

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

Team Id : PNT2022TMID09656

## **Team Members**

1. RAHUL C
2. VIGNESH P
3. SURAVARAPU SAI CHARAN REDDY
4. MYTHIRESH G

## **CONTENTS**

### **INTRODUCTION**

Project Overview  
Purpose

## **LITERATURE SURVEY**

Existing problem

References

Problem Statement

Definition

## **IDEATION & PROPOSED SOLUTION**

Empathy Map Canvas

Ideation & Brainstorming

Proposed          Solution

Problem Solution fit

## **REQUIREMENT ANALYSIS**

Functional requirement

Non-Functional  
requirements

## **PROJECT DESIGN**

Data Flow Diagrams

Solution & Technical  
Architecture

User Stories

## **PROJECT PLANNING & SCHEDULING**

Sprint Planning &

Estimation Sprint Delivery

Schedule

Reports from JIRA

## **CODING & SOLUTIONING (Explain the features added in the project along with code)**

Feature 1

Feature 2

Database Schema (if  
Applicable)

## **TESTING**

Test Cases

User Acceptance Testing

## **RESULTS**

Performance Metrics

**ADVANTAGES& DISADVANTAGES CONCLUSION FUTURE SCOPE 13.  
APPENDIX**

Source Code

GitHub & Project Demo

Link

## **Hardware Requirements:**

Processor : Intel Core i5HDD: 1TB

RAM: Minimum 2GB;

Recommended 4GB

## **Software Requirements:**

Operating system : Windows 10 Dataset: IAM

Dataset(Words, Lines)Programming Language:

Python

Numpy : Core package providing powerful tools to  
manipulate data arrays, such as our character images.

OpenCV : OpenCV is a large open-source library for image processing, character  
recognition, and machine learning. It can scan handwritten images.

Autocorrect : It is used to correct the spelling. It supports many languages.

Tensorflow : Tensorflow is the core open source library to help you develop and train  
Machine Learning models.

### **Survey Papers:**

#### **Paper 1:**

Author Name : Zhao Y, Cheng, J

Title : ECG classification using deep CNN improved by wavelet transform

Publication website : <https://opus.lib.uts.edu.au/handle/10453/142924>

Published Date : 2020-06-30

Objective: Atrial fibrillation is the most common persistent form of  
arrhythmia. A method based on wavelet transform combined with deep

convolutional neural network is applied for automatic classification of electrocardiograms. Since the ECG signal is easily inferred, the ECG signal is decomposed into 9 kinds of subsignals with different frequency scales by wavelet function, and then wavelet reconstruction is carried out after segmented filtering to eliminate the influence of noise.

Technology used: Tensor flow - Tensor flow is the core open source library to help you develop and train MachineLearning models.

### **Paper 2:**

1. ECG classification using deep CNN improved by wavelet transform
2. **Publication Year:** 2022-06-30
3. **Author:** . Zhao Y, Cheng, J . It is challenging to visually detect heart disease from the electrocardiographic (ECG) signals. Implementing an automated ECG signal detection system can help diagnosis arrhythmia in order to improve the accuracy of diagnosis. In this paper, we proposed, implemented, and compared an automated system using two different frameworks of the combination of convolutional neural network (CNN) and long-short term memory (LSTM) for classifying normal sinus signals, atrial fibrillation, and other noisy signals. The dataset we used is from the MIT-BIT Arrhythmia Physionet. Our approach demonstrated that the cascade of two deep learning network has higher performance than the concatenation of them, achieving a weighted f1 score of 0.82. The experimental results have successfully validated that the cascade of CNN and LSTM can achieve satisfactory performance on discriminating ECG signals Image. However, even after applying all the said techniques Might not possible to achieve the full accuracy in a Preprocessing system.

### **PROBLEM STATEMENT**

To identify the erythema by the user in the device and to convert images into prediction.

<b>What does this problem focus on?</b>	The generative models can perform recognition driven segmentation. The method involves a relatively small number of parameter and hence training is relatively easy and fast.
---	---

<b>When does this occur?</b>	This matter occurs classified the irregular heartbeat from the given input data personal individual .
<b>Why do we need this?</b>	To recognize the irregular heartbeat that can be used to classify the images.
<b>How to do this?</b>	Unlike many other recognition schemes, it does not rely on some form of prenormalization of input images, but can handle arbitrary scalings, translations and a limited degree of image rotation.
<b>Where it is used?</b>	It is used in personal tracker and monitor ECG.

### Ideation Phase

### Brainstorm & Idea Prioritization Template

Date	19 September 2022
Team ID	PNT2022TMID09656
Project Name	Classification of arrhythmia by using deep learning with 2-d ecg spectral image representation
Maximum Marks	4 Marks

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



## Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

- 10 minutes to prepare
- 1 hour to collaborate
- 2-8 people recommended

Share template feedback



### Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes



#### Team gathering

Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.



#### Set the goal

Think about the problem you'll be focusing on solving in the brainstorming session.



#### Learn how to use the facilitation tools

Use the Facilitation Superpowers to run a happy and productive session.

Open article



### Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes



Handwriting recognition is a challenging task because every person in this world has their own style of writing. It is the capability of the computer to automatically identify and understand the handwritten digits. Due to the technological advancements, everything is being digitalized to reduce human effort. Hence, handwritten digit recognition is a need-of-the-hour task in many real-time applications. MNIST data set, which has 70000 handwritten digit samples, is widely used for this recognition process.



### Key rules of brainstorming

To run a smooth and productive session

- Stay in topic.
- Encourage wild ideas.
- Defer judgment.
- Listen to others.
- Go for volume.
- If possible, be visual.

2

## Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil (switch to sketch) icon to start drawing!

### Rahul C

The proposed CNN model works on 2-D images of ECG signals as input data.	ECG signals, capturing, storing data manually could degrade the performance.	It should not have any limited amount of time or data.
The early detection of arrhythmias provides understanding of discomfort.	The early diagnosis of cardiac arrhythmias highly relies on the ECG.	Consume less time to finish the test and give the result.
It can also categorize according to risk.	Implemented in Python with the <b>IMEC</b> library.	The present research uses only a single-lead ECG signal.

### Vignesh P

An irregular or abnormal heartbeat.	Pause in sinus tracing.	Evaluating the tracing.
Abnormalities of regular generation or abnormalities of regular conduction or both.	Abnormalities of cardiac electrical activity result.	Based on Heart rate.
noninvasive diagnostic technique.	ECG data is the features based on the engineered.	Use mapping techniques for early detection or classification techniques using a deep neural network.

