset and optimize ECG feature space and 2) to propose the multi-label classification algorithm of arrhythmia based on ECG (MC-ECG) by establishing a multi objective optimization model.
Evaluation of performance of Cloud based Neural Network models on Arrhythmia Classification	In this paper, we propose an evaluation of different neural network models. The signal is transformed into the wavelet domain, and noise removal is

	carried out by wavelet de-noising post filtering. The features are extracted from the processed signal and are transmitted to the cloud where predictive models are applied to the extracted features to predict the class of arrhythmia thus aiding the medical diagnostic process.
ECG Arrhythmia Classification Using Relevance Vector Machine	In this paper, a experimental study was conducted to achieve the maximum accuracy the RVM classifier design by searching for the best value of the parameters that its discriminant function, and upstream by looking for the best subset of features that feed the classifier.
Automatic Detection of Cardiac Arrhythmia Classification Using Deep Learning Techniques	The goal of this paper is to apply deep learning techniques in the diagnosis of cardiac arrhythmia using ECG signals with minimal possible data pre-processing. We employ convolutional neural network (CNN), recurrent structures such as recurrent neural network (RNN), long short-term memory (LSTM) and gated recurrent unit (GRU) and hybrid of CNN and recurrent structures to automatically detect the abnormality.
Cardiac arrhythmia detection using deep learning	In this study, a deep learning framework previously trained on a general image data set is transferred to carry out automatic ECG arrhythmia diagnostics by classifying patient ECG's into corresponding cardiac conditions. Transferred deep convolutional neural network (namely AlexNet) is used as a feature extractor and the extracted features are fed into a simple back propagation neural network to carry out the final classification. Three different conditions of ECG waveform are selected from MIT-BIH arrhythmia database to evaluate the proposed framework.

Classification of Arrhythmia by Using Deep Learning with 2-D ECG Spectral Image Representation	This paper proposes a two-dimensional(2-D) convolutional neural network (CNN) model for the classification of ECG signals into eight classes.The one-dimensional ECG time series signals are transformed into 2-D spectrograms through short-time Fourier transform. The 2-D CNN model consisting of four convolutional layers and four pooling layers is designed for extracting robust features from the input spectrograms.Our proposed methodology is evaluated on a publicly available MIT-BIH arrhythmia dataset. We achieved a state-of-the-art average classification accuracy of 99.11%.
Classification of Arrhythmia in Heartbeat Detection Using Deep Learning	This paper aims to apply deep learning techniques on the publicly available dataset to classify arrhythmia. We have used two kinds of the dataset in our research paper. One dataset is the MIT-BIH arrhythmia database, with a sampling frequency of 125 Hz with 1,09,446 ECG beats. The classes included in this first dataset are N, S, V, F, and Q. The second database is PTB Diagnostic ECG Database. The second database has two classes. The techniques used in these two datasets are the CNN model, CNN+ LSTM, and CNN+ LSTM + Attention Model.

ECG Arrhythmia Classification Using STFT-based Spectrogram and Convolutional Neural Network	This paper proposes an ECG arrhythmia classification method using two-dimensional deep convolutional neural network. The time domain signals of ECG, belonging to five heartbeat types including normal beat, left bundle branch block beat, right bundle branch block beat, premature ventricular contraction beat, and atrial premature contraction beat, were firstly transformed into time-frequency spectrograms by short-time Fourier transform. Subsequently, the spectrograms of the five arrhythmia types were utilized as input to the 2D-CNN such that the ECG arrhythmia types were identified and classified finally.

