Project Report

DETECTION OF PARKINSON’S DISEASE USING MACHINE LEARNING

Submitted By

PNT2022TMID19812

DIVYA SINGH	-	727819TUIT017
JAYA LAKSHMI	-	727819TUIT031
KAVIYA VIKASHINI	-	727819TUIT040
KAVIARASU	-	727819TUIT038
BALACHANDRAN	-	727819TUIT004

TABLE OF CONTENTS

CHAPTER – 1

INTRODUCTION

1.1.PROJECT OVERVIEW:

In the present decade of accelerated advances in Medical Sciences, most studies fail to lay focus on ageing diseases. These are diseases that display their symptoms at a much advanced stage and makes a complete recovery almost improbable. Parkinson’s disease (PD) is the second most commonly diagnosed neurodegenerative disorder of the brain. One could argue, that it is almost incurable and inflicts a lot of pain on the patients. All these make it quite clear that there is an oncoming need for efficient, dependable and expandable diagnosis of Parkinson’s disease. A dilemma of this intensity requires the automating of the diagnosis to lead accurate and reliable results. It has been observed that most PD Patients demonstrate some sort of impairment in speech or speech dysphonia, which makes speech measurements and indicators one of the most important aspects in prediction of PD. The aim of this work is to compare various machine learning models in the successful prediction of the severity of Parkinson’s disease and develop an effective and accurate model in order to help diagnose the disease accurately at an earlier stage which could in turn help the doctors to assist in the cure and recovery of PD Patients. For the aforementioned purpose we plan on using the Parkinson’s Tele monitoring dataset which was acquired from the UCIML repository.

1.2.PURPOSE:

The aim of this work is to compare various machine learning models in the successful prediction of the severity of Parkinson's disease and develop an effective and accurate model in order to help diagnose the disease accurately at an earlier stage which could in turn help the doctors to assist in the cure and recovery . This project showed 90% efficiency. In our model, a huge amount of data is collected from the normal person and also previously affected person by Parkinson's disease.

CHAPTER – 2

LITERATURE SURVEY

2.1.EXISTING PROBLEM:

In existing system, PD is detected at the secondary stage only (Dopamine deficiency) which leads to medical challenges. Also doctor has to manually examine and suggest medical diagnosis in which the symptoms might vary from person to person so suggesting medicine is also a challenge. Thus the mental disorders are been poorly characterized and have many health complications. PD is generally diagnosed with the following clinical methods as,

- MRI or CT scan - Conventional MRI cannot detect early signs of Parkinson's disease
- PET scan - is used to assess activity and function of brain regions involved in movement
- SPECT scan - can reveal changes in brain chemistry, such as a decrease in dopamine .

This results in a high misdiagnosis rate (up to 25% by non-specialists) and many years before diagnosis, people can have the disease. Thus existing system is not effective in early prediction and accurate medicinal diagnosis to the affected people.

2.2.REFERENCES:

[1] Adrien Payan, Giovanni Montana, Predicting Alzheimer's disease: a neuroimaging study with 3D convolutional neural networks.

[2] Alemami, Y. and Almazaydeh, L. (2014) Detecting of Parkinson Disease through Voice Signal Features. Journal of American Science.

[3] Fayao Liu, Chunhua Shen, Learning Deep Convolutional Features for MRI Based Alzheimer’s Disease Classification.

[4] Hadjahamadi, A.H. and Askari, T.J. (2012) A Detection Support System for Parkinson’s Disease Diagnosis Using Classification and Regression Tree. Journal of Mathematics and Computer Science , 4, 257-263.

[5] Little, M.A., McSharry, P.E., Hunter, E.J. and Ramig, L.O. (2008), Suitability of Dysphonia Measurements for Telemonitoring of Parkinson's disease. IEEE Transactions on Biomed ical Engineering, 56, 1015-1022.

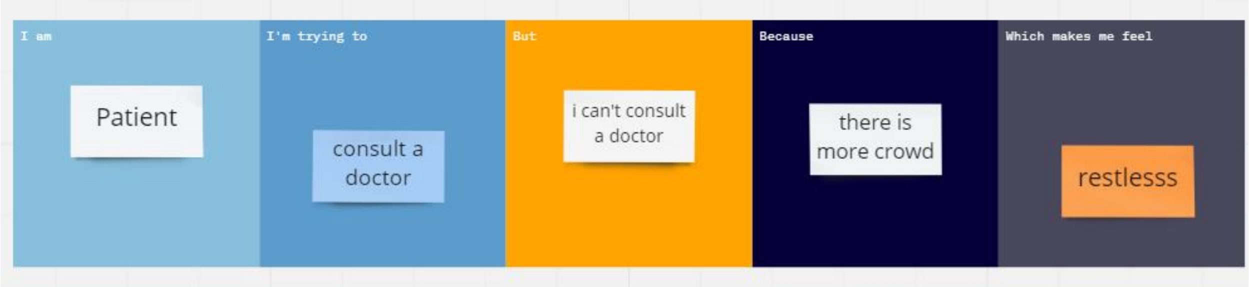
[6] Muhlenbach, F. and Rakotomalala, R. (2015) Discretization of Continuous Attributes. In: Wang, J., Ed., Encyclopedia of Data Warehousing and Mining, Idea Group Reference, 397-402.

[7] Olanrewaju, R.F., Sahari, N.S., Musa, A.A. and Hakiem, N. (2014) Application of Neural Networks in Early Detection and Diagnosis of Parkinson's Disease. International Conference on Cyber and IT Service Management.

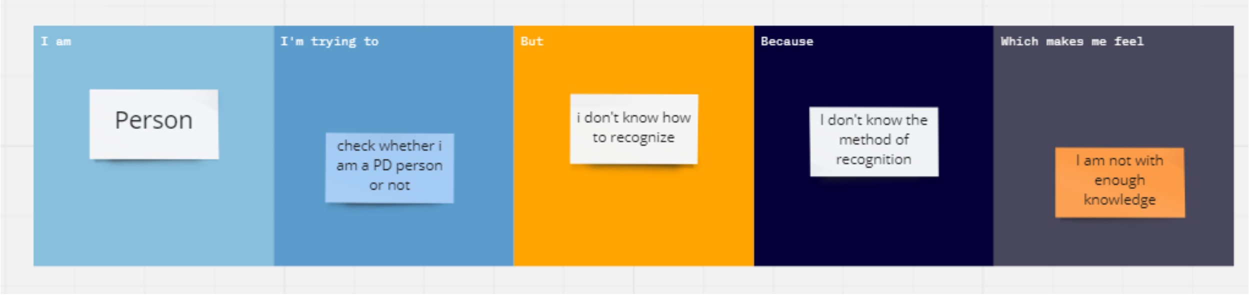
[8] Saman Sarraf, Danielle D. DeSouza, John Anderson, Ghassem Tofighi, DeepAD: Alzheimer′s Disease Classification via Deep Convolutional Neural Networks using MRI and fMRI, Cold Spring Harbor Laboratory Press.

2.3.PROBLEM STATEMENT DEFINITION:

Problem Statement (PS)	**I am (Customer)**	**I'm trying to**	**But**	**Because**	**Which makes me feel**
PS-1	Patient	Consult a doctor	I can't consult a doctor	There is more crowd.	Restless
PS-2	Person	Check whether I am a PD patient or not.	I don't know how to recognize	I don't know the method of recognition	I am not with enough knowledge.



I am	I'm trying to	But	Because	Which makes me feel
Patient	consult a doctor	i can't consult a doctor	there is more crowd	restlesss

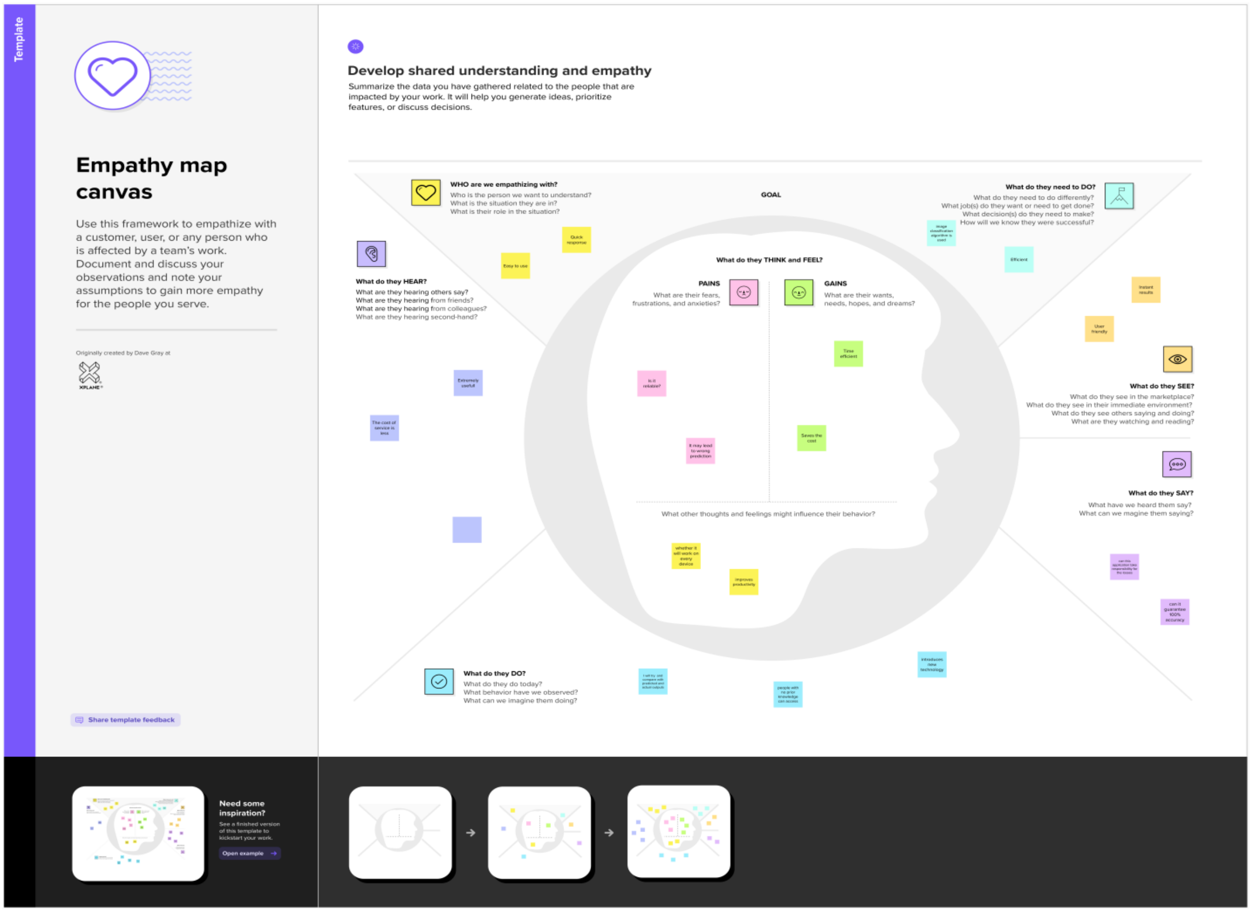


I am	I'm trying to	But	Because	Which makes me feel
Person	check whether i am a PD person or not	i don't know how to recognize	I don't know the method of recognition	I am not with enough knowledge

CHAPTER – 3

IDEATION & PROPOSED SOLUTION

3.1.EMPATHY MAP CANVAS:



Empathy map canvas

Use this framework to empathize with a customer, user, or any person who is affected by a team's work. Document and discuss your observations and note your assumptions to gain more empathy for the people you serve.

Originally created by Dave Gray at

Develop shared understanding and empathy

Summarize the data you have gathered related to the people that are impacted by your work. It will help you generate ideas, prioritize features, or discuss decisions.

WHO are we empathizing with?
Who is the person we want to understand?
What is the situation they are in?
What is their role in the situation?

GOAL

What do they need to DO?
What do they need to do differently?
What job(s) do they want or need to get done?
What decision(s) do they need to make?
How will we know they were successful?

What do they HEAR?
What are they hearing others say?
What are they hearing from friends?
What are they hearing from colleagues?
What are they hearing second-hand?

What do they THINK and FEEL?

PAINS
What are their fears, frustrations, and anxieties?

GAINS
What are their wants, needs, hopes, and dreams?

What other thoughts and feelings might influence their behavior?

What do they SEE?
What do they see in the marketplace?
What do they see in their immediate environment?
What do they see others saying and doing?
What are they watching and reading?

What do they SAY?
What have we heard them say?
What can we magine them saying?

What do they DO?
What do they do today?
What behavior have we observed?
What can we imagine them doing?

Quick response

Easy to use

Extremely usefull

The cost of service is less

image classification algorithm is used

Efficient

Instant results

User friendly

Is it reliable?