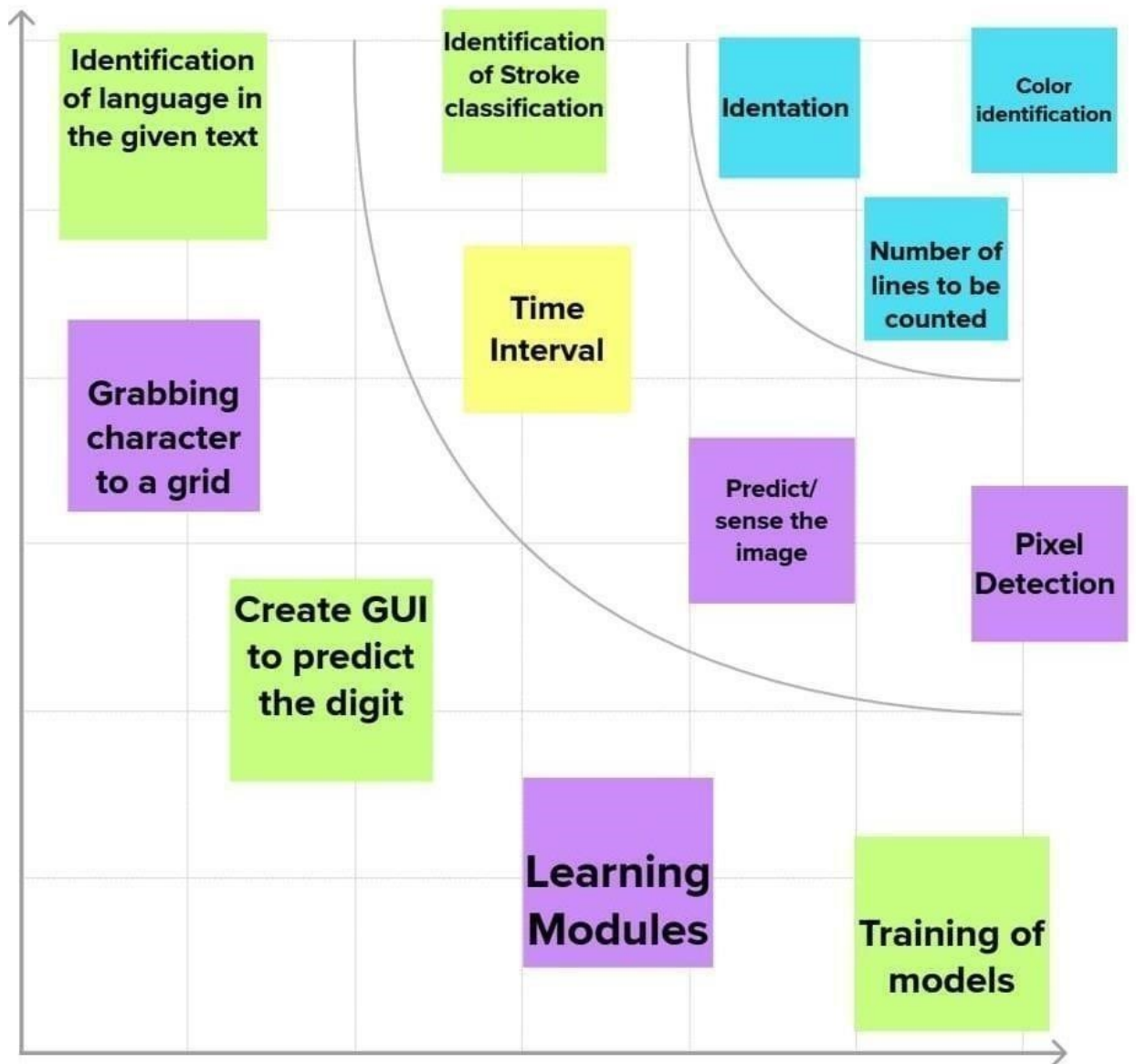
### Vijayesh G

Incorporating different algorithms of machine learning (ML) techniques.	Analysis of ECGs based on Artificial Neural network.	Detection of Obstructive Sleep Apnoea (OSA).
Detection of Intracranial Hemorrhage using Shallow CNN.	Multi-Lead ECG Classification by an Information-based Attention.	compared with and.
Each convolutional layer is followed by a pooling layer.	The model takes the ECG with features with time x 4 convolutional layers.	A fully connected layer is used between the last pooling layer and the output layer.

### SURAVARAJU SAI CHARAN REDDY

optimization parameters in the proposed 2-D CNN model.	Long term monitoring.	It is <b>ECG</b> only and gives the lead.
Detects irregular heart beats.	can ECG detect heart blockage.	can be easily added to modified.
delivering more preventive care.	poor electrode to patient contact.	Remote across and.

## Step-3: Idea Prioritization





# Step 4

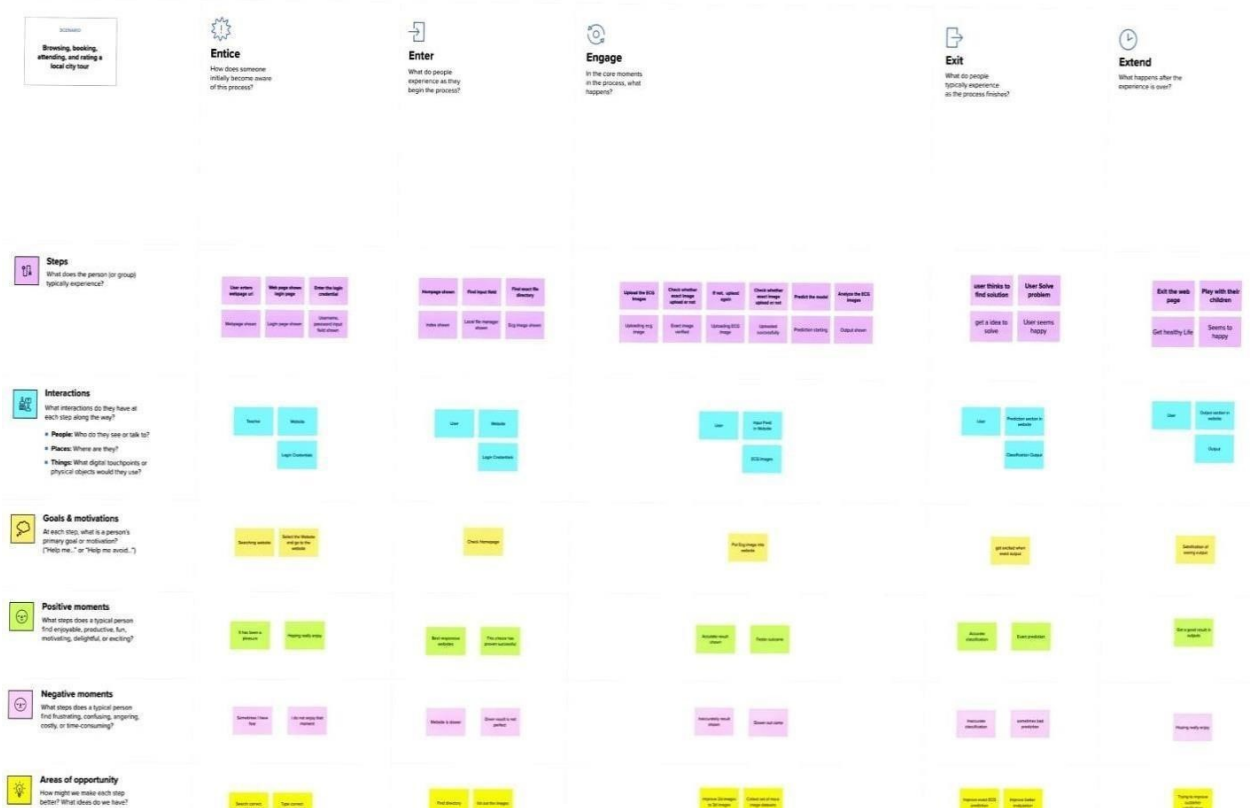
## Customer journey map

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

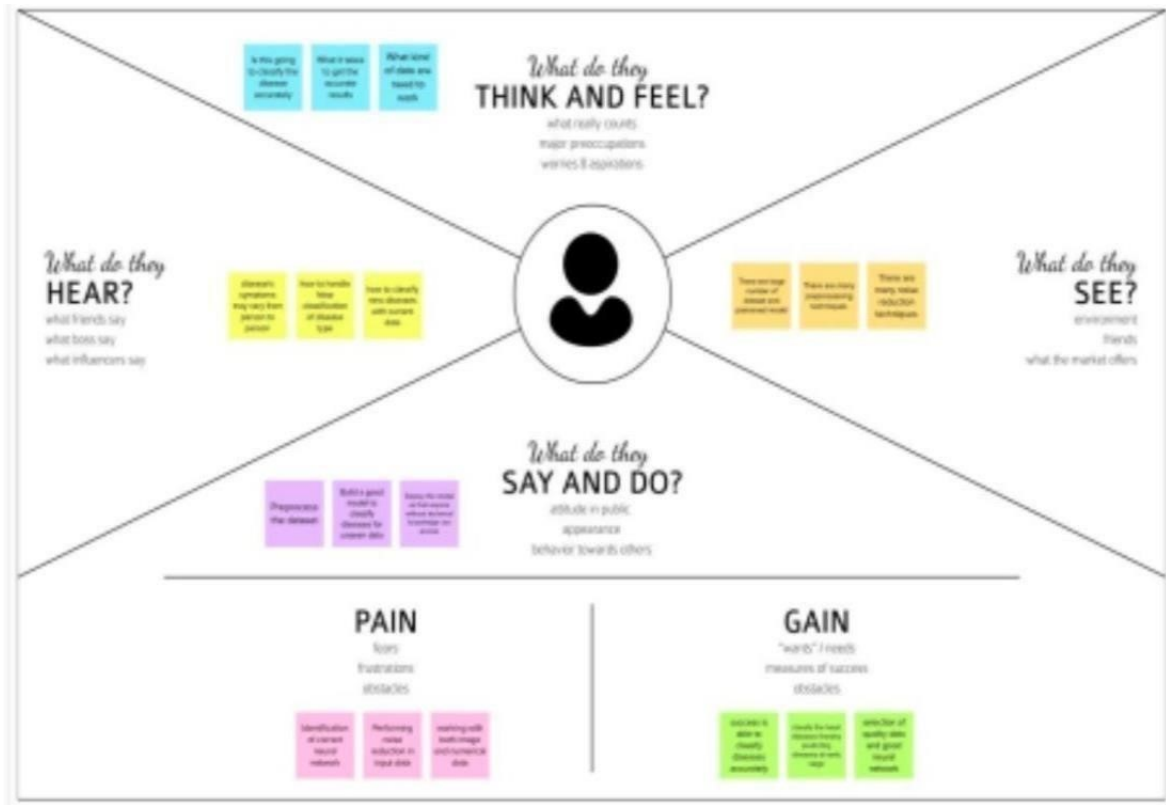
#### Document an existing experience

Narrow your focus to a specific scenario or process within an existing product or service. In the **Steps** row, document the step-by-step process someone typically experiences, then add detail to each of the other rows.