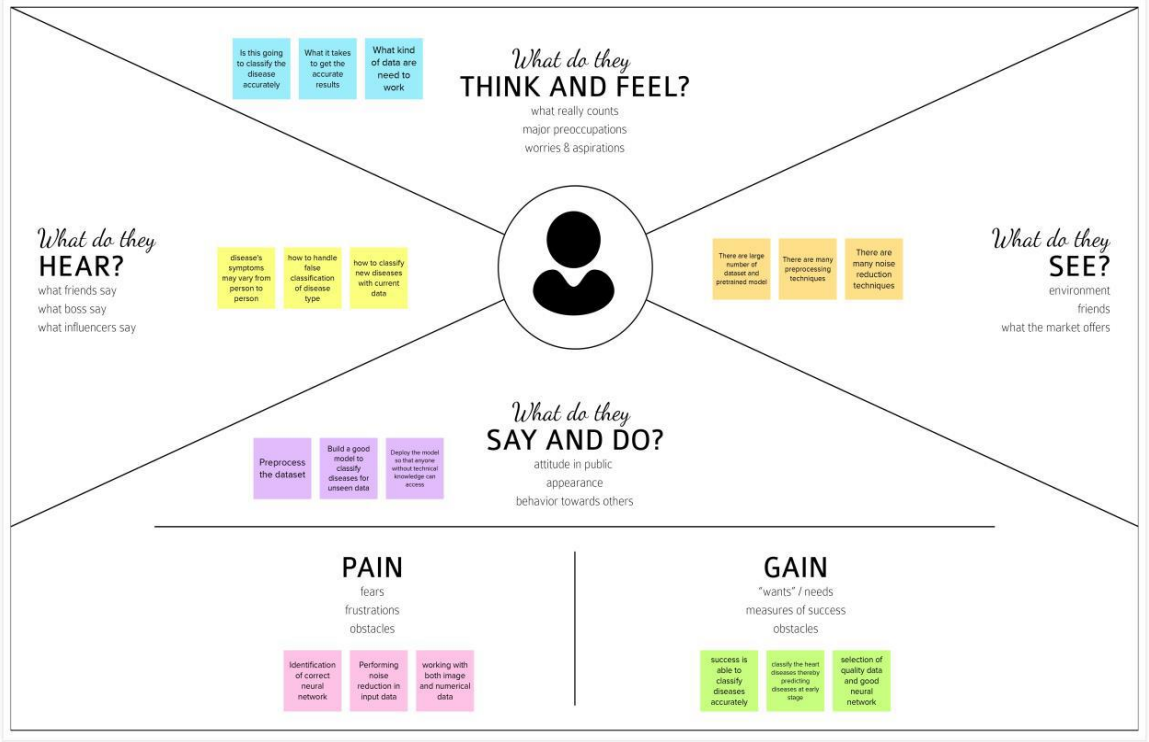
2.3 Problem Statement Definition

The Electrocardiogram (ECG) is one of the most frequently used in the identification and prediction of cardiovascular disorders (CVDs). The irregular heartbeats, or arrhythmias as they are often called, can be recorded in the ECG readings. For accurate diagnosis of patients' acute and chronic cardiac problems, an in-depth examination of ECG signals is essential. which needs a lot of time or an expert surgeon to do. The patient may have a costly hospitalization or a delayed start to their therapy as a result.

3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas

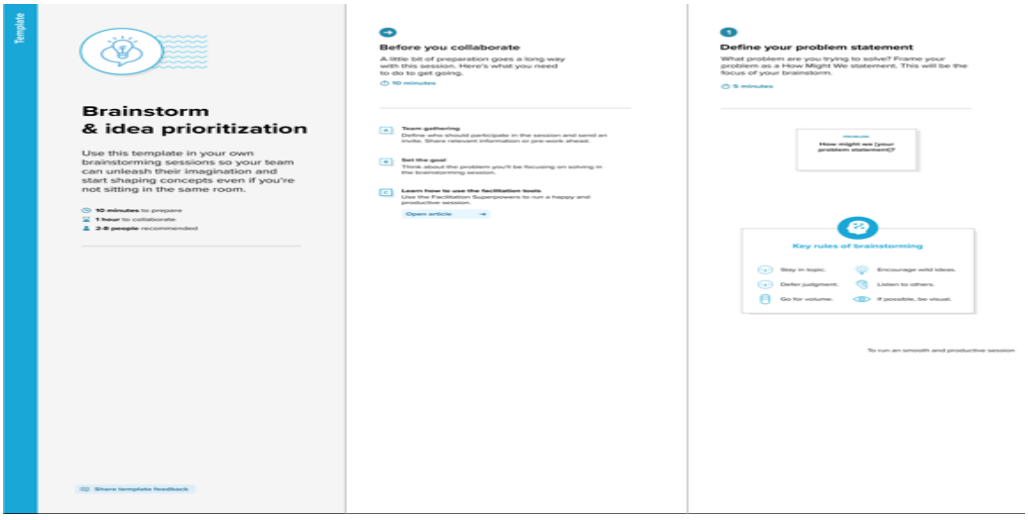
An empathy map is a simple, easy-to-digest visual that captures knowledge about a user’s behaviors and attitudes. It is a useful tool to help teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user’s perspective along with his or her goals and challenges.



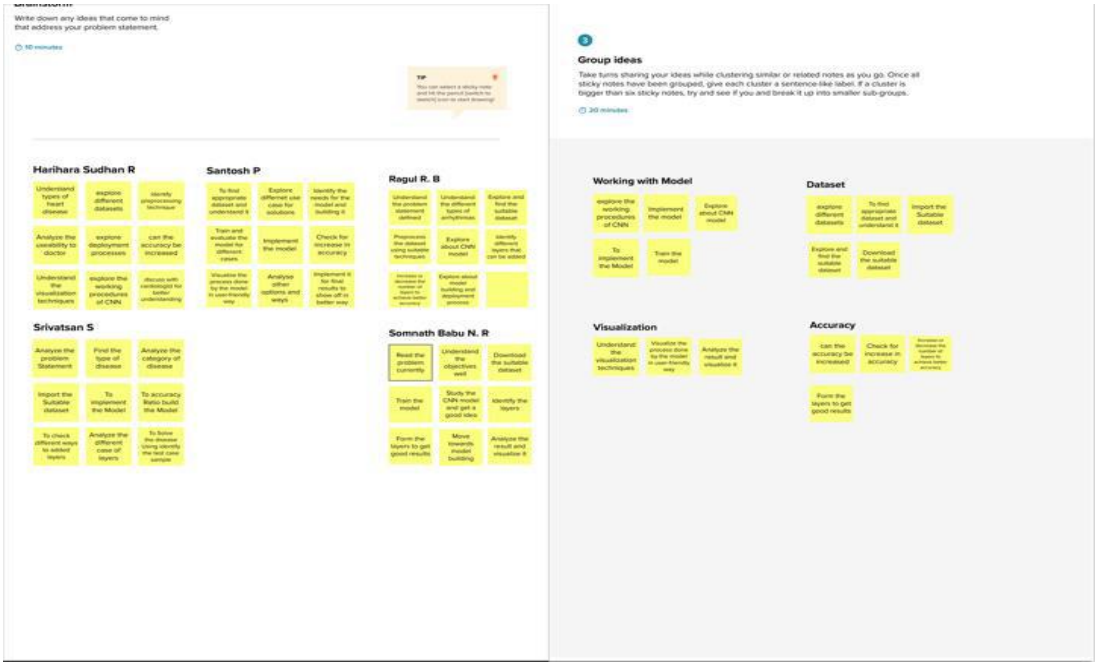
3.2 Ideation & Brainstorming

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem-solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich number of creative solutions.