It may lead to wrong prediction

Time efficient

Saves the cost

whether it will work on every device

improves productivity

can this application take responsibility for the losses

can it guarantee 100% accuracy

I will try and compare with predicted and actual outputs

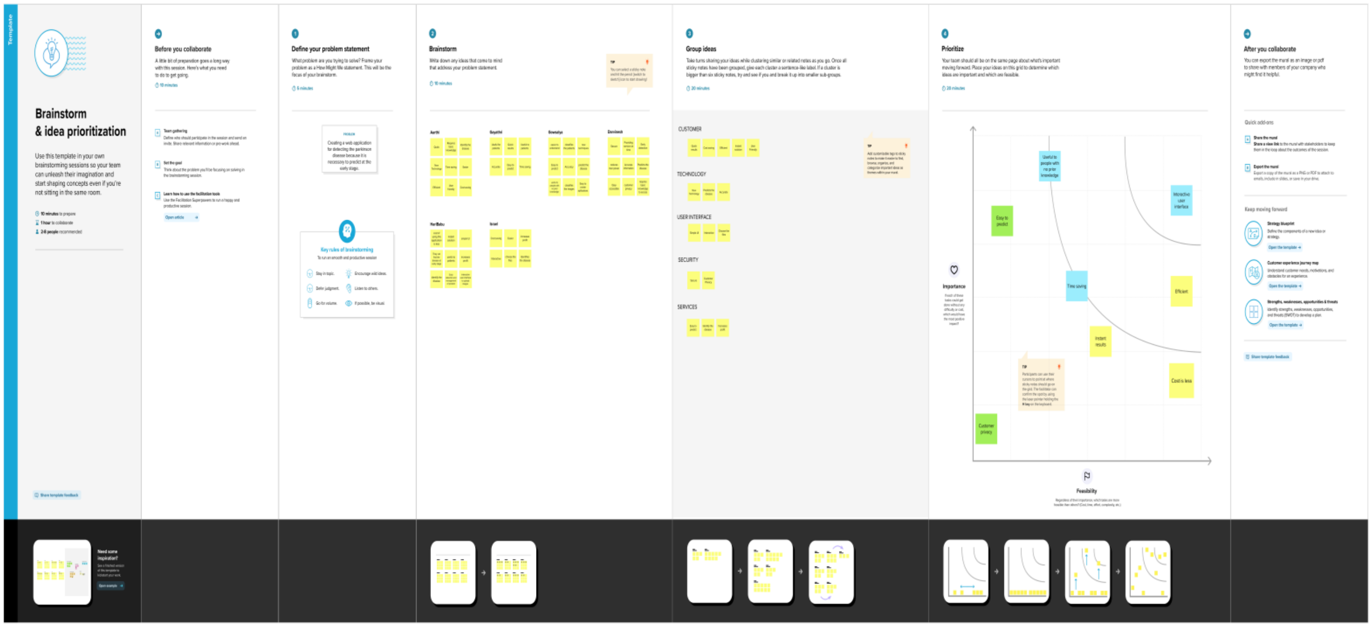
people with no prior knowledge can access

introduces new technology

Need some inspiration?

See a finished version of this template to kickstart your work.

3.2.IDEATION AND BRAINSTORMING:



Template

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

Need some inspiration?

Open example

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

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

B. **Set the goal** Think about the problem you'll be focusing on solving in the brainstorming session.

C. **Learn how to use the facilitation tools** Use the Facilitation Superpowers to run a happy and productive session.

Open article →

1 **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

PROBLEM

Creating a web application for detecting the parkinson disease because it is necessary to predict at the early stage.

Key rules of brainstorming

To run an 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!

3 **Group ideas**

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

CUSTOMER

TECHNOLOGY

USER INTERFACE

SECURITY

SERVICES

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

4 **Prioritize**

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the H key on the keyboard.

After you collaborate

You can export the mural as an image or pdf to share with members of your company who might find it helpful.

Quick add-ons

A. **Share the mural** Share a view link to the mural with stakeholders to keep them in the loop about the outcomes of the session.

B. **Export the mural** Export a copy of the mural as a PNG or PDF to attach to emails, include in slides, or save in your drive.

Keep moving forward

Strategy blueprint Define the components of a new idea or strategy.

Open the template →

Customer experience journey map Understand customer needs, motivations, and obstacles for an experience.

Open the template →

Strengths, weaknesses, opportunities & threats Identify strengths, weaknesses, opportunities, and threats (SWOT) to develop a plan.

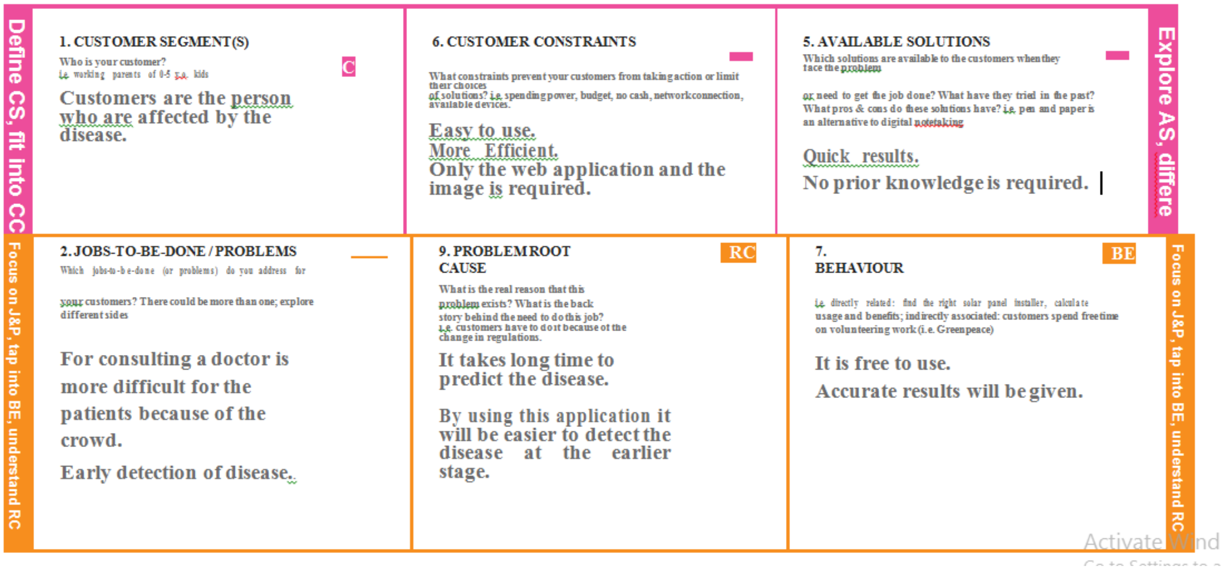
Open the template →

Share template feedback

3.3.PROPOSED SOLUTION:

S.NO	PARAMETER	DESCRIPTION
1	Problem Statement (Problem to be solved)	Creating a web application for the detection of Parkinson disease using machine learning to predict the disease at the earlier stage.
2	Idea / Solution description	They are recognized more faster and more accurate. The model trained to learn the low level to high level features and the classification results are validated.
3	Novelty / Uniqueness	Quick results. No prior knowledge is required.
4	Social Impact / Customer Satisfaction	It is free to use. Accurate results will be given. More secure.
5	Business Model (Revenue Model)	Less money is required. By using this application it will be easier to detect the disease at the earlier stage with no cost.
6	Scalability of the Solution	The performance of this application will be high. The cost of this application is decreased.

3.4.PROBLEM SOLUTION FIT:



Define CS, fit into CC
Explore AS, differe
Focus on J&P, tap into BE, understand RC
Focus on J&P, tap into BE, understand RC
C
RC
BE

1. CUSTOMER SEGMENT(S)

Who is your customer?
i.e. working parents of 0-5 y.o. kids

Customers are the person who are affected by the disease.

6. CUSTOMER CONSTRAINTS

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.

Easy to use.
More Efficient.
Only the web application and the image is required.

5. AVAILABLE SOLUTIONS

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

Quick results.
No prior knowledge is required.

2. JOBS-TO-BE-DONE / PROBLEMS

Which jobs-to-be-done (or problems) do you address for

your customers? There could be more than one; explore different sides

For consulting a doctor is more difficult for the patients because of the crowd.

Early detection of disease.

9. PROBLEM ROOT CAUSE

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.

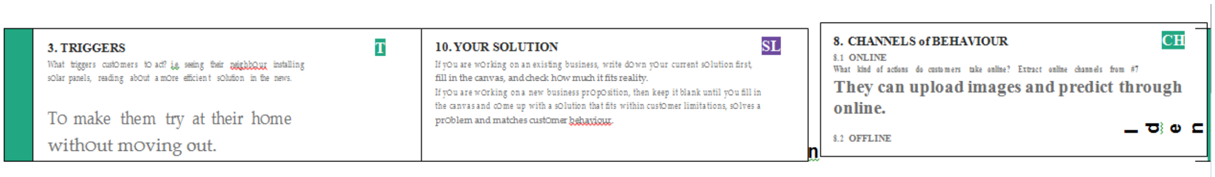
It takes long time to predict the disease.

By using this application it will be easier to detect the disease at the earlier stage.

7. BEHAVIOUR

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)

It is free to use.
Accurate results will be given.



3. TRIGGERS

What triggers customers to act? i.e. seeing their neighbour installing solar panels, reading about a more efficient solution in the news.

To make them try at their home without moving out.

10. YOUR SOLUTION

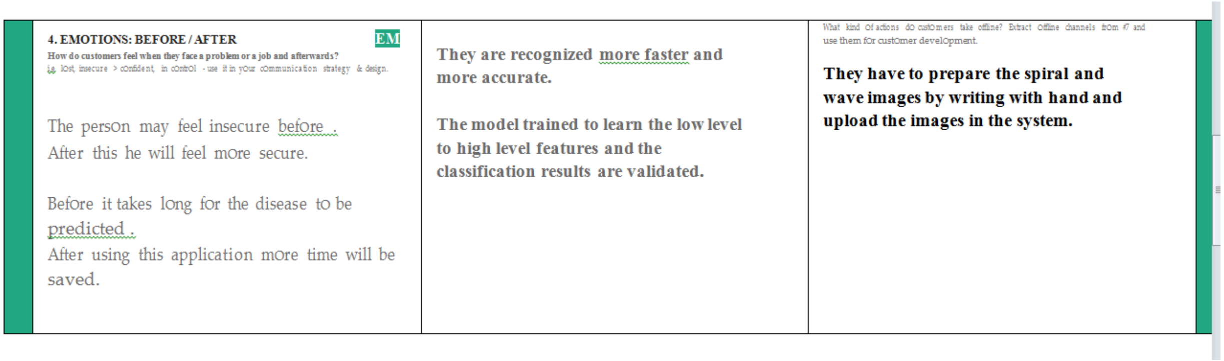
If you are working on an existing business, write down your current solution first, fill in the canvas, and check how much it fits reality.
If you are working on a new business proposition, then keep it blank until you fill in the canvas and come up with a solution that fits within customer limitations, solves a problem and matches customer behaviour.

8. CHANNELS of BEHAVIOUR

8.1 ONLINE
What kind of actions do customers take online? Extract online channels from #7

They can upload images and predict through online.

8.2 OFFLINE



4. EMOTIONS: BEFORE / AFTER

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.

The person may feel insecure before .
After this he will feel more secure.

Before it takes long for the disease to be predicted .
After using this application more time will be saved.

They are recognized more faster and more accurate.

The model trained to learn the low level to high level features and the classification results are validated.

What kind of actions do customers take offline? Extract offline channels from #7 and use them for customer development.

They have to prepare the spiral and wave images by writing with hand and upload the images in the system.

CHAPTER – 4

REQUIREMENT ANALYSIS

4.1.FUNCTIONAL REQUIREMENT:

FR NO:	FUNCTIONAL REQUIREMENT	SUB REQUIREMENT
FR-1	User Registration	Registration through Gmail
FR-2	User Confirmation	Confirmation via Email
FR-3	Uploading Dataset	Spiral and wave images are to be uploaded.
FR-4	Requesting Solution	Uploaded images are compared with the pre-defined Model and solution is generated.
FR-5	Downloading Solution	The output can be downloaded in the PDF format.

4.2.NON-FUNCTIONAL REQUIREMENT:

FR.NO:	NON-FUNCTIONAL REQUIREMENT	DESCRIPTION
NFR-1	Usability	The system allows the user to perform the tasks easily and efficiently.
NFR-2	Security	Assuring all data inside the system or its part will be protected against unauthorized access.
NFR-3	Reliability	The website takes time when

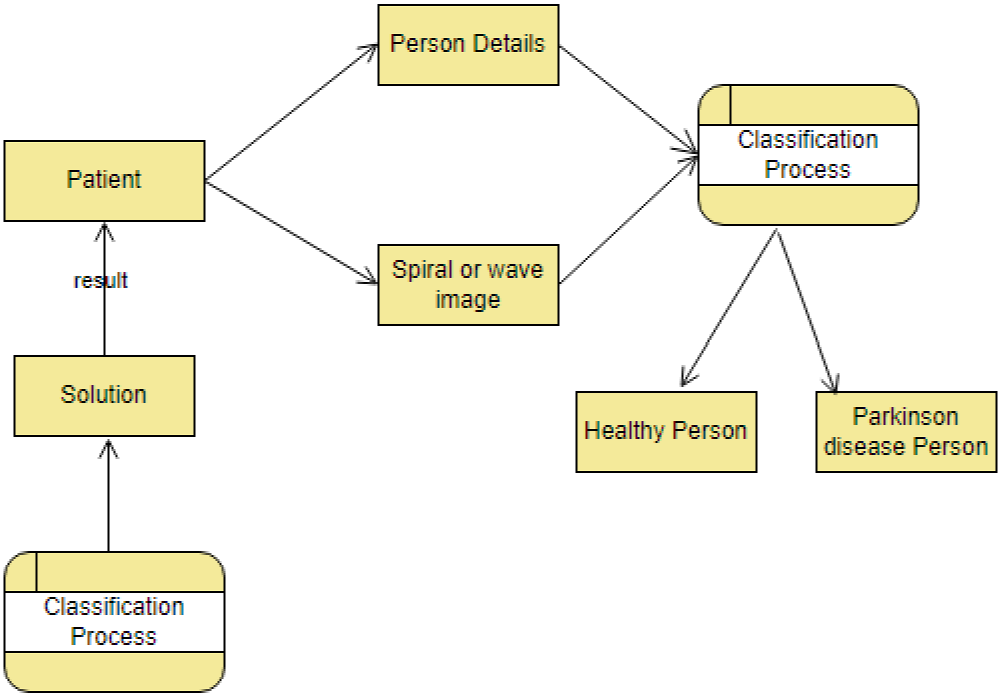
		the application runs in a single server.
NFR-4	Performance	Response time and the processing time is fast.
NFR-5	Availability	The system will be available 100% of the time.
NFR-6	Scalability	The website is scalable.

CHAPTER – 5

PROJECT DESIGN

5.1.DATA FLOW DIAGRAM:

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 AND TECHNICAL ARCHITECTURE:

Solution architecture is a complex process – with many sub-processes – that bridges the gap between business problems and technology solutions.