**TIP**  
As you will steps to the experience, map each step "True" for the left or right depending on the scenario you are documenting.



## EMPATHY MAP



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

## Project Design Phase-I Problem – Solution Fit Template

**Problem-Solution fit canvas 2.0** Purpose / Vision

Define CS, fit into CC	<b>1. CUSTOMER SEGMENT(S)</b> <span>CS</span> Who is your customer? i.e. working parents of 0-5 y.o. kids  <b>Many people and nations suffer from irregular heartbeats.</b>	<b>6. CUSTOMER CONSTRAINTS</b> <span>CC</span> What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices.  <b>In certain unique and unusual circumstances, there is a chance that findings won't be entirely accurate because of overfitting and underfitting. Additional investigation could be needed.</b>	<b>5. AVAILABLE SOLUTIONS</b> <span>AS</span> Which solutions are available to the customers when they face the problem or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? i.e. pen and paper is an alternative to digital notetaking.  <b>The most popular and affordable diagnostic technique for examining cardiac electrical impulses in medical facilities is the electrocardiogram (ECG). Arrhythmia, as it is widely called, refers to the aberrant cardiac signals. Cardiac arrhythmia has the potential to be fatal or at least hazardous. Different forms of arrhythmia can occur, and an ECG test can identify them. The automatic categorization of arrhythmias based on ECG beats has been established for years. The automated technologies that may be used as a screening tool for arrhythmia categorization are crucial for patients as well as for clinicians. Although the deep learning-based automated arrhythmia classification algorithms have shown very accurate findings, they have not yet been widely used by healthcare practitioners.</b>	Explore AS, differentiate
	<b>2. JOBS-TO-BE-DONE / PROBLEMS</b> <span>J&amp;P</span> Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides.  <b>The class of irregular heartbeat is discovered in the patient's heartbeat utilising the analysis and classification of the patients' CT scan image's provided input.</b>	<b>9. PROBLEM ROOT CAUSE</b> <span>RC</span> What is the real reason that this problem exists? What is the back story behind the need to do this job? i.e. customers have to do it because of the change in regulations.  <b>Electrolyte imbalances, such as when they are too high or too low, can disrupt the heart's signals and cause irregular heartbeats. certain medications and supplements. Arrhythmias can be brought on by several prescription medicines as well as some over-the-counter cough and cold remedies, too much booze</b>	<b>7. BEHAVIOUR</b> <span>BE</span> What does your customer do to address the problem and get the job done? i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)  <b>Based on the heart rate, bradyarrhythmias and tachyarrhythmias are the two major categories for arrhythmia. They are further categorised based on the source, mode of transmission, and related disorders.</b>	
Identify strong TR & EM	<b>3. TRIGGERS</b> <span>TR</span> What triggers customers to act? i.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news. <b>observing how other medical institutions quickly recognise their patients' problems and administer the necessary care</b>	<b>10. YOUR SOLUTION</b> <span>SL</span> If you are working on an existing business, write down your current solution first. Fill in the canvas, and check how much it fits reality. If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour.  <b>the patients' gathered photos Through the use of image processing and deep learning, CT scan pictures are analysed and categorised, making it simple to determine the kind of heartbeat.</b>	<b>8. CHANNELS of BEHAVIOUR</b> <span>CH</span> <b>8.1 ONLINE</b> What kind of actions do customers take online? Extract online channels from #7 <b>The patient attempts to research the condition they have online.</b>	Extract online & offline CH of BE
	<b>4. EMOTIONS: BEFORE / AFTER</b> <span>EM</span> How do customers feel when they face a problem or a job and afterwards? i.e. lost, insecure > confident, in control - use it in your communication strategy & design. <b>main emotions including happiness, sorrow, anger, fear, disgust, and surprise, as well as secondary emotions that conjure up an image in the mind that corresponds to a memory or primary emotion.</b>	<b>8.2 OFFLINE</b> What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.  <b>a patient makes an offline appointment to see a doctor</b>		

Problem-Solution fit canvas is licensed under a Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 license Created by Daria Neprikashina / Amaltama.com

**AMALTAMA**

## Project Design Phase-II Solution Requirements (Functional & Non-functional)

Date	14 October 2022
Team ID	PNT2022TMID09656
Project Name	Classification of arrhythmia by using deep learning with 2-d ECG spectral image representation

### Functional Requirements:

Following are the functional requirements of the proposed solution.

<b>FR No.</b>	<b>Functional Requirement (Epic)</b>	<b>Sub Requirement (Story / Sub-Task)</b>
FR-1	User Registration	Registration through Form Registration Registration through Gmail Registration through LinkedIN
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR- 3	User interface	Check your profile Choose your file Sign Out your account account and change your password
FR- 4	Data processing	Evaluating the model using test data Training DL algorithm for a accuracy result Trained CNN model using Tensorflow,Kearas
FR-5	Predict ECG image	User ECG images in our web application Collection of datasets Database read ECG images

### **Non-functional Requirements:**

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

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	<p>Wireless ECG body sensor Savvy is a feasible solution for reliable and accurate long-term heart rhythm monitoring.</p> <p>However, there were no studies dealing with usability of this sensor in field testing.</p>
NFR-2	Security	The work presented in this paper is applicable for encrypting and decrypting personalized Electrocardiograph ECG signals for secure transmission.
NFR-3	Reliability	The extent to the consistently performs the specified functions without failure
NFR-4	Performance	It essentially specifies how the system should behave and that it constrains the ECG wavelength of accurate disease information gathering.
NFR-5	Availability	Availability describes how likely the system is accessible to a user at a given point in time and the periodically for a solutions.
NFR-6	Scalability	The ability of the user problem in arrhythmia disease to handle an increase in workload without performance degradation, or its ability to quickly enlarge.

**Project Design Phase-II**  
**(Data Flow Diagram & User Stories)**

Date	17 October 2022
Team ID	PNT2022TMID09656
Project Name	A Novel Method for Handwritten Digit Recognition System
Maximum Marks	4 Marks

**Data Flow Diagrams:**

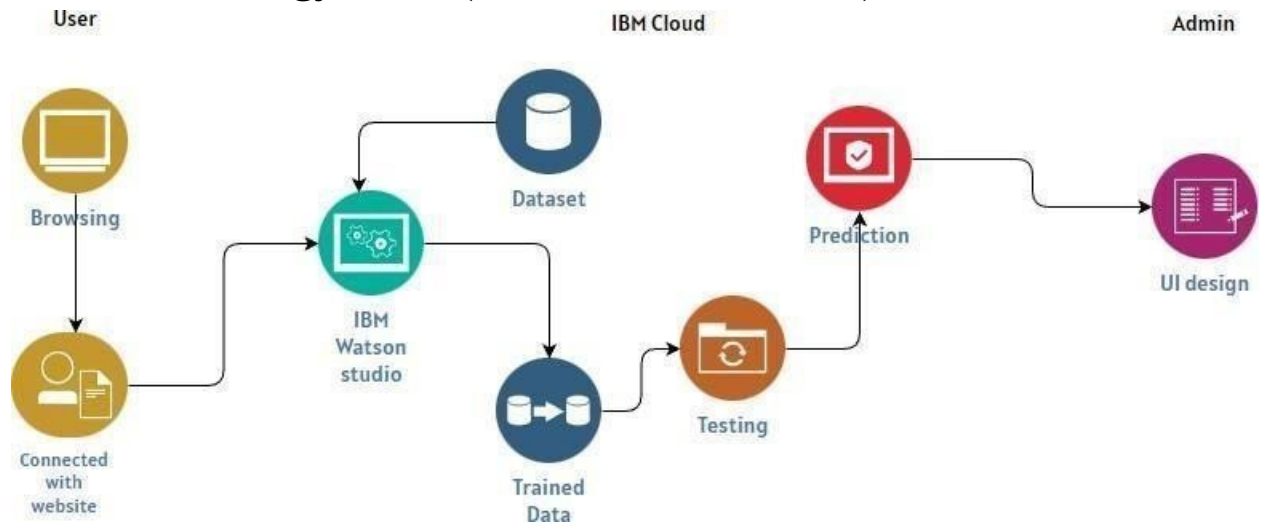
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

User Type	User Story Number	User Story / Task	Release
Customer	USN-1	As a user, I can application by opening it easily.	Sprint-1

	USN-2	As a user, I can upload images	Sprint-1
	USN-3	As a user, I can change the colour of the pen ink.	Sprint-2

## Project Design Phase-II

### Technology Stack (Architecture & Stack)



Date	14 October 2022
Team ID	PNT2022TMID09656
Project Name	Classification of Arrhythmia by Using Deep Learning with 2-D ECG Spectral Image Representation