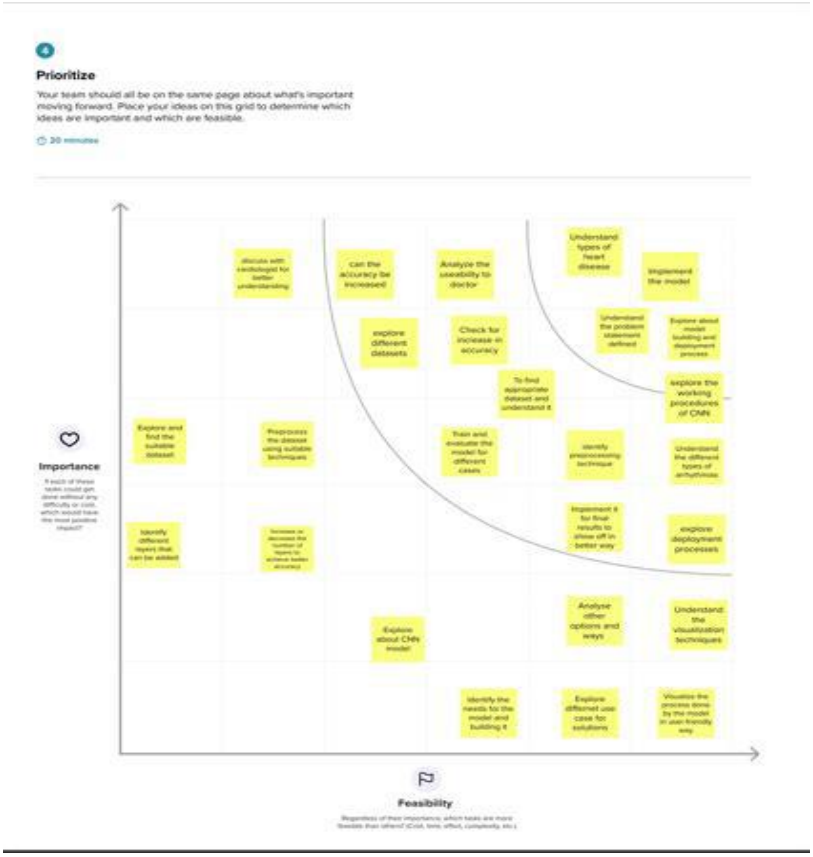
Step-1: Team Gathering, Collaboration and Select the Problem Statement