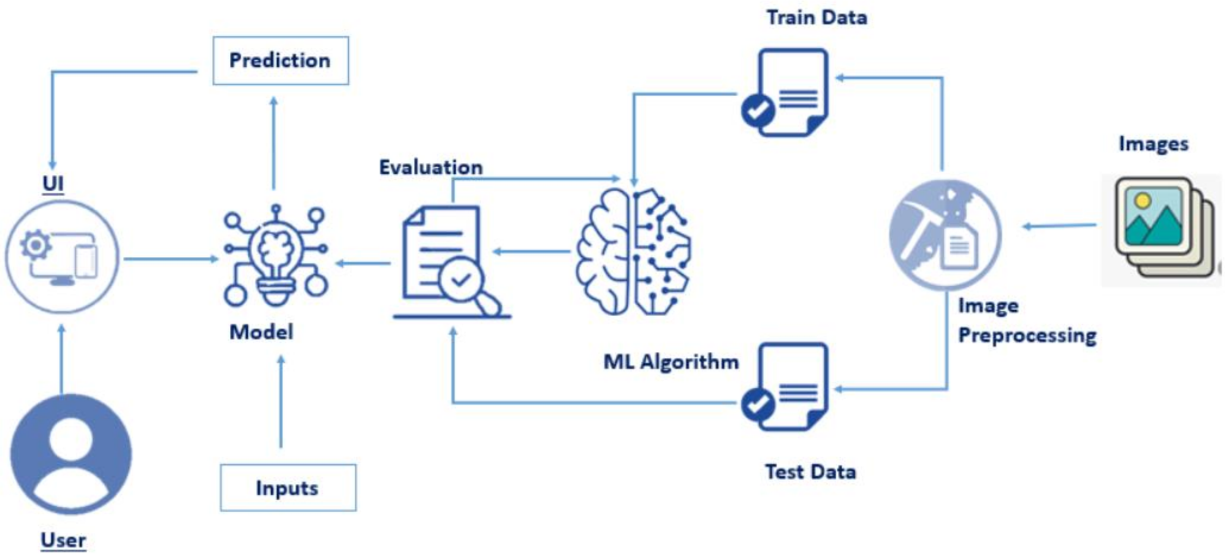
Its goals are to:

- Find the best tech solution to solve existing business problems.

- Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.

- Define features, development phases, and solution requirements.

- Provide specifications according to which the solution is defined, managed, and delivered.



5.3.USER STORIES:

USER TYPE	FUNCTIONAL REQUIREMENT	USER STORY NUMBER	USER STORY/TASK	ACCEPTANCE CRITERIA	PRIORITY	RELEAS
Customer(Mobile User)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	I can access my account /dashboard	High	Sprint-1
	Login	USN-2	As a user, I can log into the application by entering email & password	I can login using my E-mail ID accounts or user credentials.	High	Sprint-1
	Dashboard	USN-3	As a user I can view the page of the application where i can upload my images of spiral and wave.	I can access my account/Dashboard.	High	Sprint-2
Customer (Web user)	Registration	USN-4	As a user ,I can login to web dashboard just like website dashboard.	I can register using my username and password.	High	Sprint-3
	Login	USN-5	As a user, I can login to my website dashboard with the login credentials.	I Can login using my user credentials.	High	Sprint-3
	Dashboard	USN-6	As a user , I can view the web application where I can upload my images.	I can access my accounts/Dashboard.	High	Sprint-4
Administrator	Login	USN-7	As a admin , I can login to the website using	I can login to the website using my login credentials.	High	Sprint-1

			my login credentials.			
	Dashboard	USN-8	As a admin, I can view the dashboard of the application.	I can access my dashboard.	High	Sprint-2

CHAPTER – 6

PROJECT PLANNING AND SCHEDULING

6.1.SPRINT DELIVERY SCHEDULE:

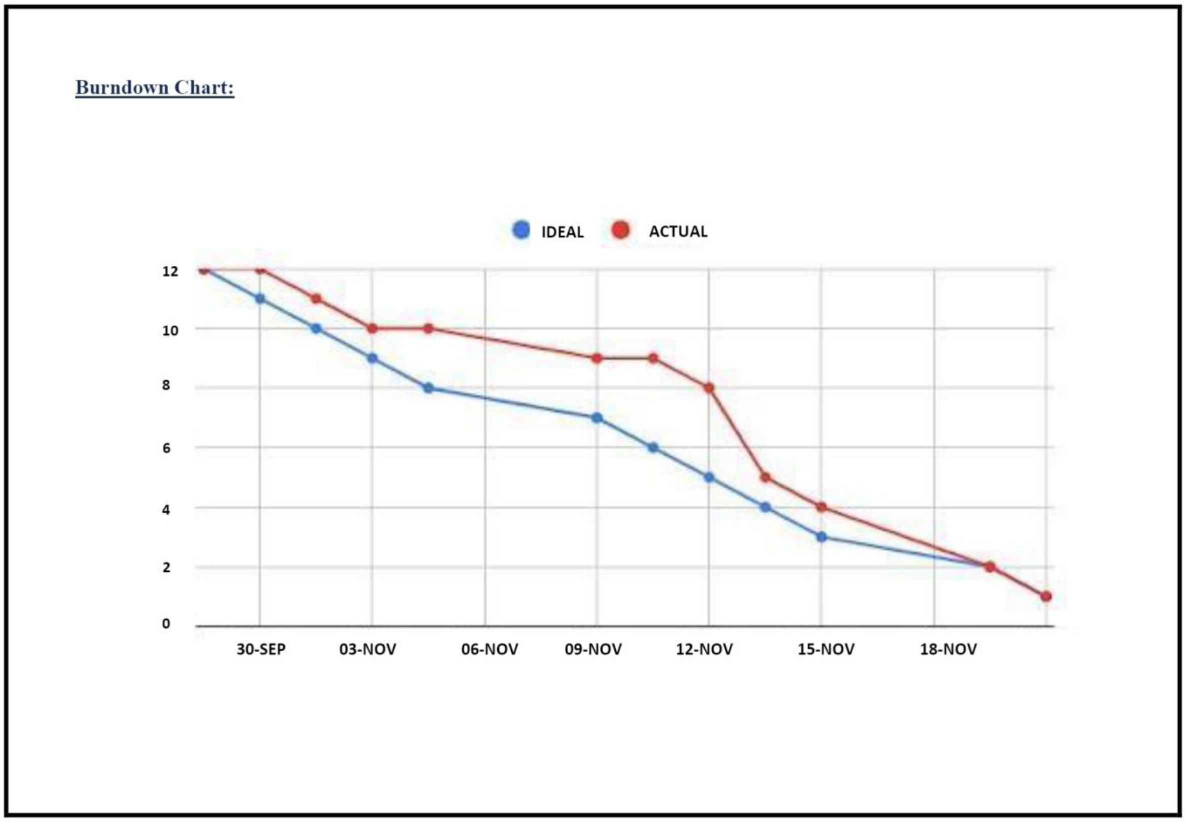
TITLE	DESCRIPTION	DATE
Ideation Phase	• Literature Survey • Empathy Map • Brainstorming • Problem Statement	29 August 2022 – 17 september 2022
Project Design Phase 1	• Problem Solution Fit • Proposed Solution • Solution Architecture	19 september 2022- 01 october 2022
Project Design Phase 2	• Requirement Analysis • Customer Journey • Data Flow Diagrams • Technical Architecture	03 october 2022- 15 october 2022
Project Planning Phase	• Sprint Delivery Plan • JIRA files	17 october 2022- 22october 2022
Project Development Phase	• Sprint 1 • Sprint 2 • Sprint 3 • Sprint 4	24 october 2022- 19 november 2022

6.2.SPRINT PLANNING AND ESTIMATION:

SPRINT	FUNCTIONAL REQUIREMENT	USER STORY NUMBER	USER STORY/TASK	STORY POINTS	PRIORITY	TEAM MEMBERS
Sprint-1	Pre-processing data	USN-1	Collect Dataset	5	High	Haribabu
Sprint-1		USN-2	Import the required libraries, Read&Clean the datasets.	5	High	Gowsalya
Sprint-2	Building the model	USN-1	Split the data into dependent and independent variables.	4	High	Gayathri

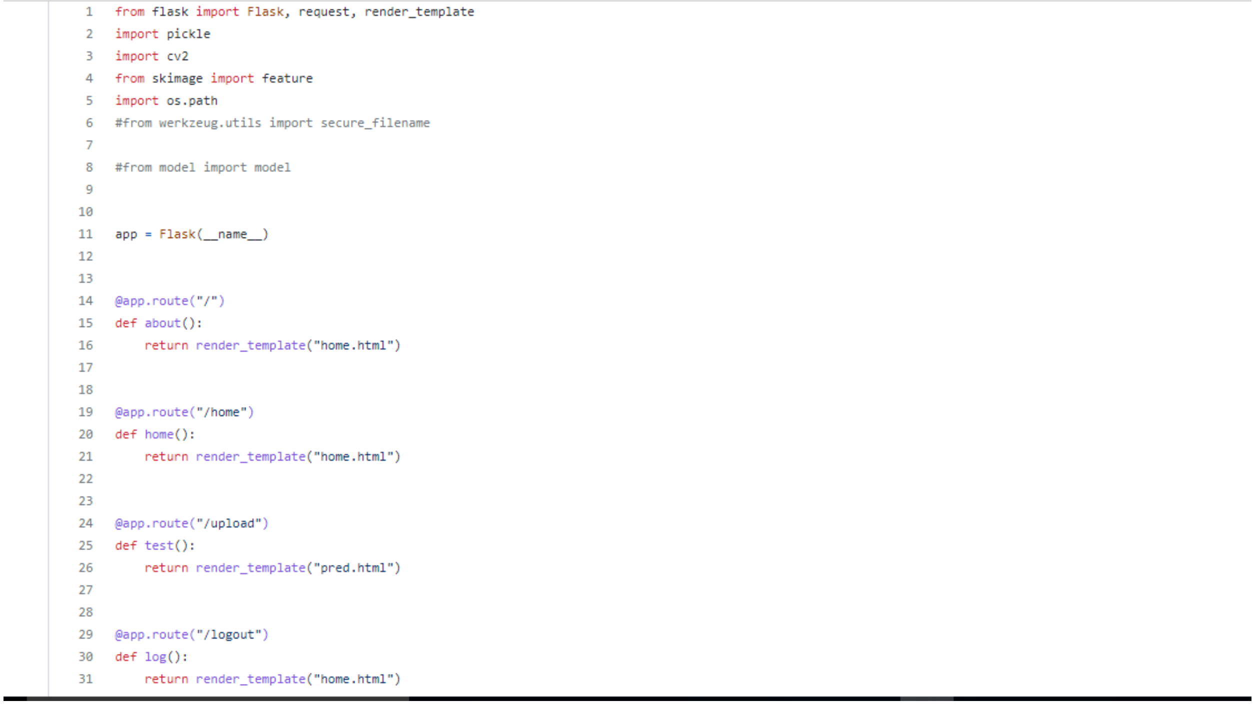
Sprint-2		USN-2	Apply using regression model.	2	Medium	Israel
Sprint-3	Application Building	USN-1	Build python flask application and HTML page.	5	High	Aarthi
Sprint-3		USN-2	Execute and test the application.	2	Medium	Darvinesh
Sprint-4	Training the model	USN-1	Train machine learning model.	5	High	Gayathri
		USN-2	Integrate flask.	5	High	Aarthi

6.3.REPORTS FROM JIRA:

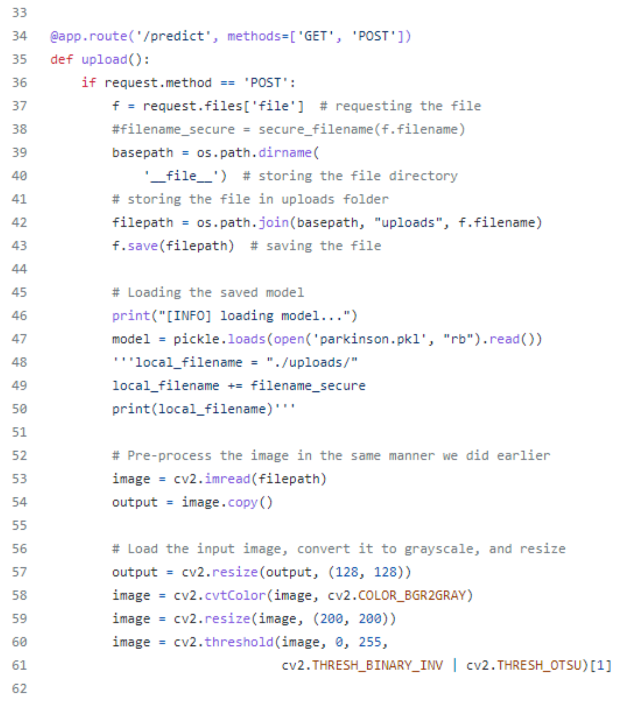


CHAPTER – 7

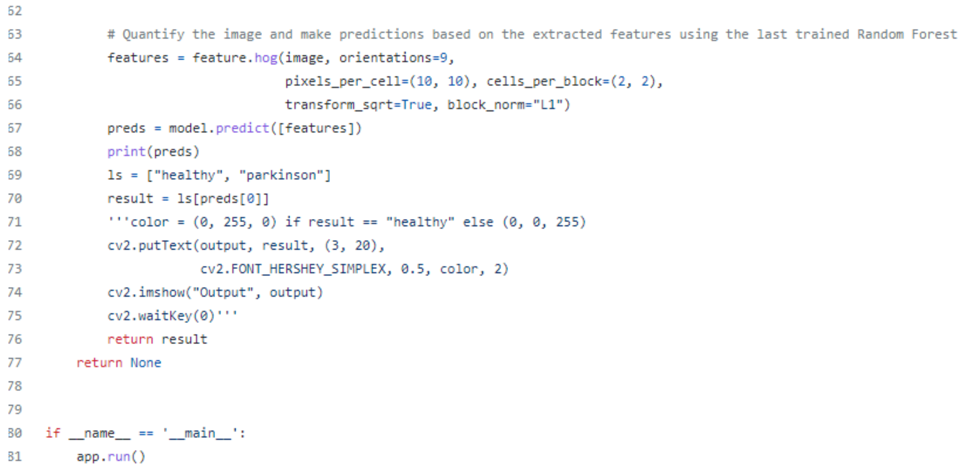
CODING AND SOLUTION



```python
from flask import Flask, request, render_template
import pickle
import cv2
from skimage import feature
import os.path
#from werkzeug.utils import secure_filename

#from model import model


app = Flask(__name__)


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


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


@app.route("/upload")
def test():
    return render_template("pred.html")


@app.route("/logout")
def log():
    return render_template("home.html")
```



```python

@app.route('/predict', methods=['GET', 'POST'])
def upload():
    if request.method == 'POST':
        f = request.files['file']  # requesting the file
        #filename_secure = secure_filename(f.filename)
        basepath = os.path.dirname(
            '__file__')  # storing the file directory
        # storing the file in uploads folder
        filepath = os.path.join(basepath, "uploads", f.filename)
        f.save(filepath)  # saving the file

        # Loading the saved model
        print("[INFO] loading model...")
        model = pickle.loads(open('parkinson.pkl', "rb").read())
        '''local_filename = "./uploads/"
        local_filename += filename_secure
        print(local_filename)'''

        # Pre-process the image in the same manner we did earlier
        image = cv2.imread(filepath)
        output = image.copy()

        # Load the input image, convert it to grayscale, and resize
        output = cv2.resize(output, (128, 128))
        image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        image = cv2.resize(image, (200, 200))
        image = cv2.threshold(image, 0, 255,
                              cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]