#### Technical Architecture:

The Deliverable shall include the architectural diagram as below and the information as per the table1 & table 2 **Table-1 : Components & Technologies:**

S. No	Component	Description	Technology
1	User Interface	Web UI, Mobile UI.	HTML, CSS, JavaScript / React



			Js.
2 .	Application Logic-1	Python is used for backend	Python
3 .	Application Logic-2	It's a symbolic math toolkit that performs a variety of tasks including deep neural network training and inference using dataflow and differentiable programming	Tensorflow
4 .	Cloud Database	A global technology company that provides hardware, software, cloud-based services and cognitive computing.	IBM Cloud
5 .	File Storage	Breaks up data into blocks and then stores those blocks as separate pieces, each with a unique identifier.	IBM Block
6 .	External API-1	Purpose of External API used in the application	IBM Weather API, etc.
7 .	External API-2	Purpose of External API used in the application	Aadhar API, etc.
8 .	Machine Learning Model	Object recognition is a subfield of computer vision, artificial intelligence, and machine learning	Object Recognition Model

9 .	Deep learning Model	The images from the created dataset are fed into a neural network algorithm.	Image Recognition Model
--------	---------------------	--	-------------------------


Table-2: Application Characteristics:

S. No	Characteristics	Description	Technology
1 .	Open-Source Frameworks	Building user interfaces based on UI components.	React Js
2 .	Security Implementations	OWASP is a nonprofit foundation that works to improve the security of software.	OWASP
3 .	Scalable Architecture	a modular client-server architecture that consists of a presentation tier, an application tier and a data tier	3-tier architecture
4 .	Availability	The data on each server can be simultaneously accessed and modified via a network.	Distributed Server

5	Performance	Increasing data retrieval performance by reducing the need to access the underlying slower storage layer.	Cache
---	-------------	---	-------

## Webpages

### Info.html

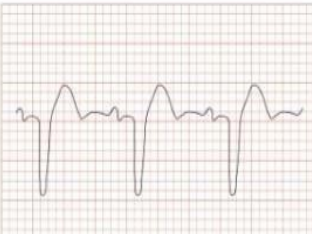

[Home](#)
[info](#)
[About Us](#)
[Predict](#)

## Left Bundle Branch

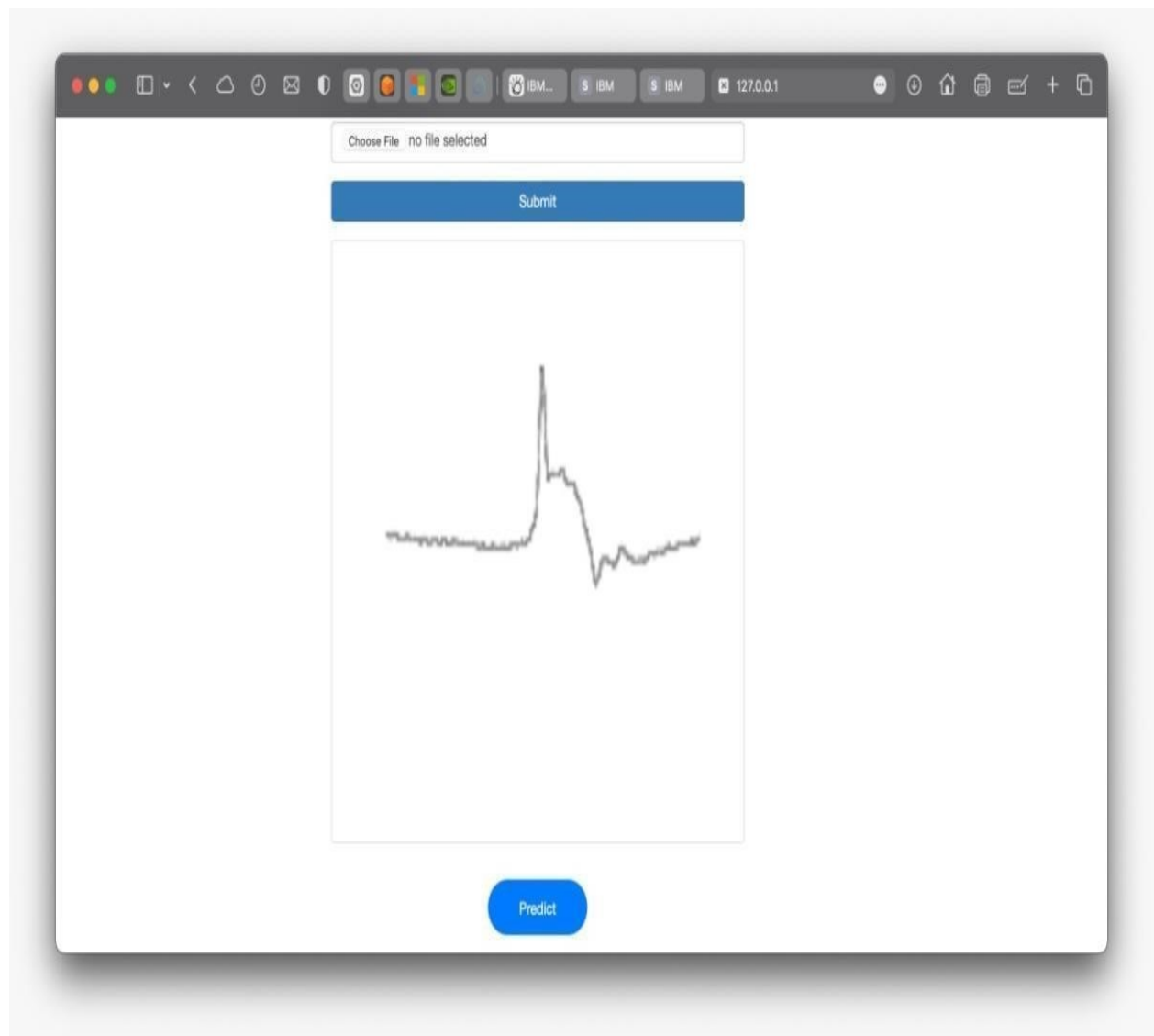
A delay blockage of electrical impulses to the left of the heart. Left bundle brach block sometimes makes it harder for the heart to pump blood efficiently through the circulatory system.

Most people don't have symptoms. If syntoms occur, they inlcude fainting or a slow heart rate.

Some people with the condition don't know they have bundle branch block. Rarely, symptoms of bundle branch block may include fainting (syncope) or feeling as if you're going to faint (presyncope).



## Predict.html

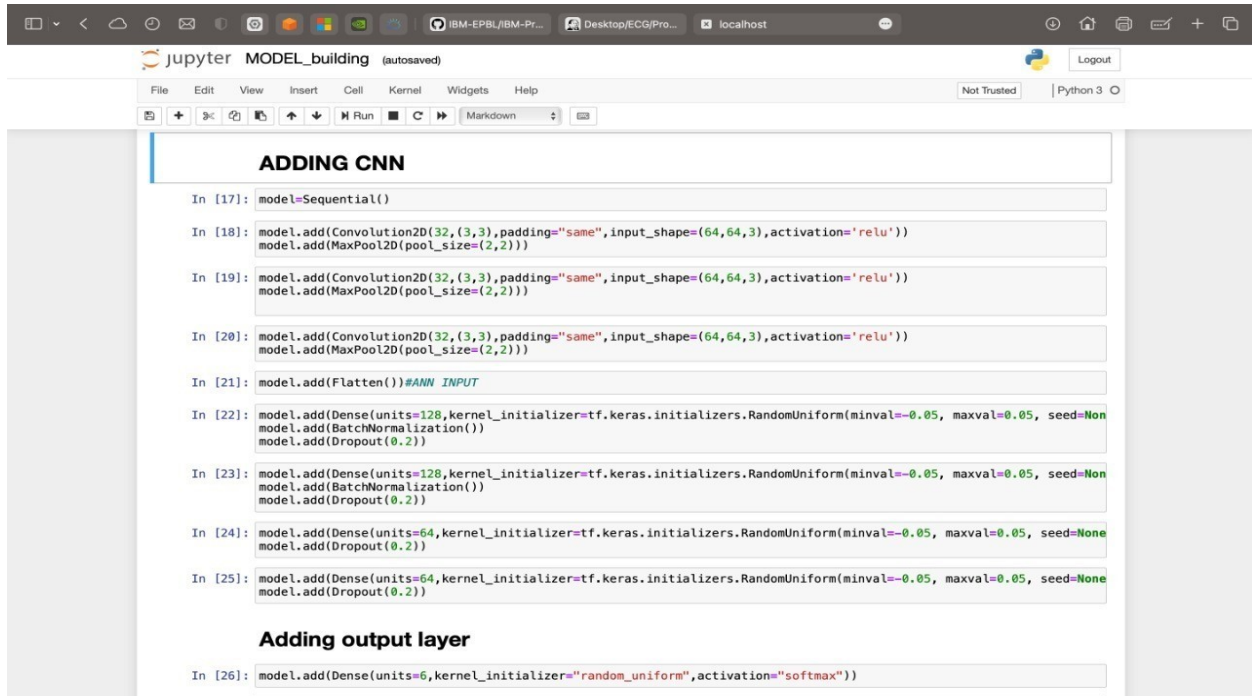


## Run the application

```
1 import os
2 import numpy as np # used for numerical analysis
3 from flask import Flask, request, render_template
4
5 from tensorflow.keras.models import load_model
6 from tensorflow.keras.preprocessing import image
7
8 app = Flask(__name__)
9 model = load_model('ECG.h5')
10
11 @app.route("/") #default route
12 @app.route("/home") #Home page set to default page
13 def default():
14     return render_template('index.html') #rendering index.html
15
16 @app.route("/info") #route to info page
17 def information():
18     return render_template("info.html") #rendering info.html
19
20 @app.route("/about") #route to about us
```