Step-2: Brainstorm, Idea Listing and Grouping



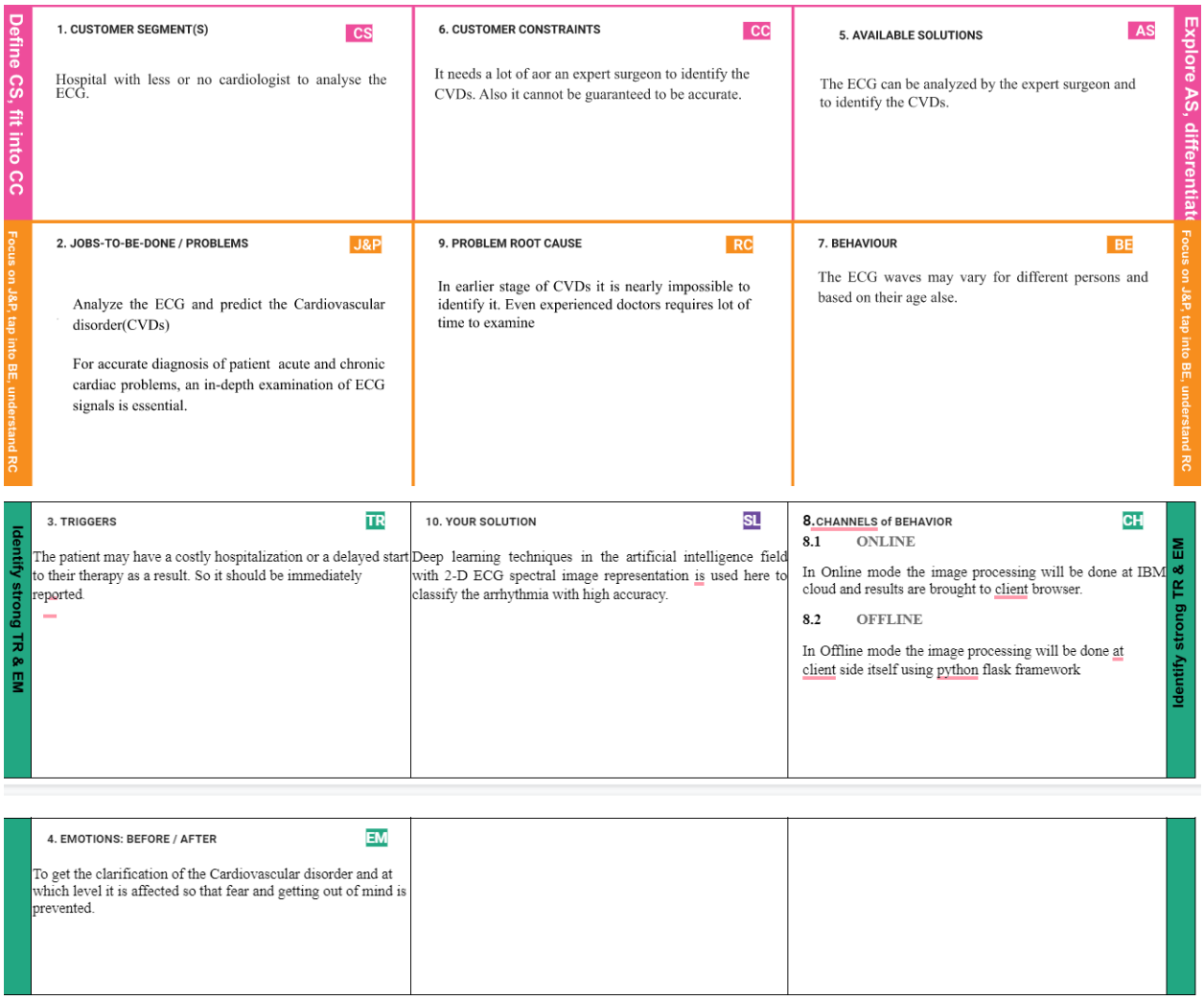
Step-3: Idea Prioritization



3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	The Electrocardiogram (ECG) is one of the signals that is most frequently utilised in the identification and prediction of cardiovascular disorders (CVDs). The irregular heartbeats, or arrhythmias as they are often called, can be recorded in the ECG readings. For accurate diagnosis of patients' acute and chronic cardiac problems, an in-depth examination of ECG signals is essential. which needs a lot of time or an expert surgeon to do. The patient may have a costly hospitalisation or a delayed start to their therapy as a result.
2.	Idea / Solution Description	We develop a convolutional neural network (CNN) based method for electrocardiogram (ECG) arrhythmia classification. Using deep two-dimensional CNN and grayscale ECG images, we divide the ECG into six categories, one of which is normal and the other five of which are various types of arrhythmia.
3.	Novelty / Uniqueness	We have done image augmentation in preprocessing step.
4.	Social Impact / Customer Satisfaction	Customer/Patient need not wait for an expert surgeon and he/she need not spend for hospitalization. He/she can use the app and upload the image and find the type of disease and take the necessary measures.
5.	Business Model (Revenue Model)	This application can be used by more people for earlier classification of arrhythmia.
6.	Scalability of the Solution	As this application is deployed in IBM cloud it is more scalable.

3.4 Problem Solution fit



4. REQUIREMENT ANALYSIS

4.1 Functional requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	**ECG image upload**	Patient is able to upload the ECG via the Flask app. The doctor is able to upload the ECG via the Flask app.
FR-2	**Noise reduction**	Convert the ECG into grayscale. Reduce the noise in the image using AutoEncoder
FR-3	**CVD prediction**	CVD prediction can be done at a local machine. CVD prediction can be done in the IBM cloud.
FR-4	**Result presentation**	CVD prediction must be presented to the end users.

		CVD prediction must be presented in detail to the doctor.

4.2 Non-Functional requirements

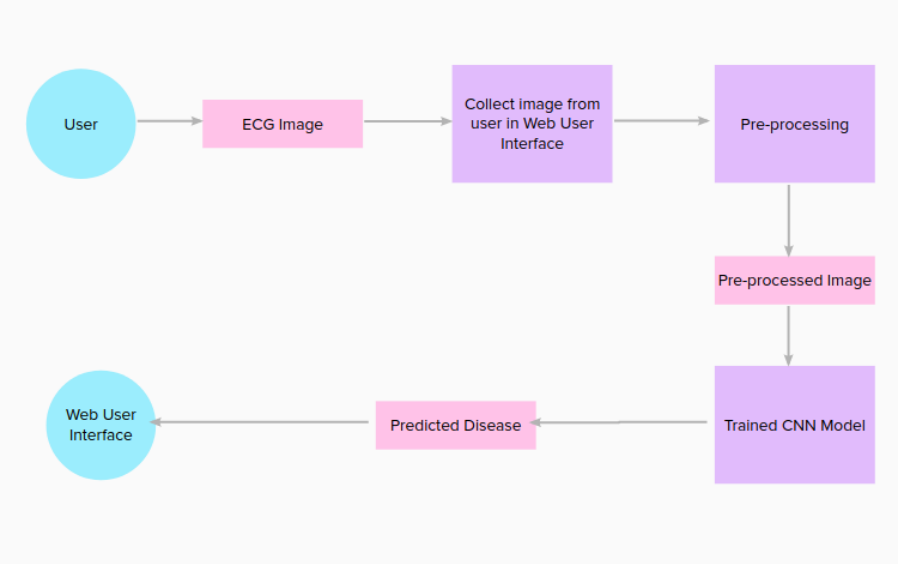
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	It should be convenient to use by anybody. It must be user-friendly.
NFR-2	**Security**	The analysed results should be kept confidential. Leakage of the results of the patient will be dangerous and a crime.
NFR-3	**Reliability**	It should be reliable and should not cause any faults or failures after a certain period of time.
NFR-4	**Performance**	The analyses of the ECGs should have a good accuracy level compared to expert surgeons who analyse.
NFR-5	**Availability**	It should be available 24/7. As there can be an emergency situation for a patient.
NFR-6	**Scalability**	There can be one patient also there can be many patients and many ECGs to be analysed. It should be scalable.

5. PROJECT DESIGN

5.1 Data Flow Diagrams

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



5.2 Solution & Technical Architecture

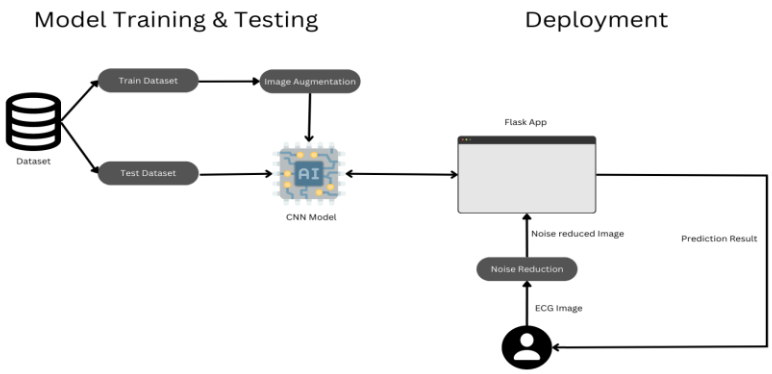


Table-1 : Components & Technologies:

S. No.	Component	Description	Technology
1.	User Interface	The user can interact via Web UI.	HTML, CSS, JavaScript / Angular Js / React Js etc.
2.	Train & Test dataset split	The dataset will be divided into training and testing datasets.	Python, Keras.
3.	Image Augmentation	Image Augmentation will be done on the training dataset.	Python, ImageDataGenerator.