```

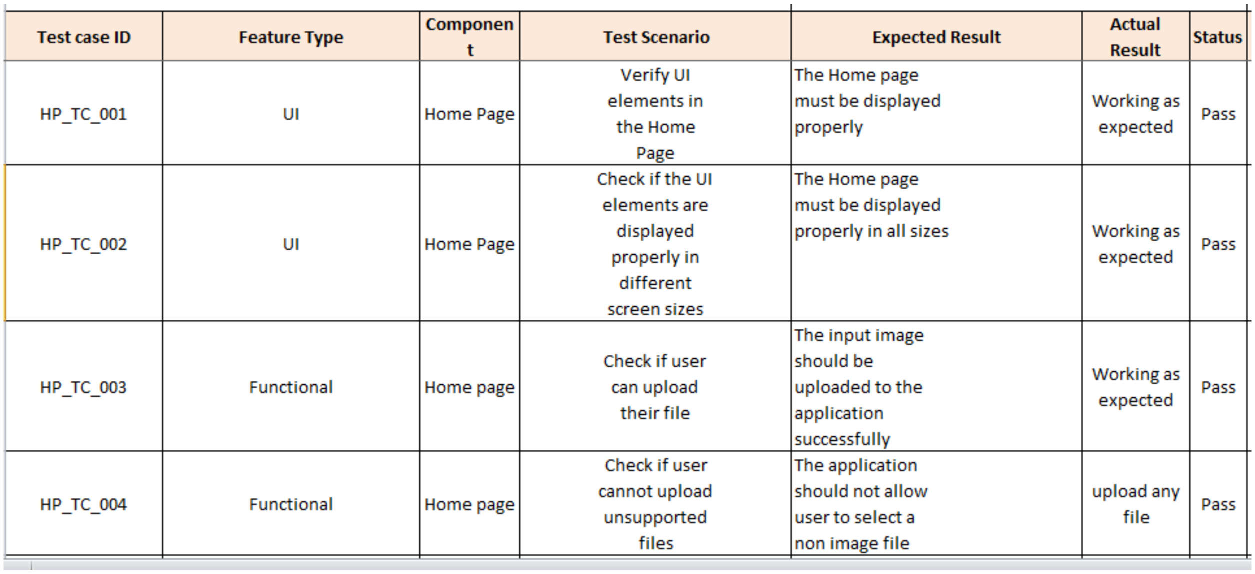


```python

        # Quantify the image and make predictions based on the extracted features using the last trained Random Forest
        features = feature.hog(image, orientations=9,
                               pixels_per_cell=(10, 10), cells_per_block=(2, 2),
                               transform_sqrt=True, block_norm="L1")
        preds = model.predict([features])
        print(preds)
        ls = ["healthy", "parkinson"]
        result = ls[preds[0]]
        '''color = (0, 255, 0) if result == "healthy" else (0, 0, 255)
        cv2.putText(output, result, (3, 20),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.5, color, 2)
        cv2.imshow("Output", output)
        cv2.waitKey(0)'''
        return result
    return None


if __name__ == '__main__':
    app.run()
```

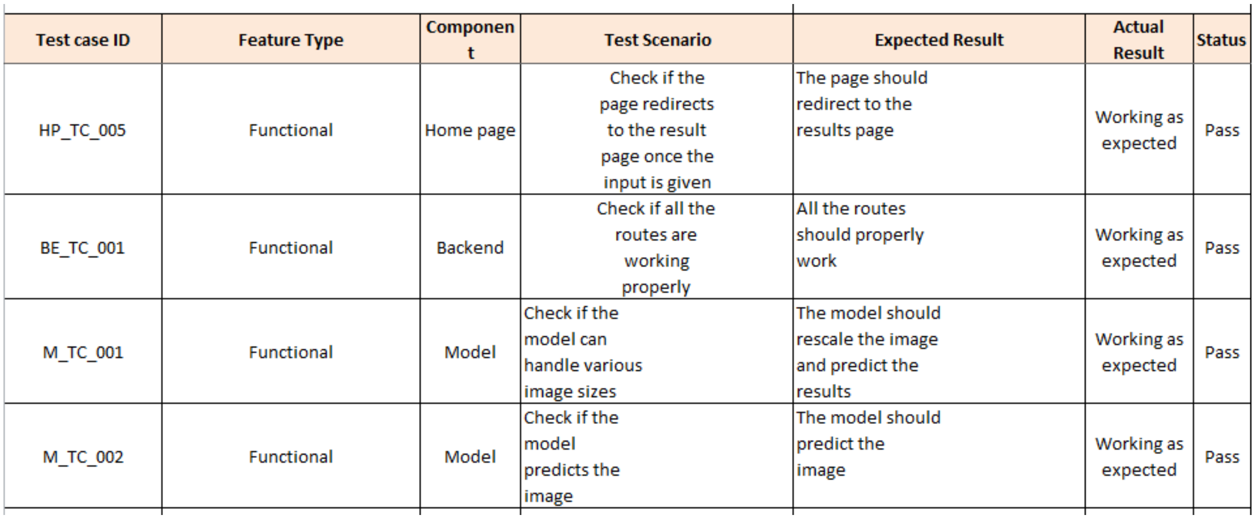
CHAPTER – 8

TESTING

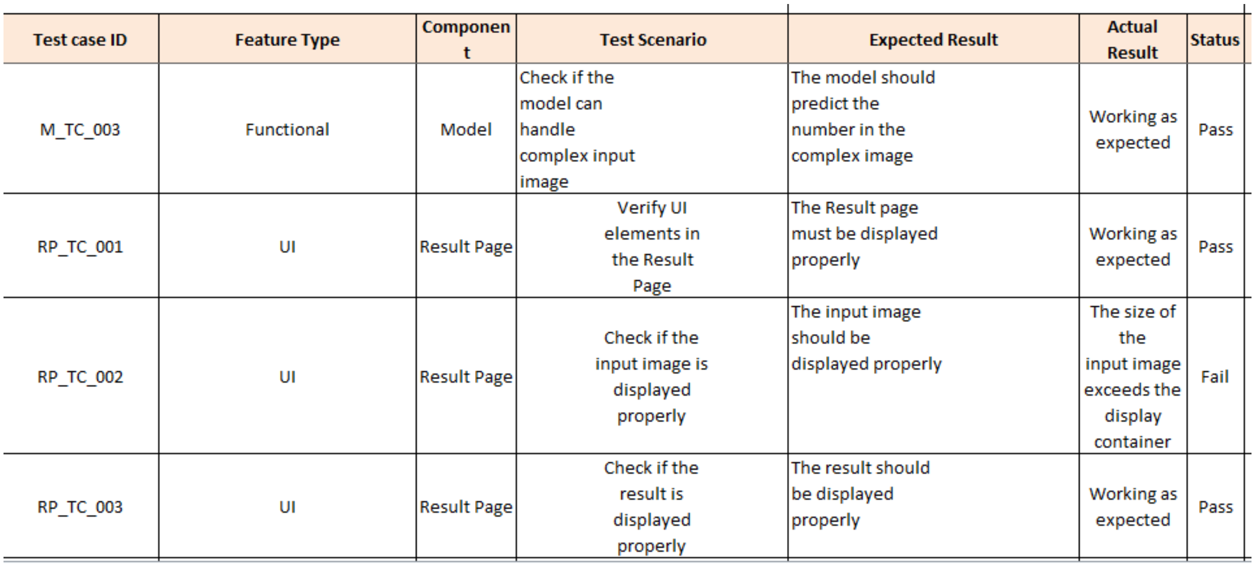
8.1.TEST CASES:



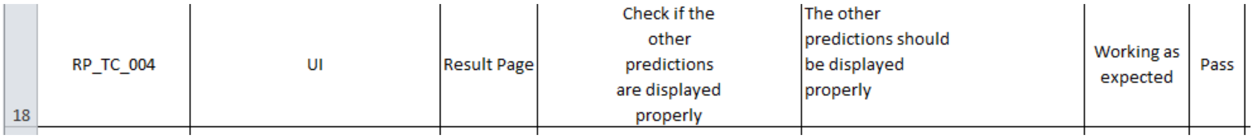
Test case ID	Feature Type	Component	Test Scenario	Expected Result	Actual Result	Status
HP_TC_001	UI	Home Page	Verify UI elements in the Home Page	The Home page must be displayed properly	Working as expected	Pass
HP_TC_002	UI	Home Page	Check if the UI elements are displayed properly in different screen sizes	The Home page must be displayed properly in all sizes	Working as expected	Pass
HP_TC_003	Functional	Home page	Check if user can upload their file	The input image should be uploaded to the application successfully	Working as expected	Pass
HP_TC_004	Functional	Home page	Check if user cannot upload unsupported files	The application should not allow user to select a non image file	upload any file	Pass



Test case ID	Feature Type	Component	Test Scenario	Expected Result	Actual Result	Status
HP_TC_005	Functional	Home page	Check if the page redirects to the result page once the input is given	The page should redirect to the results page	Working as expected	Pass
BE_TC_001	Functional	Backend	Check if all the routes are working properly	All the routes should properly work	Working as expected	Pass
M_TC_001	Functional	Model	Check if the model can handle various image sizes	The model should rescale the image and predict the results	Working as expected	Pass
M_TC_002	Functional	Model	Check if the model predicts the image	The model should predict the image	Working as expected	Pass



Test case ID	Feature Type	Component	Test Scenario	Expected Result	Actual Result	Status
M_TC_003	Functional	Model	Check if the model can handle complex input image	The model should predict the number in the complex image	Working as expected	Pass
RP_TC_001	UI	Result Page	Verify UI elements in the Result Page	The Result page must be displayed properly	Working as expected	Pass
RP_TC_002	UI	Result Page	Check if the input image is displayed properly	The input image should be displayed properly	The size of the input image exceeds the display container	Fail
RP_TC_003	UI	Result Page	Check if the result is displayed properly	The result should be displayed properly	Working as expected	Pass



RP_TC_004	UI	Result Page	Check if the other predictions are displayed properly	The other predictions should be displayed properly	Working as expected	Pass

8.2.USER ACCEPTANCE TESTING:

8.2.1.DEFECT ANALYSIS:

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Severity 5
By Design	1	0	1	0	2
Duplicate	0	0	0	0	0
External	0	0	2	0	2
Fixed	4	1	0	1	6
Not Reproduced	0	0	0	1	1
Skipped	0	0	0	1	1
Won’t Fix	1	0	1	0	2
Total	6	1	4	3	14

8.2.2.TEST CASE ANALYSIS:

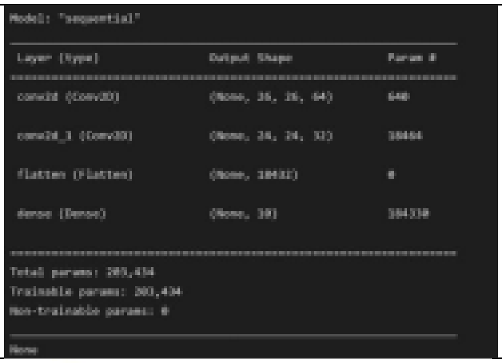
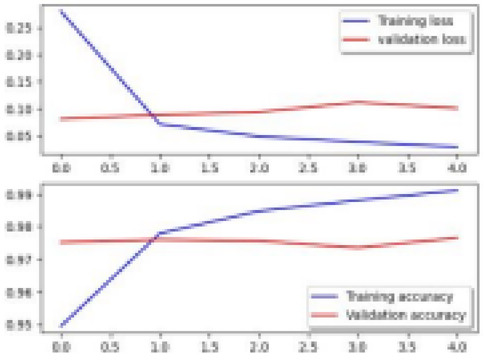
SECTION	TOTAL CASES	NOT TESTED	FAIL	PASS
Client Application	10	0	3	7
Security	2	0	1	1
Performance	3	0	1	2

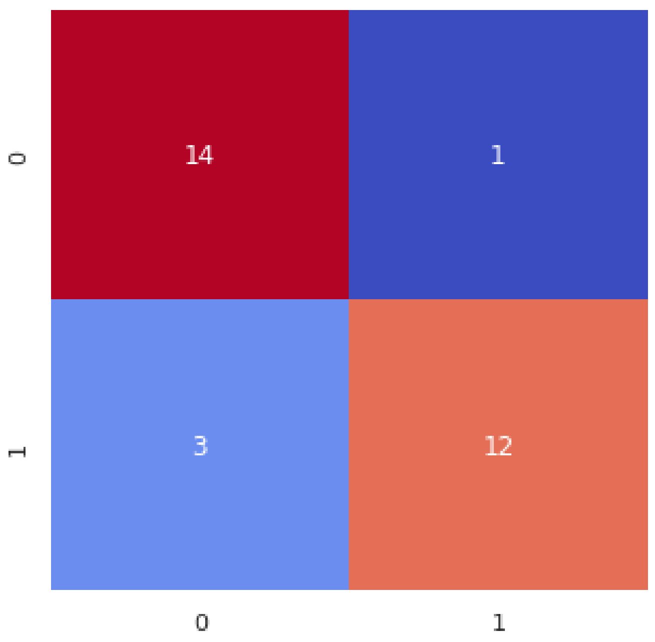
Exception Reporting	2	0	0	2

CHAPTER – 9

RESULT

9.1.PERFORMANCE METRICES:

S.NO	PARAMETER	VALUES	SCREENSHOT
1.	MODEL SUMMARY		 Model: "sequential" Layer (type) Output Shape Param # conv2d (Conv2D) (None, 26, 26, 64) 640 conv2d_1 (Conv2D) (None, 24, 24, 32) 18464 flatten (Flatten) (None, 18432) 0 dense (Dense) (None, 10) 184330 Total params: 203,434 Trainable params: 203,434 Non-trainable params: 0 None
2.	ACCURACY	Training Accuracy-98% Validation Accuracy-97%	

3.	CONFUSION MATRIX		

CHAPTER – 10

ADVANTAGES AND DISADVANTAGES

ADVANTAGES:

● Reduces manual work

● More accurate than average human

● Capable of handling a lot of data

● Can be used anywhere from any device

DISADVANTAGES:

● Cannot handle complex data

● All the data must be in image format

● Requires a high performance server for faster predictions

● Prone to occasional errors

CHAPTER – 11

CONCLUSION

Parkinson’s Disease is a totally grave disease and has no cure till date. since it impacts the actions of the parts of the body, the speech additionally stands affected. here, the gadget tries to offer a way of detecting Parkinson’s ailment so one can bring about a quick action to reduce or even put off it from affecting the whole body. This gadget aims to make this method of expertise a case of Parkinson’s on the earliest via each, the affected person as well as scientific experts. hence, the goal is to apply numerous machine getting to know strategies like Random Forest Classifier , CNN, for buying the maximum accurate result. Here using Decision Tree and building a classifier results in an accuracy of 98%.