127.0.0.1 -- [18/Nov/2022 01:02:06] "GET /static/images/LBB.svg HTTP/1.1" 304 -  
127.0.0.1 -- [18/Nov/2022 01:02:06] "GET /static/images/banner\_img.jpg HTTP/1.1" 304 -  
127.0.0.1 -- [18/Nov/2022 01:02:06] "GET /static/images/normal.svg HTTP/1.1" 304 -  
127.0.0.1 -- [18/Nov/2022 01:02:06] "GET /static/images/RBD.svg HTTP/1.1" 304 -  
127.0.0.1 -- [18/Nov/2022 01:02:06] "GET /static/images/VF.png HTTP/1.1" 304 -  
127.0.0.1 -- [18/Nov/2022 01:02:06] "GET /static/images/PAC.jpg HTTP/1.1" 304 -  
127.0.0.1 -- [18/Nov/2022 01:02:44] "GET /home HTTP/1.1" 200 -  
127.0.0.1 -- [18/Nov/2022 01:02:44] "GET /static/images/logo.png HTTP/1.1" 404 -  
127.0.0.1 -- [18/Nov/2022 01:02:44] "GET /static/images/next.png HTTP/1.1" 404 -  
127.0.0.1 -- [18/Nov/2022 01:02:44] "GET /static/images/app.png HTTP/1.1" 304 -  
127.0.0.1 -- [18/Nov/2022 01:02:44] "GET /static/css/style.css HTTP/1.1" 304 -  
127.0.0.1 -- [18/Nov/2022 01:02:44] "GET /static/images/banner\_img.jpg HTTP/1.1" 304 -  
127.0.0.1 -- [18/Nov/2022 01:02:44] "GET /static/images/web.jpg HTTP/1.1" 304 -  
127.0.0.1 -- [18/Nov/2022 01:02:44] "GET /static/images/consultPatient.png HTTP/1.1" 304 -  
127.0.0.1 -- [18/Nov/2022 01:02:44] "GET /static/images/banner\_1.svg HTTP/1.1" 404 -  
127.0.0.1 -- [18/Nov/2022 01:02:44] "GET /static/images/logo.png HTTP/1.1" 404 -  
127.0.0.1 -- [18/Nov/2022 01:02:44] "GET /static/images/next.png HTTP/1.1" 404 -  
127.0.0.1 -- [18/Nov/2022 01:03:03] "GET /about HTTP/1.1" 200 -  
127.0.0.1 -- [18/Nov/2022 01:03:03] "GET /static/static/css/about.css HTTP/1.1" 404 -  
127.0.0.1 -- [18/Nov/2022 01:03:03] "GET /static/static/css/style.css HTTP/1.1" 404 -  
127.0.0.1 -- [18/Nov/2022 01:03:03] "GET /static/images/logo.jpg HTTP/1.1" 304 -  
127.0.0.1 -- [18/Nov/2022 01:03:03] "GET /sprints/static/images/profile\_avatar.png HTTP/1.1" 404 -  
127.0.0.1 -- [18/Nov/2022 01:03:03] "GET /static/images/about\_us.png HTTP/1.1" 200 -  
127.0.0.1 -- [18/Nov/2022 01:03:03] "GET /static/images/profile\_avatar.png HTTP/1.1" 200 -  
127.0.0.1 -- [18/Nov/2022 01:03:09] "GET /upload HTTP/1.1" 200 -  
127.0.0.1 -- [18/Nov/2022 01:03:09] "GET /static/images/fig\_12.png HTTP/1.1" 304 -  
127.0.0.1 -- [18/Nov/2022 01:03:09] "GET /static/js/jquery.min.js HTTP/1.1" 304 -  
127.0.0.1 -- [18/Nov/2022 01:03:09] "GET /static/css/bootstrap.min.css HTTP/1.1" 304 -

# MODEL BUILDING



The image shows a Jupyter Notebook titled "MODEL\_building" with a toolbar at the top. The notebook contains a series of code cells for building a CNN model. The code starts with creating a Sequential model and adding three convolutional layers, each followed by a max pooling layer. The input shape for the first layer is (64, 64, 3). The model is then flattened and passed through two dense layers, each with 128 units, followed by a dropout layer. The final output layer is a dense layer with 6 units and a softmax activation. The notebook interface includes a "File" menu, "Edit", "View", "Insert", "Cell", "Kernel", "Widgets", and "Help" options. The status bar at the bottom indicates "Not Trusted" and "Python 3".

```
In [17]: model=Sequential()

In [18]: model.add(Convolution2D(32,(3,3),padding="same",input_shape=(64,64,3),activation='relu'))
model.add(MaxPool2D(pool_size=(2,2)))

In [19]: model.add(Convolution2D(32,(3,3),padding="same",input_shape=(64,64,3),activation='relu'))
model.add(MaxPool2D(pool_size=(2,2)))

In [20]: model.add(Convolution2D(32,(3,3),padding="same",input_shape=(64,64,3),activation='relu'))
model.add(MaxPool2D(pool_size=(2,2)))

In [21]: model.add(Flatten())#ANN INPUT

In [22]: model.add(Dense(units=128,kernel_initializer=tf.keras.initializers.RandomUniform(minval=-0.05, maxval=0.05, seed=None)))
model.add(BatchNormalization())
model.add(Dropout(0.2))

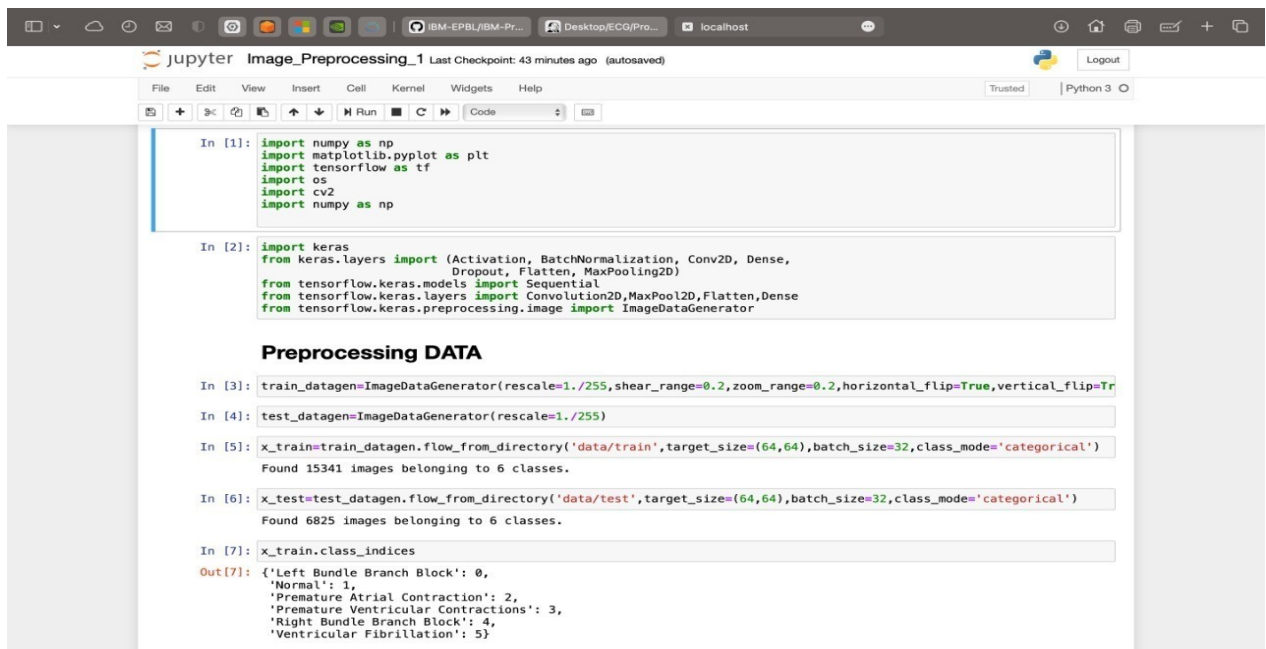
In [23]: model.add(Dense(units=128,kernel_initializer=tf.keras.initializers.RandomUniform(minval=-0.05, maxval=0.05, seed=None)))
model.add(BatchNormalization())
model.add(Dropout(0.2))

In [24]: model.add(Dense(units=64,kernel_initializer=tf.keras.initializers.RandomUniform(minval=-0.05, maxval=0.05, seed=None)))
model.add(Dropout(0.2))

In [25]: model.add(Dense(units=64,kernel_initializer=tf.keras.initializers.RandomUniform(minval=-0.05, maxval=0.05, seed=None)))
model.add(Dropout(0.2))

Adding output layer

In [26]: model.add(Dense(units=6,kernel_initializer="random_uniform",activation="softmax"))
```