4.	CNN Model	The core image processing will be done in this CNN model.	Python, Tensorflow

Table-2: Application Characteristics:

S. No.	Characteristics	Description	Technology
1.	Open-Source Frameworks	Flask, Tensorflow, python	python, CUDA, HTML, CSS
2.	Scalable Architecture	The Application will be deployed in IBM cloud. The three tiers are IBM cloud, Web service and can be accessible using the Internet.	IBM cloud.
3.	Availability	Since the application is deployed in a cloud environment it ensures the availability.	IBM cloud
4.	Performance	Tensorflow ensures the performance of the model by using GPU in Host system.	Python, CUDA, IBM cloud

5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Patient/Doctor (Web user)	Web app	USN-1	As a user, can access the web application	I can open the website	High	Sprint-1
Patient/Doctor (Web user)	Dashboard	USN-2	As a user, can see the home page / dashboard	I can see the homepage of the website	High	Sprint-1
Patient/Doctor (Web user)	Home page	USN-3	As a user, can see the article	I can read the detail about the project	Low	Sprint-1

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Patient/Doctor (Web user)	Navigate between pages	USN-4	As a user, can see navigation buttons	I can see the navigation buttons	Medium	Sprint-2
Patient/Doctor (Web user)	Info page	USN-5	As a user, can click on the “info” button	I can click on the “info” button	Medium	Sprint-2
Patient/Doctor (Web user)	Predict page	USN-6	As a user, can click on the “prediction” button	I can click on the “prediction”	High	Sprint-3
Patient/Doctor (Web user)	Get to know about heart disease	USN-7	As a user, can see the the CVDs details	I can read the article of CVD types	Medium	Sprint-3
Patient/Doctor (Web user)	Prediction page	USN-8	As a user, can see the options available	I can see the facility of this page	High	Sprint-4
Patient/Doctor (Web user)	Image upload	USN-9	As a user, can upload the image in this page	I can upload the image in this page	High	Sprint-4
Patient/Doctor (Web user)	Result page	USN-10	As a user, can see the type of CVD classified for the image uploaded	I can see the type of CVD classified	High	Sprint-4

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Home Page	USN-1	As a user, I can view the home page of the web application.	15	Low	P. Santosh , Srivatsan

Sprint-2	Info Page	USN-2	As a user, I can view the information regardingthe heart disease.	15	Medium	N R Somnath Babu
Sprint-3	Prediction Page	USN-3	As a user, I can insert the scan images toclassify the arrhythmia	15	High	R B Ragul
Sprint-4	Prediction	USN-4	As a user, I expect the application to classify thelevel of arrhythmia the patient has. It must be accurate.	15	High	R Hariharasudhan

Project Tracker, Velocity & Burndown Chart:

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	15	6 Days	2 Oct 2022 4	2 Oct 2022 9	15	2 Oct 2022 9
Sprint-2	15	6 Days	3 Oct 2022 1	0 Nov 2022 5	15	0 Nov 2022 5
Sprint-3	15	6 Days	0 Nov 2022 7	1 Nov 2022 2	15	1 Nov 2022 2
Sprint-4	15	6 Days	1 Nov 2022 4	1 Nov 2022 9	15	1 Nov 2022 9

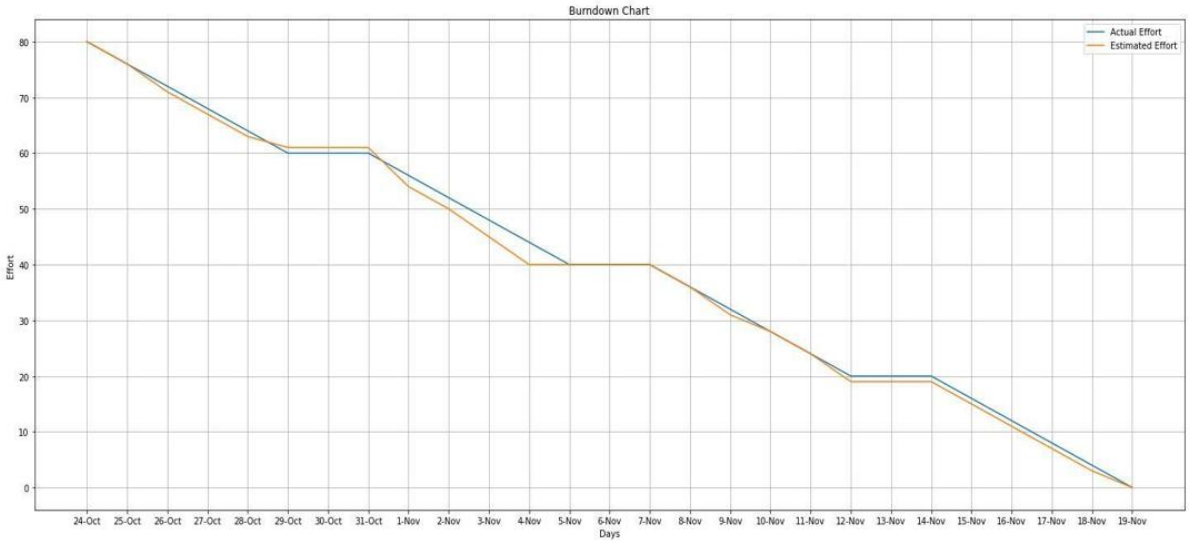
Velocity:

Imagine we have a 6-day sprint duration, and the velocity of the team is 15 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$\text{Average Velocity} = 15 / 6 = 2.5$$

Burndown Chart:

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

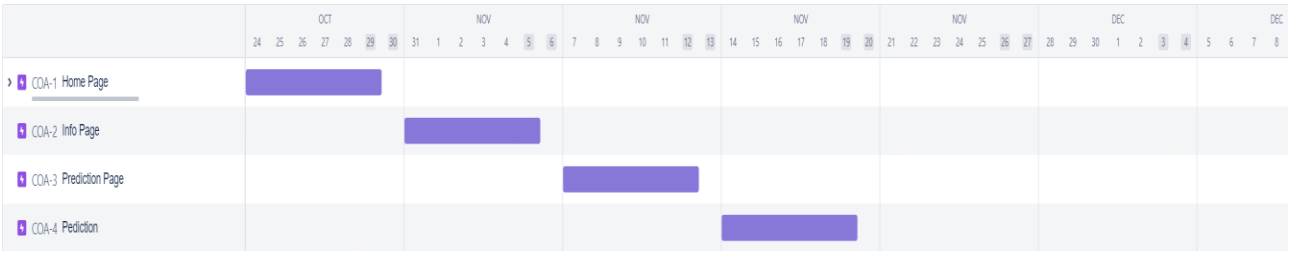


6.2 Sprint Delivery Schedule

7. Title	Description	Date
Literature Survey and Information Gathering	Gathering Information by referring the technical papers, research publications etc.	3 September 2022
Prepare Empathy Map	To capture user pain and gains Prepare List of Problem Statement	10 September 2022
Ideation	Prioritize a top 3 ideas based on feasibility and Importance	17 September 2022
Proposed Solution	Solution include novelty, feasibility, business model, social impact and scalability of solution	24 September 2022
Problem Solution Fit	Solution fit document	1 October 2022
Solution Architecture	Solution Architecture	1 October 2022

Customer Journey	To Understand User Interactions and experiences with application	8 October 2022
Functional Requirement	Prepare functional Requirement	12 October 2022
Data flow Diagrams	Data flow diagram	12 October 2022
Technology Architecture	Technology Architecture diagram	12 October 2022
Milestone & sprint delivery plan	Activity what we done &further plans	22 October 2022
Project Development- Delivery of sprint 1,2,3 &4	Develop and submit the developed code by testing it	24 October 2022 – 19 November 2022

7.1 Reports from JIRA

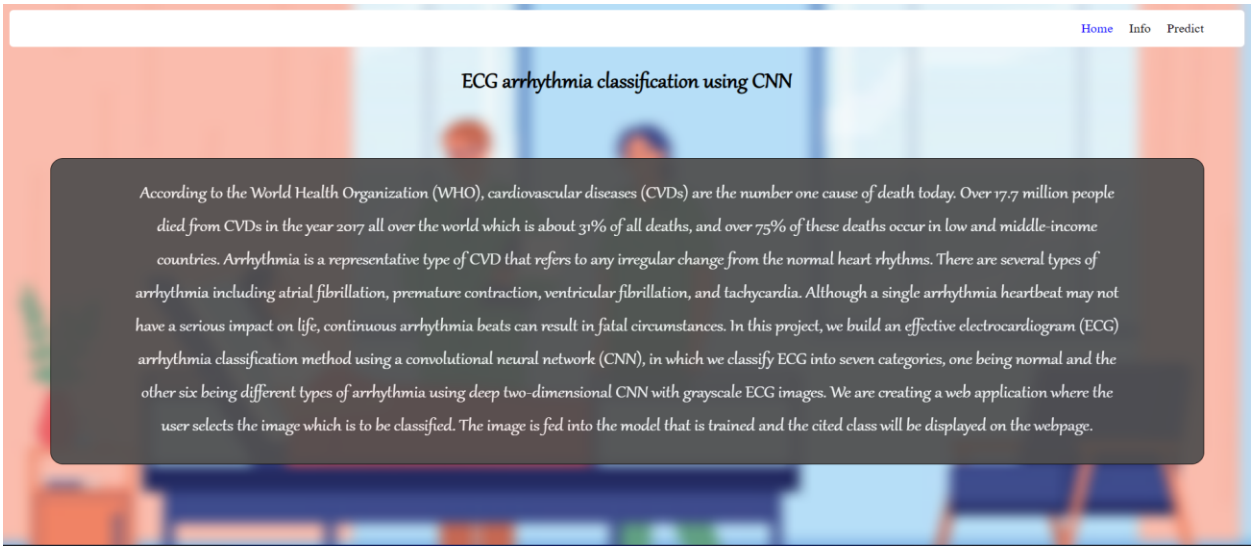


8. CODING & SOLUTIONING (Explain the features added in the project along with code)

8.1 Home page

Displays the Home page of the application. This feature aimed to give overview about the project and heart diseases.