CHAPTER – 12

FUTURE SCOPE

This project is far from complete and there is a lot of room for improvement.

Some of the improvements that can be made to this project are as follows:

- Add support to detect from multiple images and save the results
- Add support to detect multiple images
- Improve model to detect from complex images

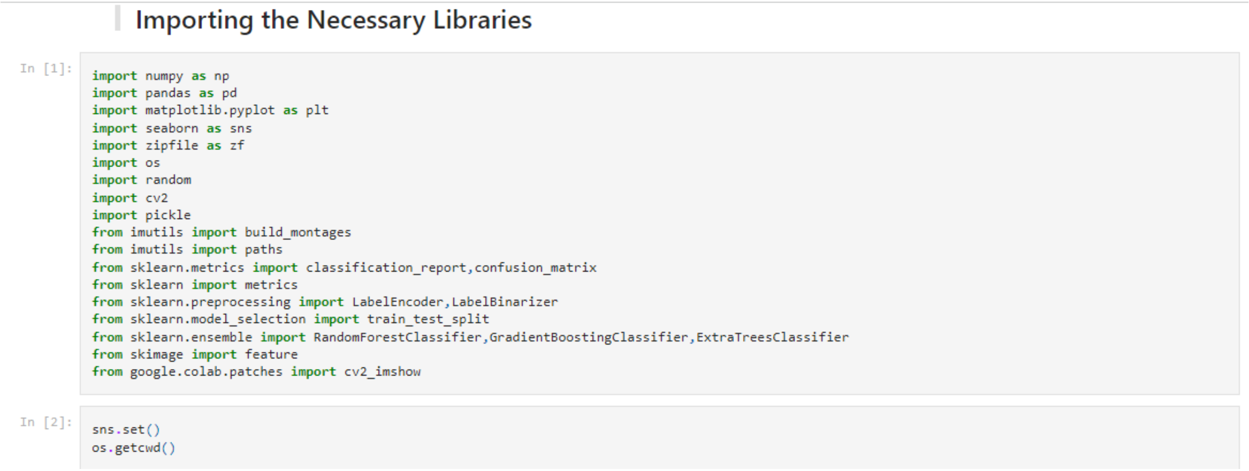
This project has endless potential and can always be enhanced to become better. Implementing this concept in the real world will benefit several industries and reduce the workload on many workers, enhancing overall work efficiency.

CHAPTER – 13

APPENDIX

SOURCE CODE:

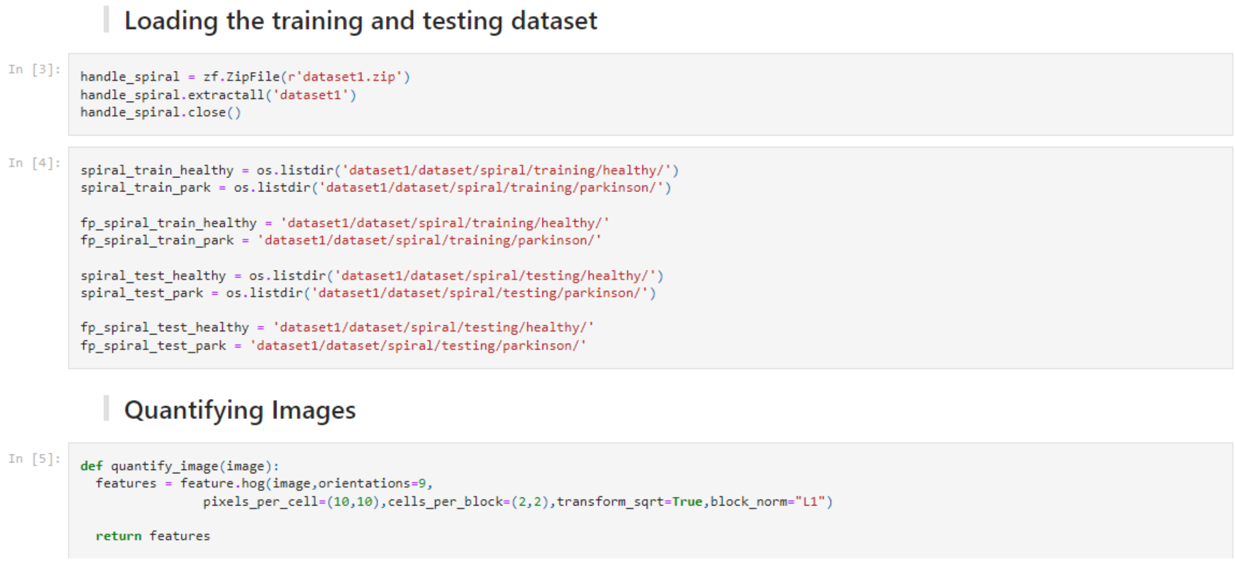
MODEL CREATION



Importing the Necessary Libraries

In [1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import zipfile as zf
import os
import random
import cv2
import pickle
from imutils import build_montages
from imutils import paths
from sklearn.metrics import classification_report,confusion_matrix
from sklearn import metrics
from sklearn.preprocessing import LabelEncoder,LabelBinarizer
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier,GradientBoostingClassifier,ExtraTreesClassifier
from skimage import feature
from google.colab.patches import cv2_imshow
```

In [2]:
```
sns.set()
os.getcwd()
```



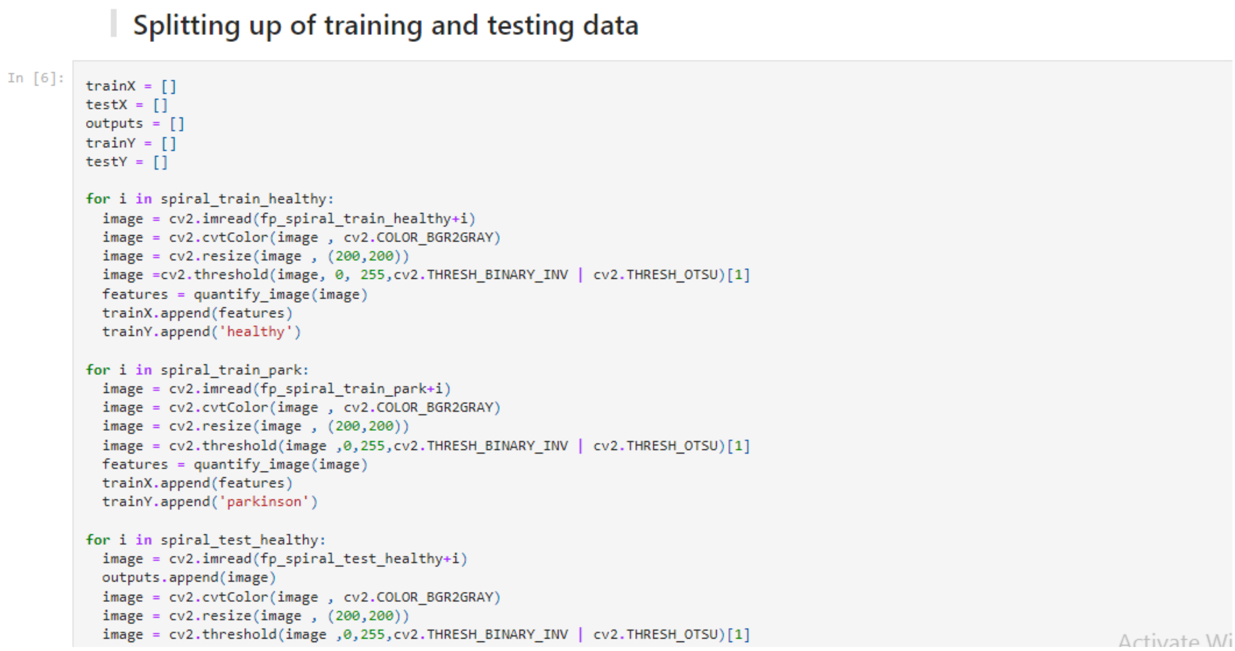
Loading the training and testing dataset

In [3]:
```
handle_spiral = zf.ZipFile(r'dataset1.zip')
handle_spiral.extractall('dataset1')
handle_spiral.close()
```

In [4]:
```
spiral_train_healthy = os.listdir('dataset1/dataset/spiral/training/healthy/')
spiral_train_park = os.listdir('dataset1/dataset/spiral/training/parkinson/')

fp_spiral_train_healthy = 'dataset1/dataset/spiral/training/healthy/'
fp_spiral_train_park = 'dataset1/dataset/spiral/training/parkinson/'

spiral_test_healthy = os.listdir('dataset1/dataset/spiral/testing/healthy/')
spiral_test_park = os.listdir('dataset1/dataset/spiral/testing/parkinson/')

fp_spiral_test_healthy = 'dataset1/dataset/spiral/testing/healthy/'
fp_spiral_test_park = 'dataset1/dataset/spiral/testing/parkinson/'
```

Quantifying Images

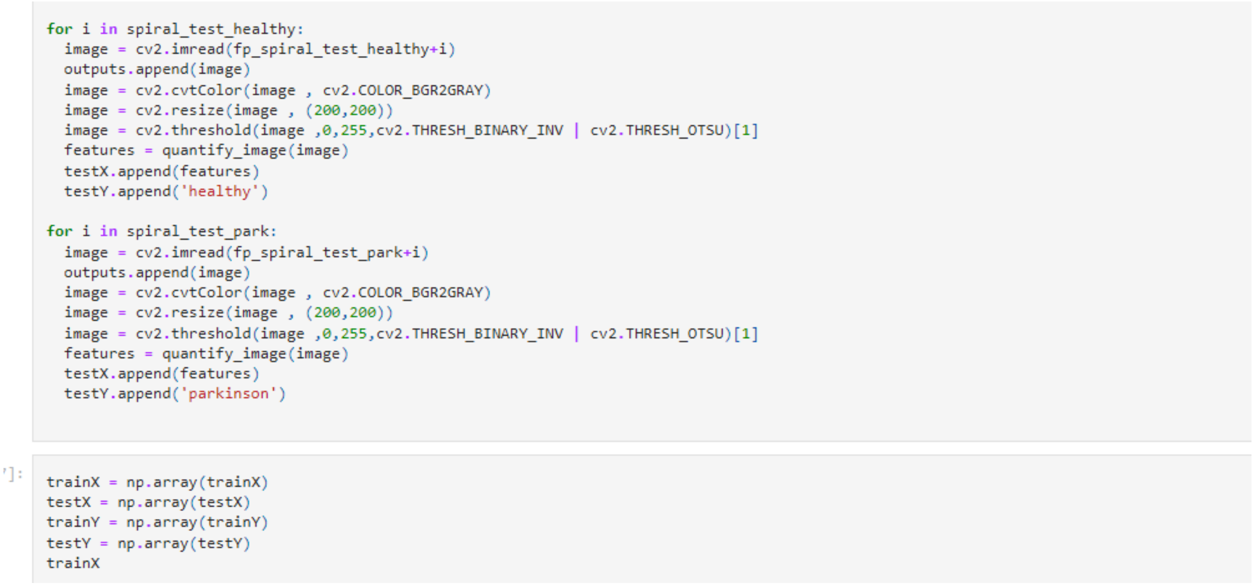
In [5]:
```
def quantify_image(image):
  features = feature.hog(image,orientations=9,
                pixels_per_cell=(10,10),cells_per_block=(2,2),transform_sqrt=True,block_norm="L1")

  return features
```



Splitting up of training and testing data

In [6]:

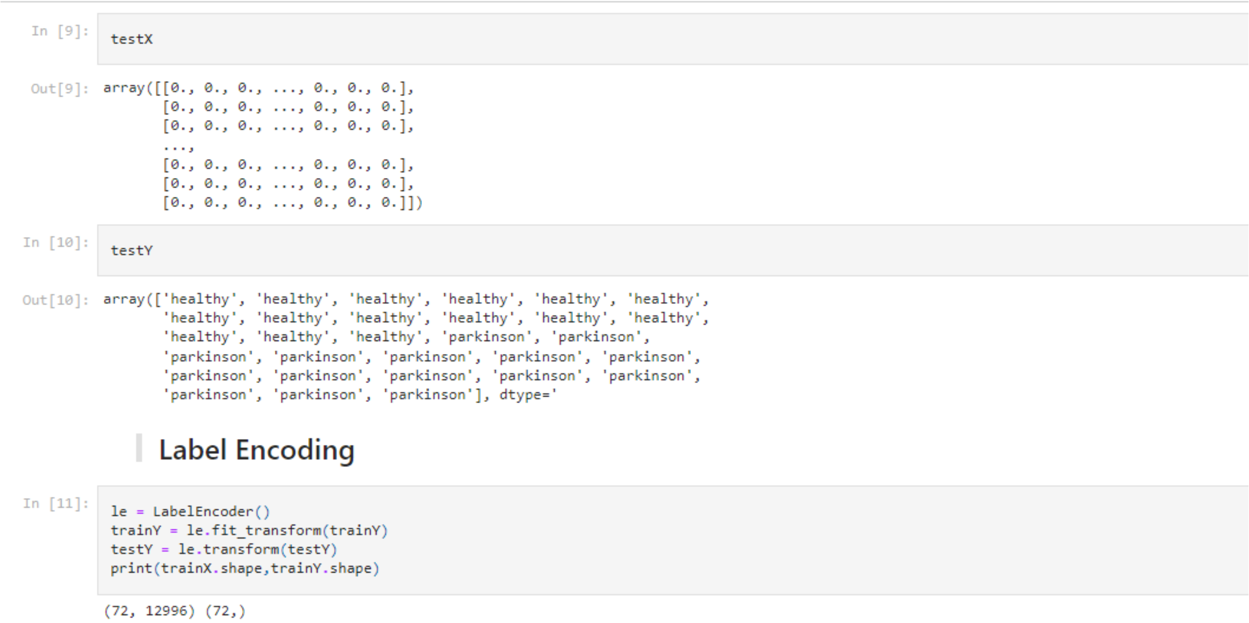
```
trainX = []
testX = []
outputs = []
trainY = []
testY = []