The image shows a Jupyter Notebook titled "Image\_Preprocessing\_1" with a toolbar at the top. The notebook contains a series of code cells for data preprocessing. The code starts with importing necessary libraries: numpy, matplotlib.pyplot, tensorflow, os, cv2, and keras. It then defines the training and testing data generators using ImageDataGenerator. The training data is loaded from the 'data/train' directory, and the testing data is loaded from the 'data/test' directory. The code also shows the class indices for the training data, which are: 'Left Bundle Branch Block': 0, 'Normal': 1, 'Premature Atrial Contraction': 2, 'Premature Ventricular Contractions': 3, 'Right Bundle Branch Block': 4, and 'Ventricular Fibrillation': 5. The notebook interface includes a "File" menu, "Edit", "View", "Insert", "Cell", "Kernel", "Widgets", and "Help" options. The status bar at the bottom indicates "Trusted" and "Python 3".

```
In [1]: import numpy as np
import matplotlib.pyplot as plt
import tensorflow as tf
import os
import cv2
import numpy as np

In [2]: import keras
from keras.layers import (Activation, BatchNormalization, Conv2D, Dense,
Dropout, Flatten, MaxPooling2D)
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Convolution2D,MaxPool2D,Flatten,Dense
from tensorflow.keras.preprocessing.image import ImageDataGenerator

Preprocessing DATA

In [3]: train_datagen=ImageDataGenerator(rescale=1./255, shear_range=0.2, zoom_range=0.2, horizontal_flip=True, vertical_flip=True)

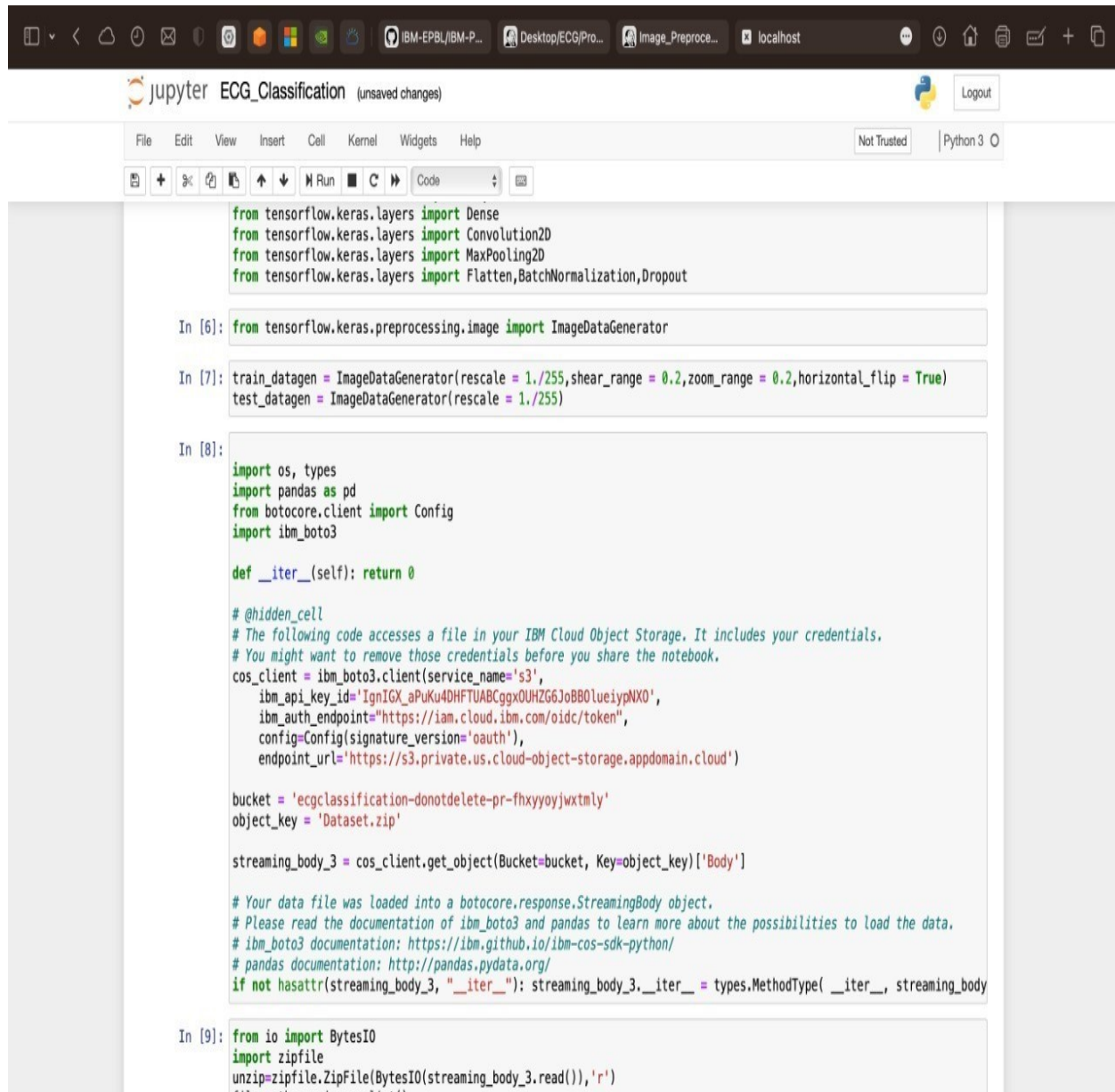
In [4]: test_datagen=ImageDataGenerator(rescale=1./255)

In [5]: x_train=train_datagen.flow_from_directory('data/train', target_size=(64,64), batch_size=32, class_mode='categorical')
Found 15341 images belonging to 6 classes.

In [6]: x_test=test_datagen.flow_from_directory('data/test', target_size=(64,64), batch_size=32, class_mode='categorical')
Found 6825 images belonging to 6 classes.

In [7]: x_train.class_indices
Out[7]: {'Left Bundle Branch Block': 0,
'Normal': 1,
'Premature Atrial Contraction': 2,
'Premature Ventricular Contractions': 3,
'Right Bundle Branch Block': 4,
'Ventricular Fibrillation': 5}
```

# ECG Classification



The image shows a JupyterLab interface with a dark theme. The top bar includes a file explorer on the left, a central toolbar with icons for file operations, and a right sidebar with a 'Logout' button. The main area displays a notebook titled 'ECG\_Classification' with several code cells. The first cell contains imports for TensorFlow Keras layers. The second cell imports ImageDataGenerator. The third cell creates train and test datagen objects. The fourth cell imports os, pandas, boto3, and ibm\_boto3, and defines a generator function. The fifth cell imports BytesIO and zipfile, and uses them to load data from a streaming body.

```
from tensorflow.keras.layers import Dense
from tensorflow.keras.layers import Convolution2D
from tensorflow.keras.layers import MaxPooling2D
from tensorflow.keras.layers import Flatten, BatchNormalization, Dropout

In [6]: from tensorflow.keras.preprocessing.image import ImageDataGenerator

In [7]: train_datagen = ImageDataGenerator(rescale = 1./255, shear_range = 0.2, zoom_range = 0.2, horizontal_flip = True)
test_datagen = ImageDataGenerator(rescale = 1./255)

In [8]:
import os, types
import pandas as pd
from boto3.client import Config
import ibm_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 = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='IgnIGX_aPuKu4DHFTUABCGgx0UHZG6JoBB0lueiypNX0',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

bucket = 'ecgclassification-donotdelete-pr-fhxyoyjwxtmly'
object_key = 'Dataset.zip'

streaming_body_3 = cos_client.get_object(Bucket=bucket, Key=object_key)['Body']

# Your data file was loaded into a boto3.response.StreamingBody object.
# Please read the documentation of ibm_boto3 and pandas to learn more about the possibilities to load the data.
# ibm_boto3 documentation: https://ibm.github.io/ibm-cos-sdk-python/
# pandas documentation: http://pandas.pydata.org/
if not hasattr(streaming_body_3, "__iter__"): streaming_body_3.__iter__ = types.MethodType(__iter__, streaming_body_3)

In [9]: from io import BytesIO
import zipfile
unzip=zipfile.ZipFile(BytesIO(streaming_body_3.read()), 'r')
```