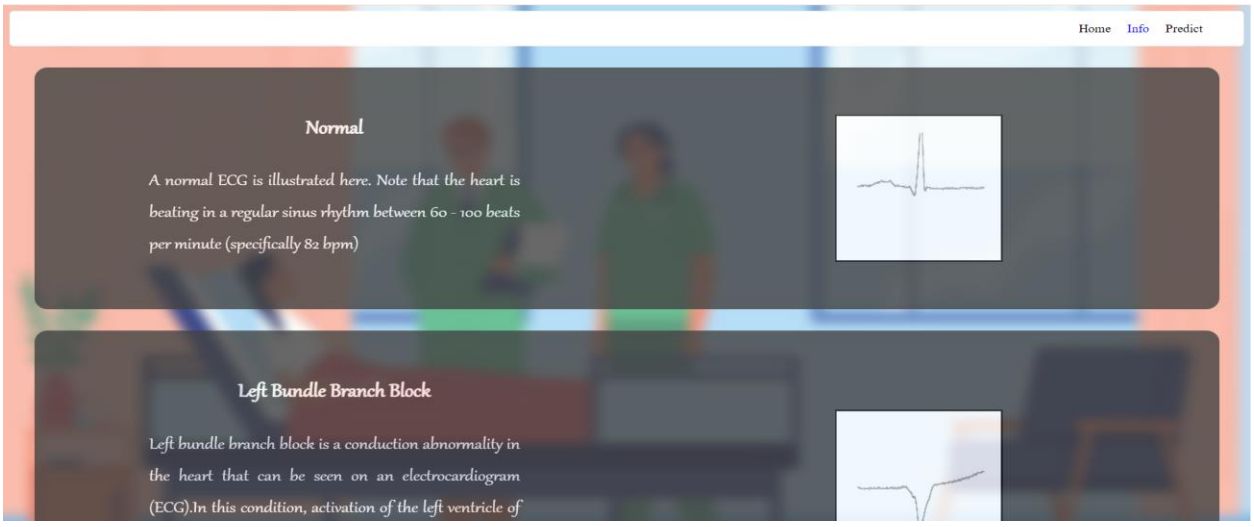
Output:



8.2 Info page

Info page is aimed to give a complete overview of all types of cardiovascular diseases with its ECG image.

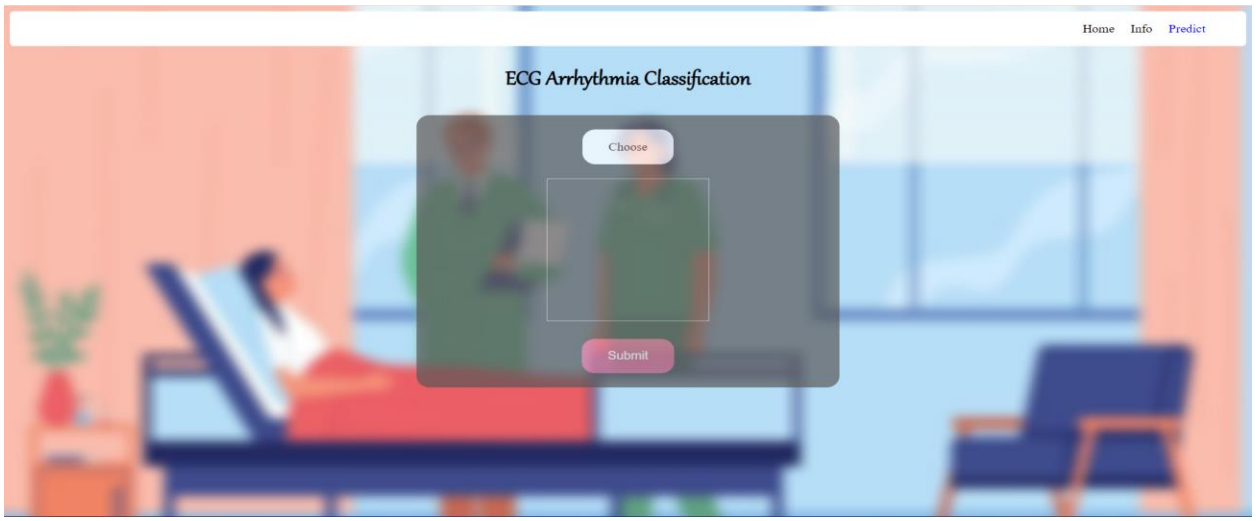
Output:



8.3 Predict page

This feature of the project is used to upload the ECG images to the server for prediction.

Output:



8.4 CVD predictor

This feature of the project is aimed to give a fault tolerant CVD predict no matter how this module will predict the CVD and gives output.

Code:

```
from ibm_watson_machine_learning import APIClient
from tensorflow.keras.preprocessing import image
import numpy as np
from tensorflow.keras.models import load_model


class PredictCVD:
    def __init__(self):
        self.__image_to_predict = None
        self.__CVDs = [
            'left_bundle_branch_block.html',
            'normal.html',
            'premature_atrial_contraction.html',
            'premature_ventricular_contractions.html',
            'right_bundle_branch_block.html',
            'ventricular_fibrillation.html'
        ]
        wml_credential = {
            'url': 'https://eu-de.ml.cloud.ibm.com',
            'apikey': 'Ndj3UnNat-ZW_rOzehbxfzaMRt5DfuLeISJfNW_vuebX'
        }
        self.__deployment_id = "b48cd709-a43a-45af-8f80-8535affa8849"
        try:
            self.__client = APIClient(wml_credential)
            self.__run_locally = False
            self.__local_model = load_model("model_30_oct_22.h5")
            self.__client.set.default_space("4ecc4a52-91f7-4d31-a885-
6851624e76ec")
        except Exception:
            print("Unable to communicate with IBM cloud")
            print("Preparing to run model locally")
            self.__run_locally = True
```

```
            self.__local_model = load_model("model_30_oct_22.h5")

    def predict(self, image_path) -> str:
        self.__image_to_predict = np.expand_dims(image.img_to_array(
            image.load_img(image_path, target_size=(64, 64),
color_mode="grayscale")), axis=0)
        if self.__run_locally:
            return self.__predict_cvd_locally()
        else:
            try:
                return self.__predict_cvd_cloud()
            except Exception as e:
                self.__run_locally = True
                self.__local_model = load_model("model_30_oct_22.h5")
                return self.__predict_cvd_locally()

    def __predict_cvd_locally(self):
        print("Prediction starts in Local Machine")
        prediction = self.__local_model.predict(self.__image_to_predict)
        return self.__CVDs[list(prediction[0]).index(1)]

    def __predict_cvd_cloud(self):
        print("Prediction starts in Cloud")
        scoring_payload = {
            self.__client.deployments.ScoringMetaNames.INPUT_DATA: [{
                "values": self.__image_to_predict
            }]
        }
        class_value =
self.__client.deployments.score(deployment_id=self.__deployment_id,

meta_props=scoring_payload)['values'][1]
        return self.__CVDs[class_value]
```