for i in spiral_train_healthy:
  image = cv2.imread(fp_spiral_train_healthy+i)
  image = cv2.cvtColor(image , cv2.COLOR_BGR2GRAY)
  image = cv2.resize(image , (200,200))
  image =cv2.threshold(image, 0, 255,cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]
  features = quantify_image(image)
  trainX.append(features)
  trainY.append('healthy')

for i in spiral_train_park:
  image = cv2.imread(fp_spiral_train_park+i)
  image = cv2.cvtColor(image , cv2.COLOR_BGR2GRAY)
  image = cv2.resize(image , (200,200))
  image = cv2.threshold(image ,0,255,cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]
  features = quantify_image(image)
  trainX.append(features)
  trainY.append('parkinson')

for i in spiral_test_healthy:
  image = cv2.imread(fp_spiral_test_healthy+i)
  outputs.append(image)
  image = cv2.cvtColor(image , cv2.COLOR_BGR2GRAY)
  image = cv2.resize(image , (200,200))
  image = cv2.threshold(image ,0,255,cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]
```



```
for i in spiral_test_healthy:
  image = cv2.imread(fp_spiral_test_healthy+i)
  outputs.append(image)
  image = cv2.cvtColor(image , cv2.COLOR_BGR2GRAY)
  image = cv2.resize(image , (200,200))
  image = cv2.threshold(image ,0,255,cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]
  features = quantify_image(image)
  testX.append(features)
  testY.append('healthy')

for i in spiral_test_park:
  image = cv2.imread(fp_spiral_test_park+i)
  outputs.append(image)
  image = cv2.cvtColor(image , cv2.COLOR_BGR2GRAY)
  image = cv2.resize(image , (200,200))
  image = cv2.threshold(image ,0,255,cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]
  features = quantify_image(image)
  testX.append(features)
  testY.append('parkinson')
```

']:

```
trainX = np.array(trainX)
testX = np.array(testX)
trainY = np.array(trainY)
testY = np.array(testY)
trainX
```



```
In [9]:  testX
```

```
Out[9]: array([[0., 0., 0., ..., 0., 0., 0.],
               [0., 0., 0., ..., 0., 0., 0.],
               [0., 0., 0., ..., 0., 0., 0.],
               ...,
               [0., 0., 0., ..., 0., 0., 0.],
               [0., 0., 0., ..., 0., 0., 0.],
               [0., 0., 0., ..., 0., 0., 0.]])
```

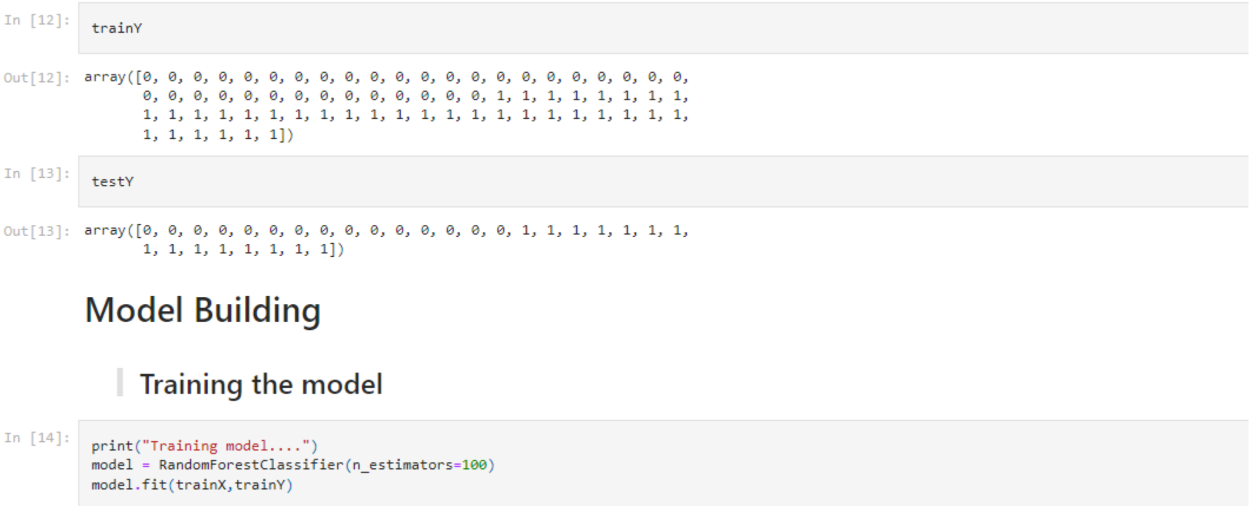
```
In [10]:  testY
```

```
Out[10]: array(['healthy', 'healthy', 'healthy', 'healthy', 'healthy', 'healthy',
                'healthy', 'healthy', 'healthy', 'healthy', 'healthy', 'healthy',
                'healthy', 'healthy', 'healthy', 'parkinson', 'parkinson',
                'parkinson', 'parkinson', 'parkinson', 'parkinson', 'parkinson',
                'parkinson', 'parkinson', 'parkinson', 'parkinson', 'parkinson',
                'parkinson', 'parkinson', 'parkinson'], dtype='
```

Label Encoding

```
In [11]:  le = LabelEncoder()
          trainY = le.fit_transform(trainY)
          testY = le.transform(testY)
          print(trainX.shape,trainY.shape)
```

```
(72, 12996) (72,)
```



```
In [12]:  trainY
```

```
Out[12]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
                0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1])
```

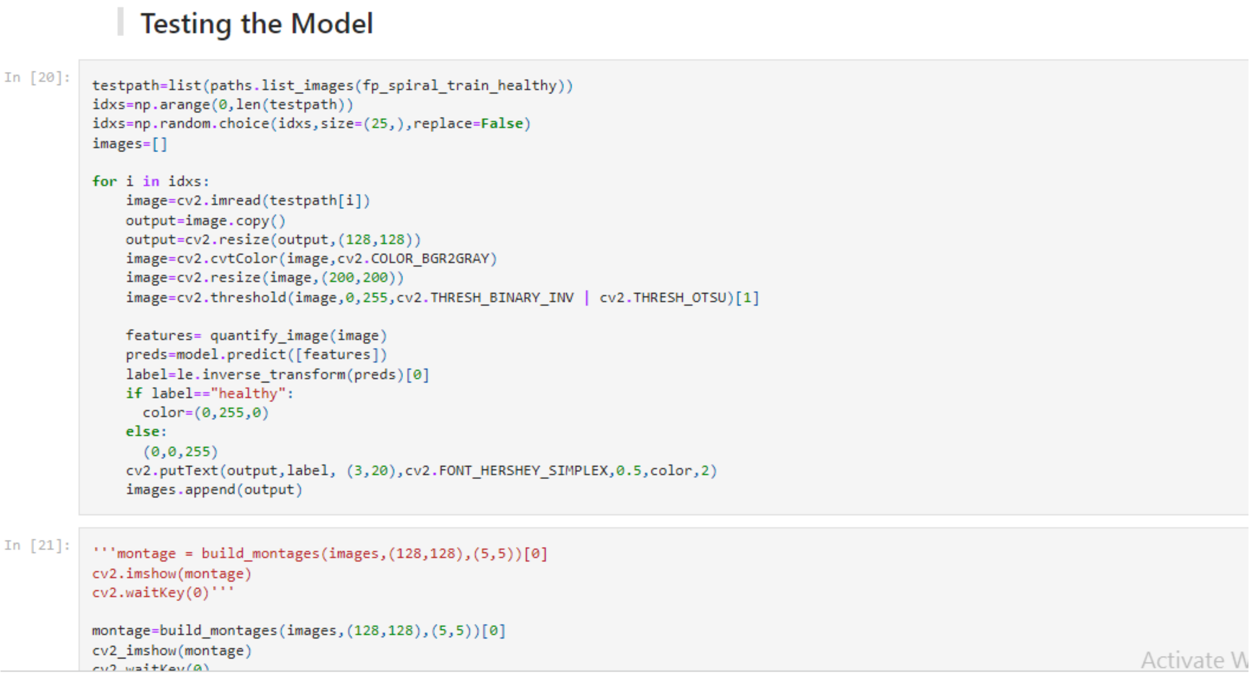
```
In [13]:  testY
```

```
Out[13]: array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 1, 1, 1, 1, 1])
```

Model Building

Training the model

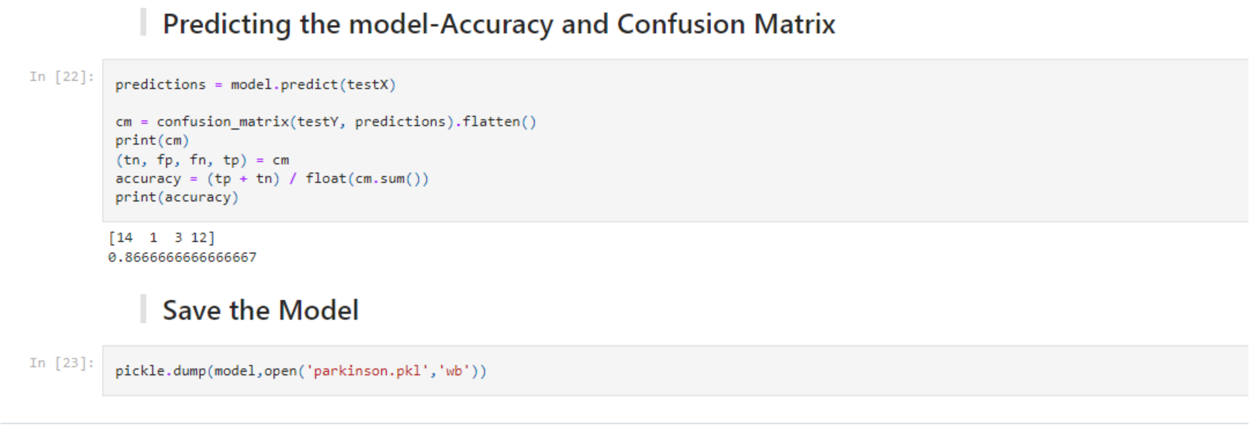
```
In [14]:  print("Training model....")
          model = RandomForestClassifier(n_estimators=100)
          model.fit(trainX,trainY)
```



Testing the Model

```
In [20]:
testpath=list(paths.list_images(fp_spiral_train_healthy))
idxs=np.arange(0,len(testpath))
idxs=np.random.choice(idxs,size=(25,),replace=False)
images=[]

for i in idxs:
    image=cv2.imread(testpath[i])
    output=image.copy()
    output=cv2.resize(output,(128,128))
    image=cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
    image=cv2.resize(image,(200,200))
    image=cv2.threshold(image,0,255,cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]

    features= quantify_image(image)
    preds=model.predict([features])
    label=le.inverse_transform(preds)[0]
    if label=="healthy":
      color=(0,255,0)
    else:
      (0,0,255)
    cv2.putText(output,label, (3,20),cv2.FONT_HERSHEY_SIMPLEX,0.5,color,2)
    images.append(output)
```

```
In [21]:
'''montage = build_montages(images,(128,128),(5,5))[0]
cv2.imshow(montage)
cv2.waitKey(0)'''

montage=build_montages(images,(128,128),(5,5))[0]
cv2_imshow(montage)
cv2.waitKey(0)
```



Predicting the model-Accuracy and Confusion Matrix

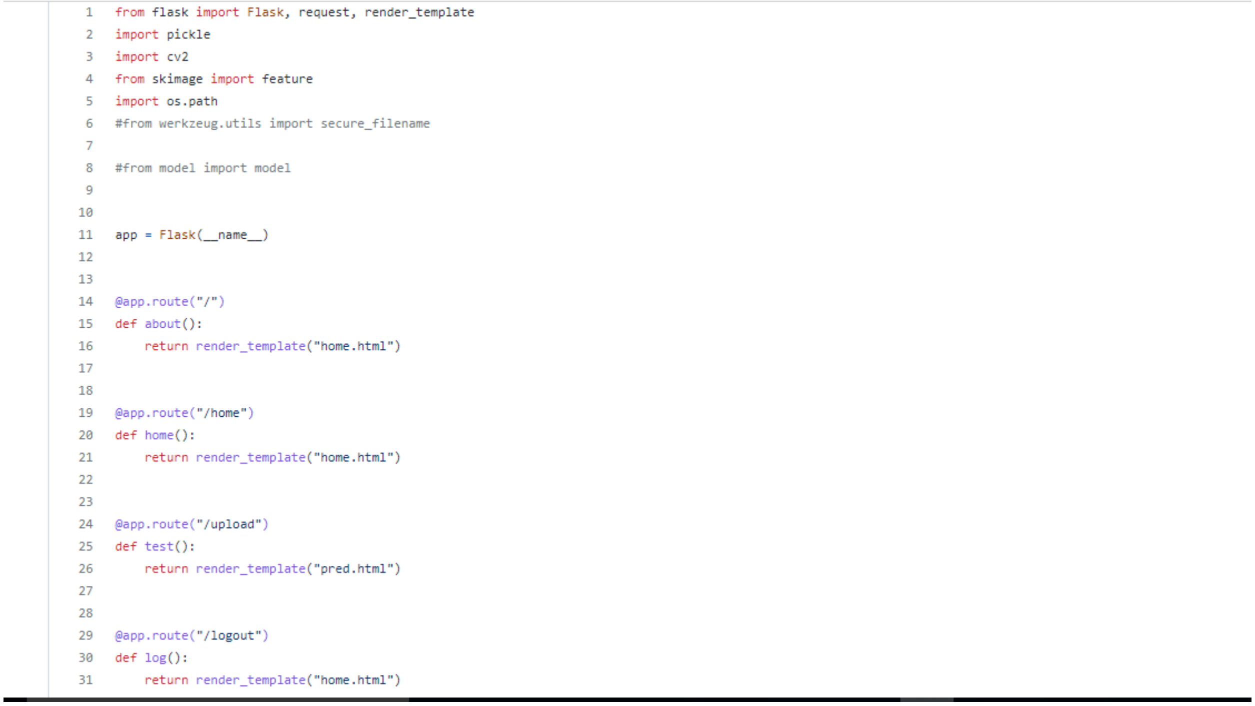
```
In [22]:
predictions = model.predict(testX)

cm = confusion_matrix(testY, predictions).flatten()
print(cm)
(tn, fp, fn, tp) = cm
accuracy = (tp + tn) / float(cm.sum())
print(accuracy)
```