# HTML CODING

```
about.html | contact.html | index.html | info.html | predict.html
Users > apple > Desktop > ECG > Project_development_phase > Sprint3 > Application_Building > templates > > info.html > ...
1  <!DOCTYPE html>
2  <html lang="en">
3
4  <head>
5    <meta charset="UTF-8" />
6    <meta http-equiv="X-UA-Compatible" content="IE=edge" />
7    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
8    <title>HEART Care - About Classification of Arrhythmia</title>
9
10   <link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/aos/2.3.1/aos.css" />
11   <link href="https://fonts.googleapis.com/css2?family=Playfair+Display:wght@600&display=swap" rel="stylesheet" />
12   <link rel="stylesheet" href="{{url_for('static', filename='css/style.css')}}" />
13   <script src="https://kit.fontawesome.com/64d58efce2.js" crossorigin="anonymous">
14   </script>
15   </head>
16   <body>
17     <div class="banner">
18       <div class="bannerText">
19         <h1>HEART Care</h1>
20         <p>About Classification of Arrhythmia</p>
21       </div>
22     </div>
23   </body>
24 </html>
```

```
about.html | contact.html | index.html | info.html | predict.html
Users > apple > Desktop > ECG > Project_development_phase > Sprint3 > Application_Building > templates > > about.html > ...
13 </div>
14 </div>
15 </div>
16 <div class="footer">
17   <div class="footerText">
18     <p>HEART Care</p>
19   </div>
20   <div class="links">
21     <a href="/">Home</a>
22     <a href="/info">Info</a>
23     <a href="/about" class="mainLink">About Us</a>
24     <a href="/contact">Contact Us</a>
25     <a href="/upload" class="btn1">Predict</a>
26   </div>
27 </div>
28 <div class="landing">
29   <div class="landingText">
30     <h1>HEART Care</h1>
31     <h2>About Classification of Arrhythmia</h2>
32     <p>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.</p>
33   </div>
34   <div class="landingImage">
35     
36   </div>
37 </div>
38 </div>
39 </div>
40 </div>
41 </div>
42 </div>
43 </div>
44 </div>
45 </div>
46 </div>
47 </div>
48 </div>
49 </div>
50 </div>
51 </div>
52 </div>
53 </div>
54 </div>
55 </div>
56 </div>
57 </div>
```

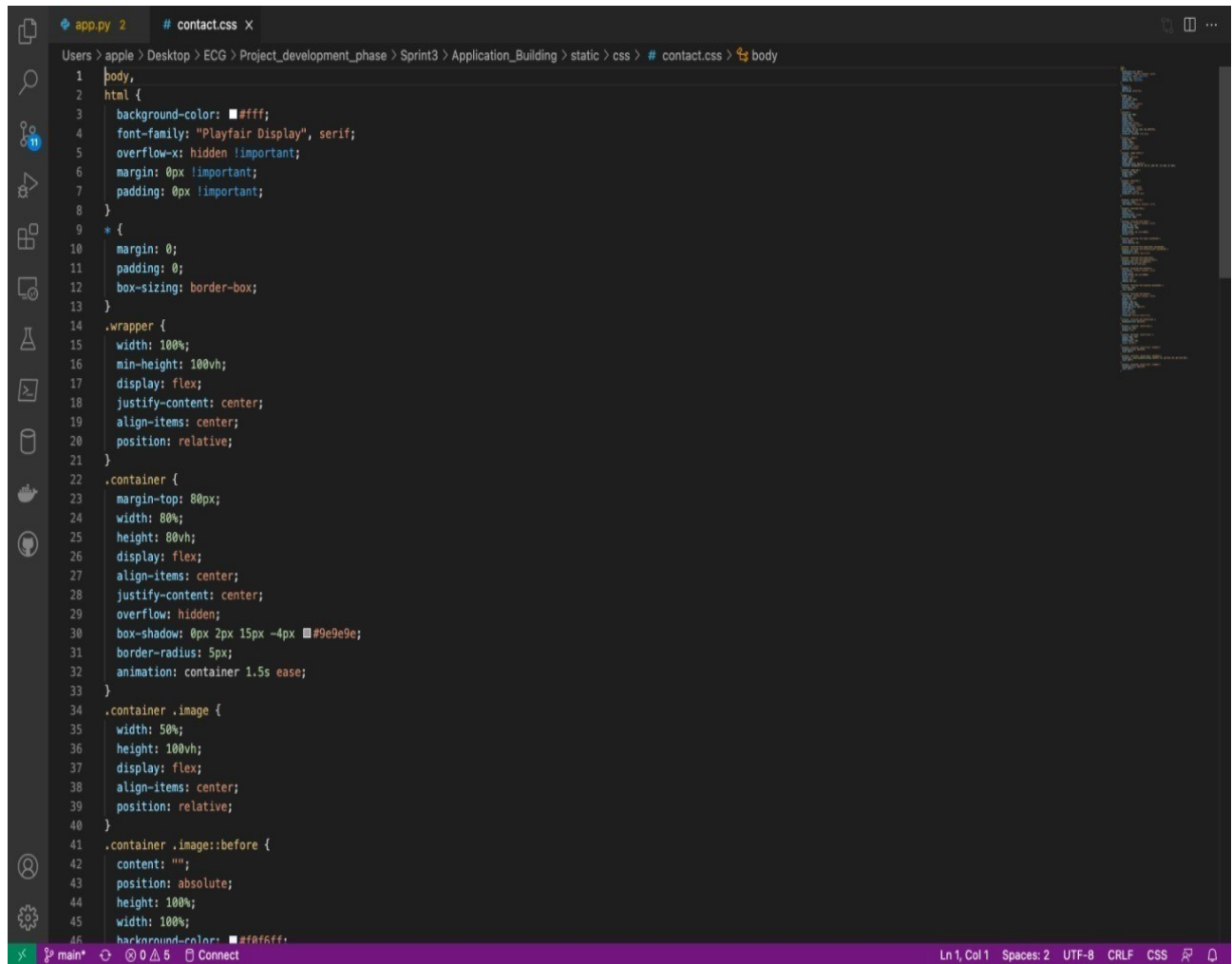


# APP.PY

```
app.py 2 X
Users > apple > Desktop > ECG > Project_development_phase > Sprint3 > Application_Building > app.py > ...

1 import os
2 import numpy as np # used for numerical analysis
3 from flask import Flask, request, render_template
4
5 from tensorflow.keras.models import load_model
6 from tensorflow.keras.preprocessing import image
7
8 app = Flask(__name__)
9 model = load_model('ECG.h5')
10 target_img = os.path.join(os.getcwd(), 'static/testing')
11
12
13 @app.route("/") #default route
14 @app.route("/home") #Home page set to default page
15 def default():
16     return render_template('index.html') #rendering index.html
17
18 @app.route("/info") #route to info page
19 def information():
20     return render_template("info.html") #rendering info.html
21
22 @app.route("/about") #route to about us page
23 def about_us():
24     return render_template('about.html') #rendering about.html
25
26 @app.route("/contact") #route to contact us page
27 def contact_us():
28     return render_template('contact.html') #rendering contact.html
29
30 #Allow files with extension png, jpg and jpeg
31 @app.route("/upload") #default route
32 def test():
33     return render_template("predict.html")
34 app.config['UPLOAD_FOLDER']='static/testing'
35 @app.route("/", methods=['GET','POST'])
36 def upload():
37     if request.method=='POST':
38         upload_image=request.files['upload_image']
39
40         if upload_image.filename!='':
41             filepath=os.path.join(app.config["UPLOAD_FOLDER"],upload_image.filename)
42             upload_image.save(filepath)
43             path=filepath
44
45             #flash("File Upload Successfully","success")
46     return render_template("predict.html",data=path)
```