8.5 Flask App

This is the server of the application which serves all the html pages and gets the ECG images as input and predict the CVD using Deep learning technique.

Code:

```
import os
import numpy as np
from flask import Flask, request, render_template, send_from_directory,
make_response
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
from predict_cvd import PredictCVD

CVDs = [
    'Left Bundle Branch Block',
    'Normal',
    'Premature Atrial Contraction',
    'Premature Ventricular Contractions',
    'Right Bundle Branch Block',
    'Ventricular Fibrillation'
]
```

```
app = Flask(__name__)


@app.route("/")
@app.route("/index.html")
def home():
    return render_template("index.html")


@app.route("/info.html")
def info():
    return render_template("info.html")


@app.route("/predict.html", methods=['GET', 'POST'])
def predict():
    if request.method == 'POST':
        print(request.files)
        request.files['image'].save("ecg.png")
        predictor = PredictCVD()
        response = make_response("response")
        response.data = "Hello"
        print(predictor.predict("ecg.png"))
        return render_template(""+predictor.predict("ecg.png"))
    return render_template("predict.html")


@app.route('/favicon.ico')
def favicon():
    return send_from_directory(os.path.join(app.root_path, 'static'),
                               'favicon.ico',
mimetype='image/vnd.microsoft.icon')


if __name__ == "__main__":
    app.run(debug=True)
```

9. TESTING

9.1 Test Cases

1	Verify user is able to see Home page
2	Verify user can navigate Info Page
3	Verify user can navigate Predict Page
4	Verify user can see Info page
5	Verify user can see Predict page
6	Verify user can upload ECG image
7	Verify user can redirect to predicted result page

9.2 User Acceptance Testing

Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and howthey were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	2	3	1	1	7
Duplicate	1	0	3	0	4
External	2	0	0	1	3
Fixed	2	2	1	2	7
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	2	2	1	5
Totals	7	7	9	6	29

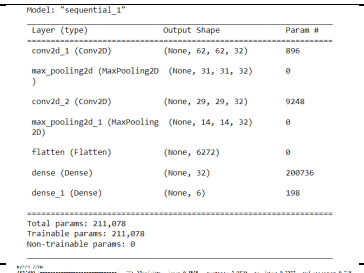
Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Predict Page	7	0	0	7
Info Page	5	0	0	5
Home Page	4	0	0	4
CNN ECG	3	0	0	3
Flask	9	0	0	9
Model Building	4	0	0	4
Train model on IBM	8	0	0	8
Final Report Output	2	0	0	2

10. RESULTS

10.1 Performance Metrics

S.No.	Parameter	Values	Screenshot
1.	Model Summary	-	
2.	Accuracy	Training Accuracy - 96% Validation Accuracy - 90%	

Model Summary:

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_1 (Conv2D)           (None, 62, 62, 32)        896

 max_pooling2d (MaxPooling2D  (None, 31, 31, 32)       0
 )

 conv2d_2 (Conv2D)           (None, 29, 29, 32)        9248

 max_pooling2d_1 (MaxPooling  (None, 14, 14, 32)       0
 2D)

 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
_________________________________________________________________
```

Model Accuracy:

```
Epoch 7/10
480/480 [==============================] - 34s 70ms/step - loss: 0.1545 - accuracy: 0.9530 - val_loss: 0.2707 - val_accuracy: 0.918
7
Epoch 8/10
480/480 [==============================] - 33s 68ms/step - loss: 0.1362 - accuracy: 0.9580 - val_loss: 0.3744 - val_accuracy: 0.885
9
Epoch 9/10
480/480 [==============================] - 34s 71ms/step - loss: 0.1344 - accuracy: 0.9585 - val_loss: 0.3191 - val_accuracy: 0.902
7
Epoch 10/10
480/480 [==============================] - 33s 69ms/step - loss: 0.1201 - accuracy: 0.9633 - val_loss: 0.3093 - val_accuracy: 0.906
8
```

11. ADVANTAGES & DISADVANTAGES

Advantages:

- Cost efficient.
- Accurate in predict of CVDs.
- User friendly

Disadvantages:

- Doesn’t maintain any records for prediction.
- No authentication.
- Unable to process video data.

12. CONCLUSION

Application was built using various web technologies such as HTML, CSS, Flask, etc. and model was also build using Machine Learning Techniques such as Data Preprocessing and model building using TensorFlow and other packages to classify arrhythmia accurately and time-efficiently.

13. FUTURE SCOPE

This project only aimed to process image data. In future a system can be developed with live ECG video processing for real-time and quick response. Sometime noise may be present in prediction images that can be eliminated by employing noise reduction AutoEncoder.

14. APPENDIX

GitHub & Project Demo Link: https://github.com/IBM-EPBL/IBM-Project-2512-1658473101.git