```
[14  1  3 12]
0.8666666666666667
```

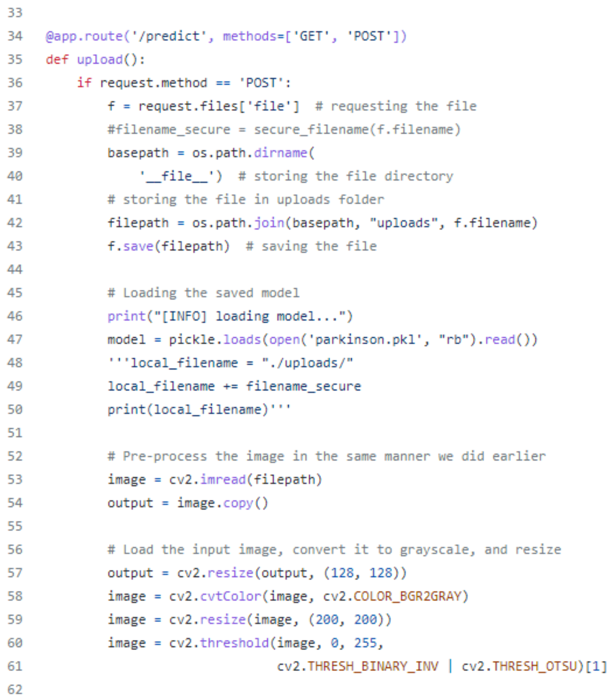
Save the Model

```
In [23]:
pickle.dump(model,open('parkinson.pkl','wb'))
```

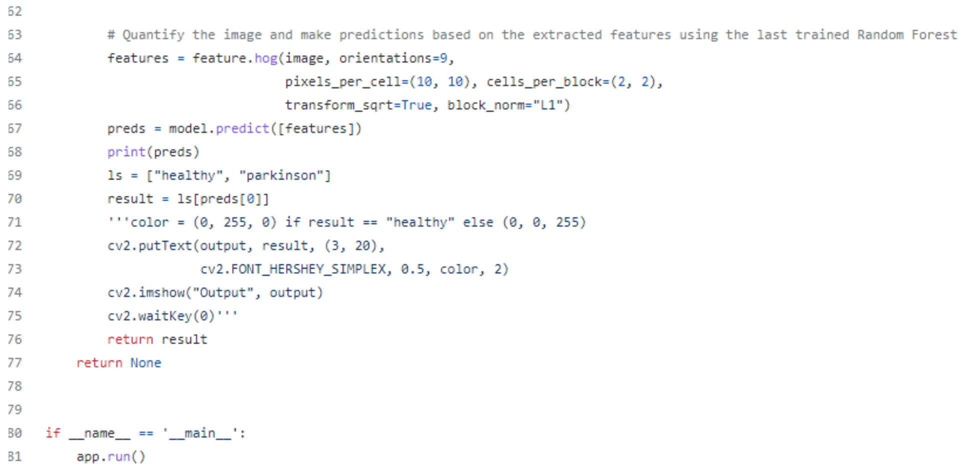
FLASK APP



```python
from flask import Flask, request, render_template
import pickle
import cv2
from skimage import feature
import os.path
#from werkzeug.utils import secure_filename

#from model import model


app = Flask(__name__)


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


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


@app.route("/upload")
def test():
    return render_template("pred.html")


@app.route("/logout")
def log():
    return render_template("home.html")
```

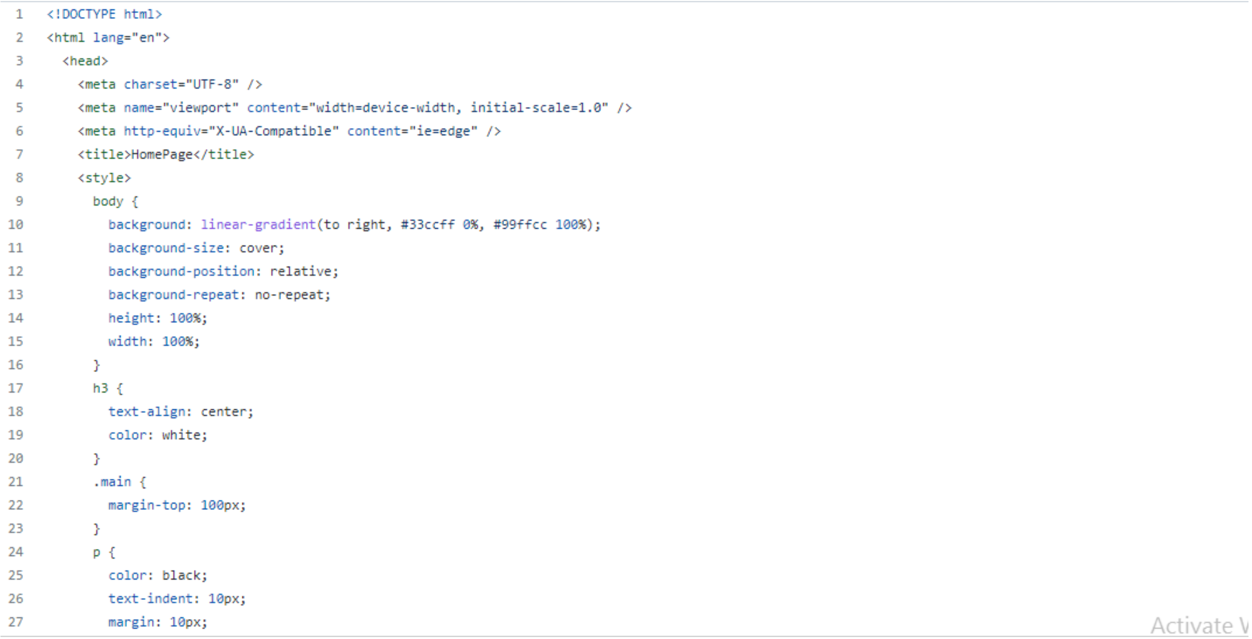


```python

@app.route('/predict', methods=['GET', 'POST'])
def upload():
    if request.method == 'POST':
        f = request.files['file']  # requesting the file
        #filename_secure = secure_filename(f.filename)
        basepath = os.path.dirname(
            '__file__')  # storing the file directory
        # storing the file in uploads folder
        filepath = os.path.join(basepath, "uploads", f.filename)
        f.save(filepath)  # saving the file

        # Loading the saved model
        print("[INFO] loading model...")
        model = pickle.loads(open('parkinson.pkl', "rb").read())
        '''local_filename = "./uploads/"
        local_filename += filename_secure
        print(local_filename)'''

        # Pre-process the image in the same manner we did earlier
        image = cv2.imread(filepath)
        output = image.copy()

        # Load the input image, convert it to grayscale, and resize
        output = cv2.resize(output, (128, 128))
        image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        image = cv2.resize(image, (200, 200))
        image = cv2.threshold(image, 0, 255,
                                      cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]

```

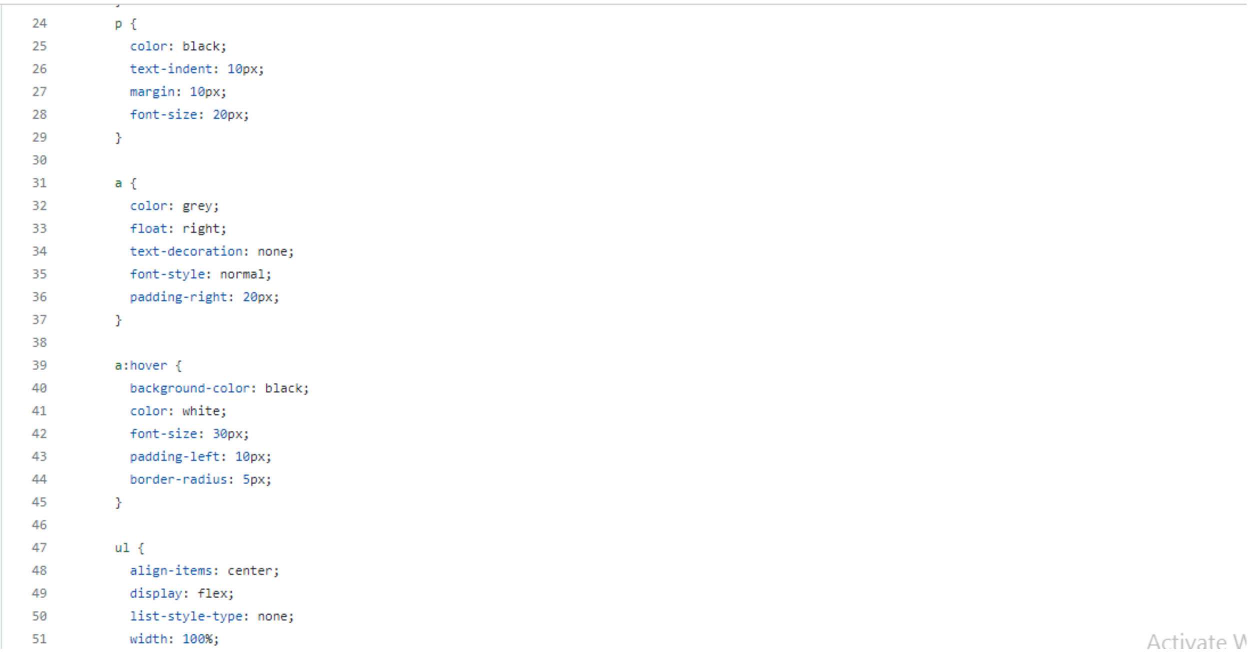


```python

        # Quantify the image and make predictions based on the extracted features using the last trained Random Forest
        features = feature.hog(image, orientations=9,
                               pixels_per_cell=(10, 10), cells_per_block=(2, 2),
                               transform_sqrt=True, block_norm="L1")
        preds = model.predict([features])
        print(preds)
        ls = ["healthy", "parkinson"]
        result = ls[preds[0]]
        '''color = (0, 255, 0) if result == "healthy" else (0, 0, 255)
        cv2.putText(output, result, (3, 20),
                    cv2.FONT_HERSHEY_SIMPLEX, 0.5, color, 2)
        cv2.imshow("Output", output)
        cv2.waitKey(0)'''
        return result
    return None