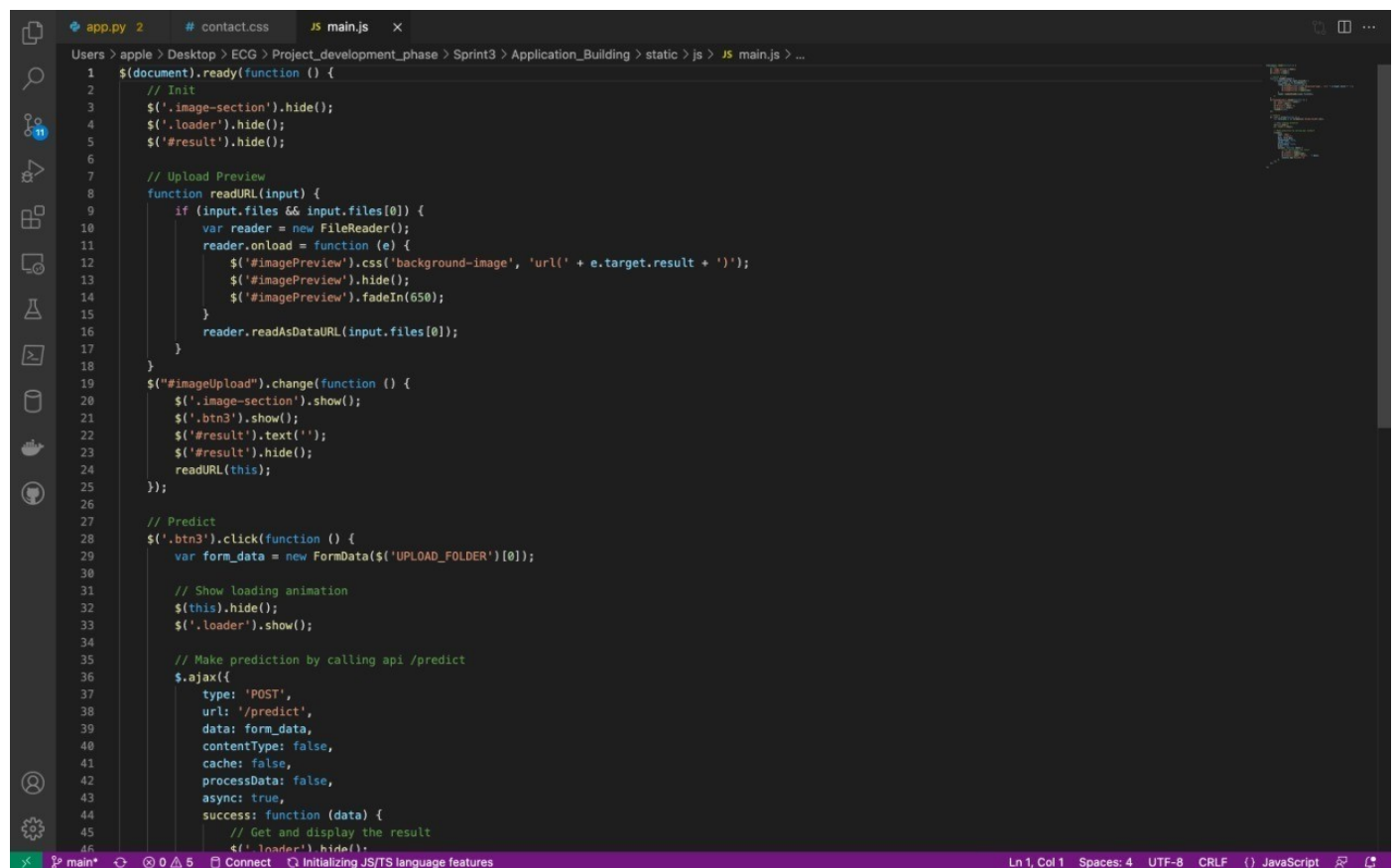
# CSS



```
1 body,
2 html {
3   background-color: #fff;
4   font-family: "Playfair Display", serif;
5   overflow-x: hidden !important;
6   margin: 0px !important;
7   padding: 0px !important;
8 }
9 * {
10  margin: 0;
11  padding: 0;
12  box-sizing: border-box;
13 }
14 .wrapper {
15  width: 100%;
16  min-height: 100vh;
17  display: flex;
18  justify-content: center;
19  align-items: center;
20  position: relative;
21 }
22 .container {
23  margin-top: 80px;
24  width: 80%;
25  height: 80vh;
26  display: flex;
27  align-items: center;
28  justify-content: center;
29  overflow: hidden;
30  box-shadow: 0px 2px 15px -4px #9e9e9e;
31  border-radius: 5px;
32  animation: container 1.5s ease;
33 }
34 .container .image {
35  width: 50%;
36  height: 100vh;
37  display: flex;
38  align-items: center;
39  position: relative;
40 }
41 .container .image::before {
42  content: "";
43  position: absolute;
44  height: 100%;
45  width: 100%;
46  background-color: #f0f5ff;
```

Ln 1, Col 1 Spaces: 2 UTF-8 CRLF CSS Connect

## JAVA SCRIPT



```
1 $(document).ready(function () {
2     // Init
3     $('#image-section').hide();
4     $('#loader').hide();
5     $('#result').hide();
6
7     // Upload Preview
8     function readURL(input) {
9         if (input.files && input.files[0]) {
10             var reader = new FileReader();
11             reader.onload = function (e) {
12                 $('#imagePreview').css('background-image', 'url(' + e.target.result + ')');
13                 $('#imagePreview').hide();
14                 $('#imagePreview').fadeIn(650);
15             }
16             reader.readAsDataURL(input.files[0]);
17         }
18     }
19     $("#imageUpload").change(function () {
20         $('#image-section').show();
21         $('#btn3').show();
22         $('#result').text('');
23         $('#result').hide();
24         readURL(this);
25     });
26
27     // Predict
28     $('#btn3').click(function () {
29         var form_data = new FormData($('#UPLOAD_FOLDER')[0]);
30
31         // Show loading animation
32         $(this).hide();
33         $('#loader').show();
34
35         // Make prediction by calling api /predict
36         $.ajax({
37             type: 'POST',
38             url: '/predict',
39             data: form_data,
40             contentType: false,
41             cache: false,
42             processData: false,
43             async: true,
44             success: function (data) {
45                 // Get and display the result
46                 $('#loader').hide();
```

app.py 2 xcontact.cssJS main.js

Users > apple > Desktop > ECG > Project\_development\_phase > Sprint3 > Application\_Building > app.py > ...

```
1 import os
2 import numpy as np # used for numerical analysis
3 from flask import Flask, request, render_template
4
5 from tensorflow.keras.models import load_model
6 from tensorflow.keras.preprocessing import image
7
8 app = Flask(__name__)
9 model = load_model('ECG.h5')
10 target_img = os.path.join(os.getcwd(), 'static/testing')
11
12
13 @app.route("/") #default route
14 @app.route("/home") #Home page set to default page
15 def default():
16     return render_template('index.html') #rendering index.html
17
18 @app.route("/info") #route to info page
19 def information():
20     return render_template("info.html") #rendering info.html
21
22 @app.route("/about") #route to about us page
23 def about_us():
24     return render_template('about.html') #rendering about.html
25
26 @app.route("/contact") #route to contact us page
27 def contact_us():
28     return render_template('contact.html') #rendering contact.html
29
30 #Allow files with extension png, jpg and jpeg
31 @app.route("/upload") #default route
32 def test():
33     return render_template("predict.html")
34 app.config['UPLOAD_FOLDER'] = "static/testing"
35 app.run(host='0.0.0.0', port=5000)
```

PROBLEMS 0 OUTPUT TERMINAL SQL CONSOLE DEBUG CONSOLE

python3.9 - Application\_Building

127.0.0.1 - [18/Nov/2022 11:35:06] "GET /static/images/banner\_img.jpg HTTP/1.1" 200 -
127.0.0.1 - [18/Nov/2022 11:35:06] "GET /static/images/web.jpg HTTP/1.1" 200 -
127.0.0.1 - [18/Nov/2022 11:35:06] "GET /static/images/banner\_1.svg HTTP/1.1" 404 -
127.0.0.1 - [18/Nov/2022 11:35:07] "GET /static/images/logo.png HTTP/1.1" 404 -
127.0.0.1 - [18/Nov/2022 11:35:07] "GET /static/images/next.png HTTP/1.1" 404 -
127.0.0.1 - [18/Nov/2022 11:35:07] "GET /static/images/banner\_1.svg HTTP/1.1" 404 -
127.0.0.1 - [18/Nov/2022 11:35:08] "GET /upload HTTP/1.1" 200 -
127.0.0.1 - [18/Nov/2022 11:35:08] "GET /static/images/fig\_12.png HTTP/1.1" 200 -
127.0.0.1 - [18/Nov/2022 11:35:08] "GET /static/js/jquery.min.js HTTP/1.1" 200 -
127.0.0.1 - [18/Nov/2022 11:35:08] "GET /static/css/bootstrap.min.css HTTP/1.1" 200 -

Ln 1, Col 1 Spaces: 4 UTF-8 LF Python 3.11.0 64-bit