if __name__ == '__main__':
    app.run()
```

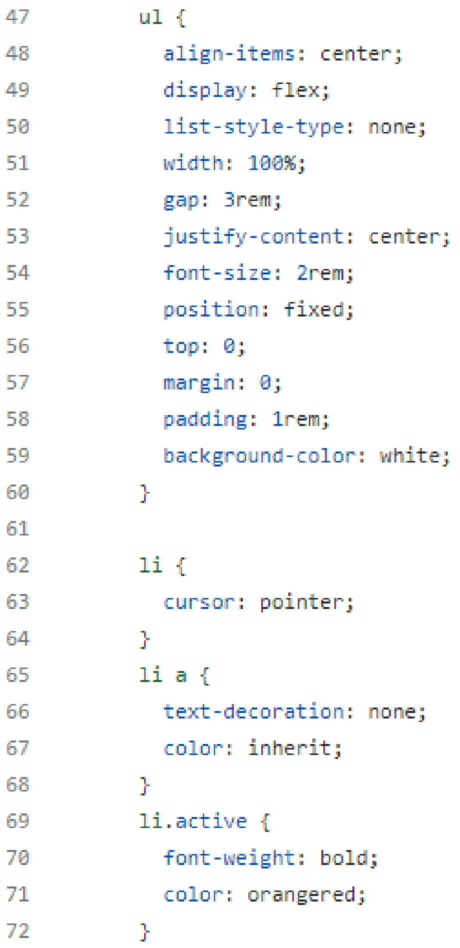
HOME PAGE(HTML)



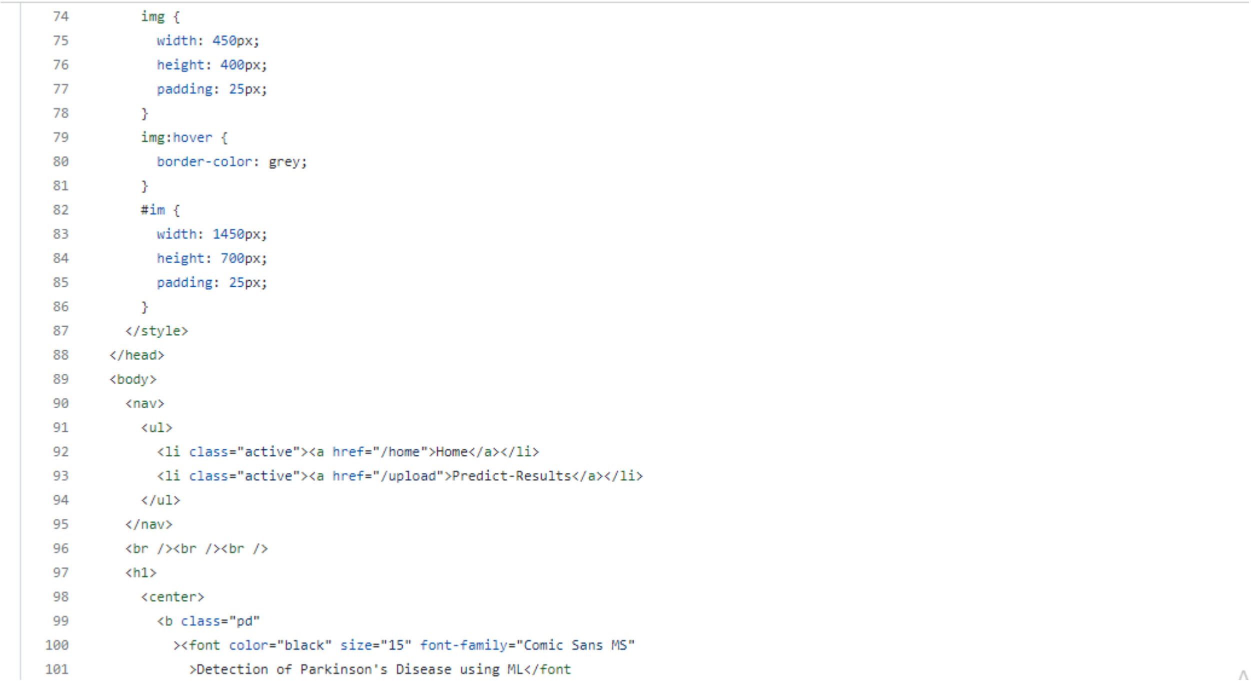
```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <title>HomePage</title>
    <style>
      body {
        background: linear-gradient(to right, #33ccff 0%, #99ffcc 100%);
        background-size: cover;
        background-position: relative;
        background-repeat: no-repeat;
        height: 100%;
        width: 100%;
      }
      h3 {
        text-align: center;
        color: white;
      }
      .main {
        margin-top: 100px;
      }
      p {
        color: black;
        text-indent: 10px;
        margin: 10px;
```



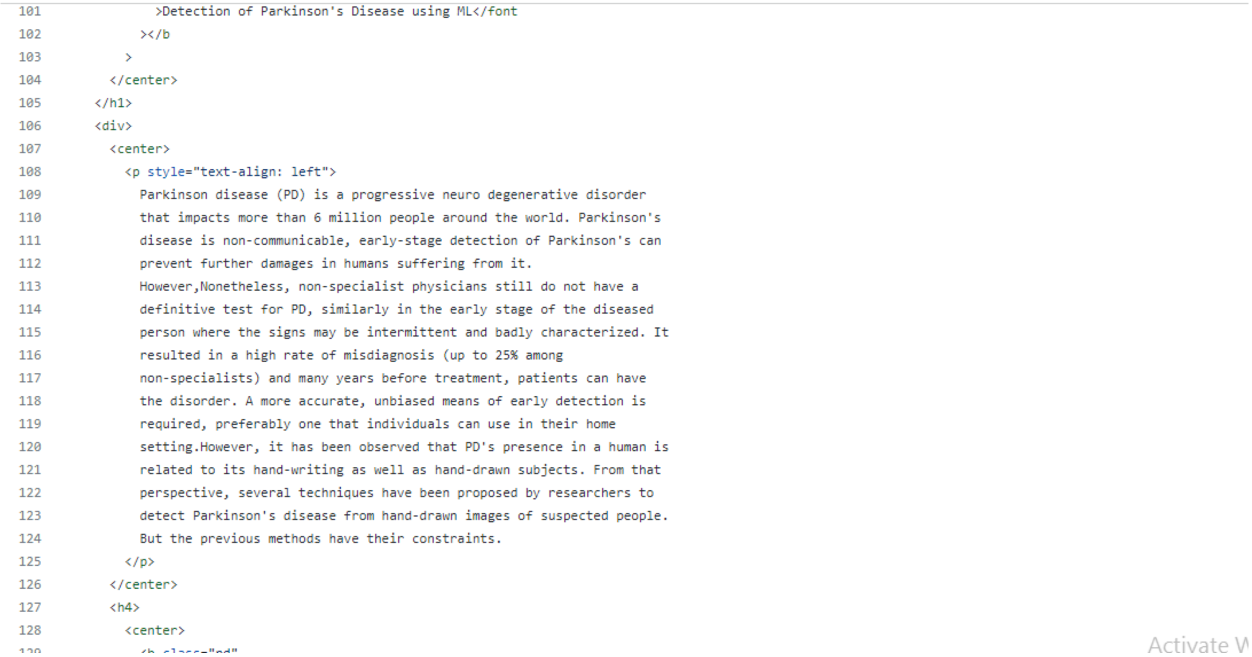
```
p {
  color: black;
  text-indent: 10px;
  margin: 10px;
  font-size: 20px;
}

a {
  color: grey;
  float: right;
  text-decoration: none;
  font-style: normal;
  padding-right: 20px;
}

a:hover {
  background-color: black;
  color: white;
  font-size: 30px;
  padding-left: 10px;
  border-radius: 5px;
}

ul {
  align-items: center;
  display: flex;
  list-style-type: none;
  width: 100%;
```



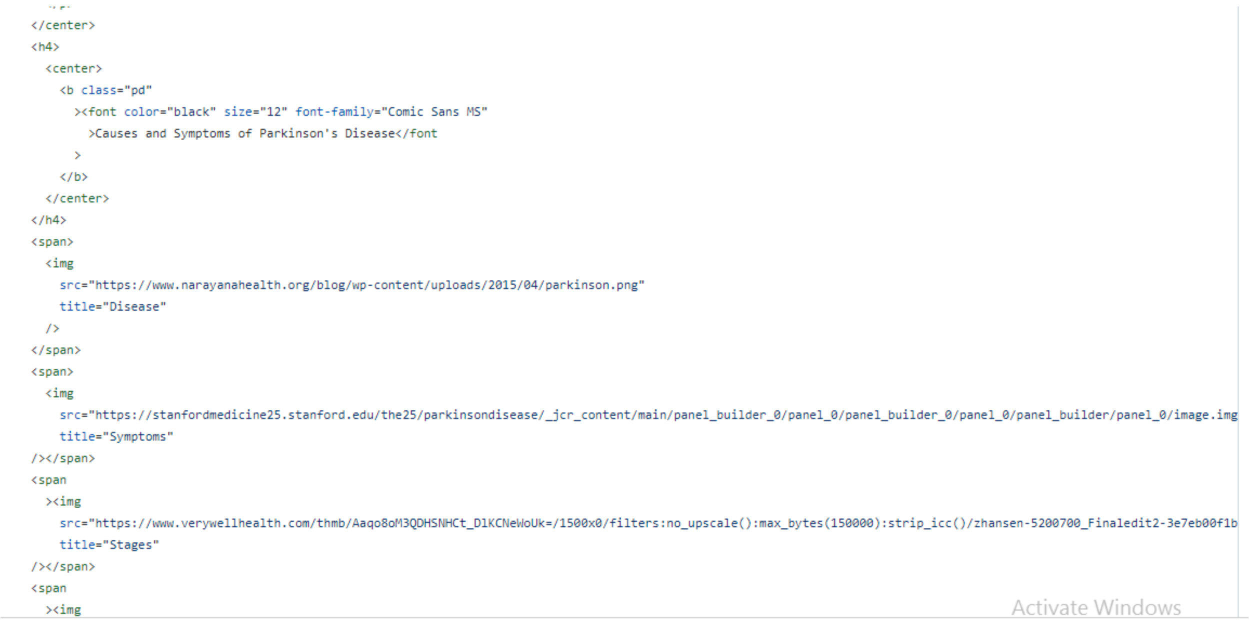
```
ul {
  align-items: center;
  display: flex;
  list-style-type: none;
  width: 100%;
  gap: 3rem;
  justify-content: center;
  font-size: 2rem;
  position: fixed;
  top: 0;
  margin: 0;
  padding: 1rem;
  background-color: white;
}

li {
  cursor: pointer;
}
li a {
  text-decoration: none;
  color: inherit;
}
li.active {
  font-weight: bold;
  color: orangered;
}
```



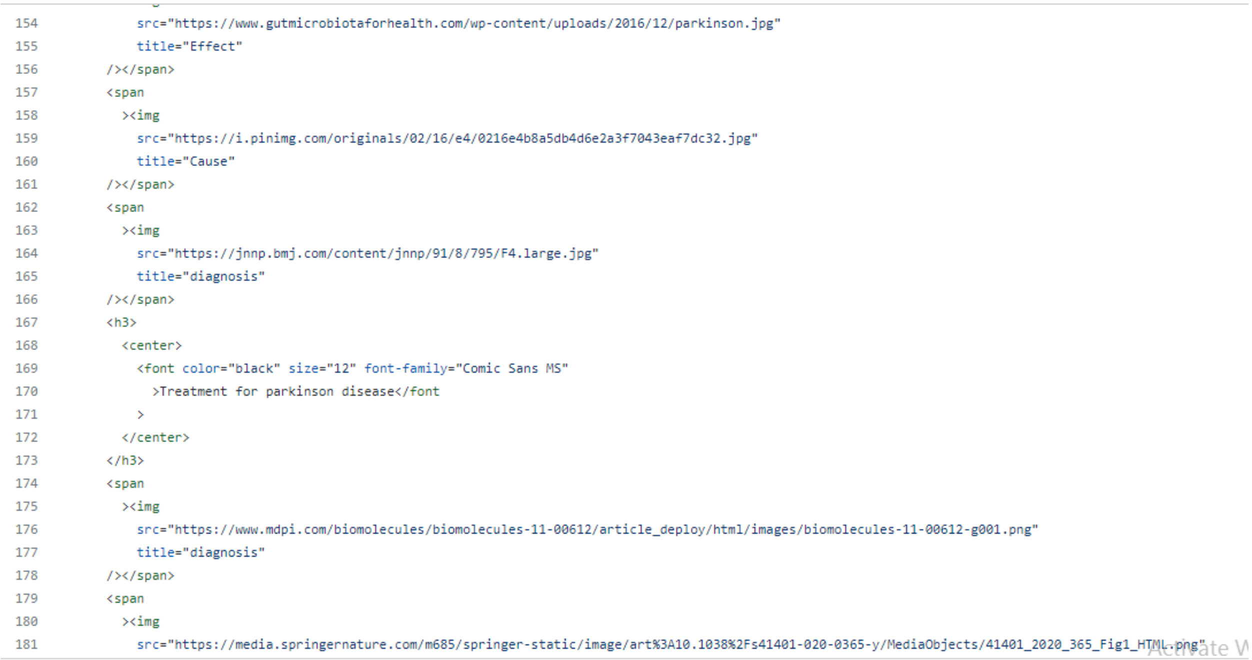
```
    img {
      width: 450px;
      height: 400px;
      padding: 25px;
    }
    img:hover {
      border-color: grey;
    }
    #im {
      width: 1450px;
      height: 700px;
      padding: 25px;
    }
  </style>
</head>
<body>
  <nav>
    <ul>
      <li class="active"><a href="/home">Home</a></li>
      <li class="active"><a href="/upload">Predict-Results</a></li>
    </ul>
  </nav>
  <br /><br /><br />
  <h1>
    <center>
      <b class="pd"
        ><font color="black" size="15" font-family="Comic Sans MS"
          >Detection of Parkinson's Disease using ML</font
```



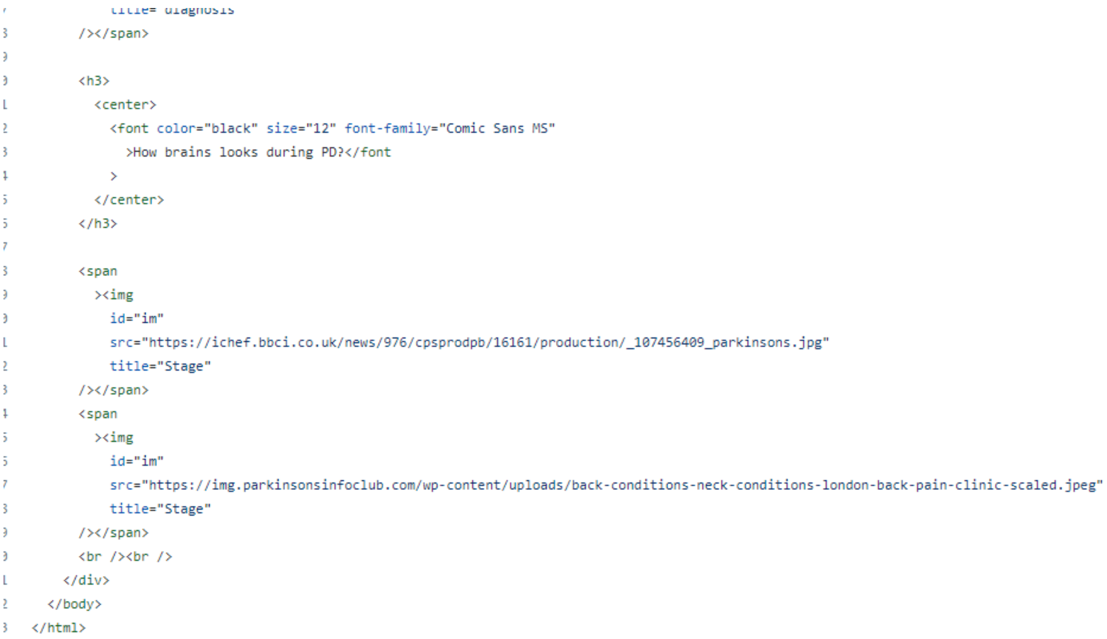
```
          >Detection of Parkinson's Disease using ML</font
        ></b
      >
    </center>
  </h1>
  <div>
    <center>
      <p style="text-align: left">
        Parkinson disease (PD) is a progressive neuro degenerative disorder
        that impacts more than 6 million people around the world. Parkinson's
        disease is non-communicable, early-stage detection of Parkinson's can
        prevent further damages in humans suffering from it.
        However,Nonetheless, non-specialist physicians still do not have a
        definitive test for PD, similarly in the early stage of the diseased
        person where the signs may be intermittent and badly characterized. It
        resulted in a high rate of misdiagnosis (up to 25% among
        non-specialists) and many years before treatment, patients can have
        the disorder. A more accurate, unbiased means of early detection is
        required, preferably one that individuals can use in their home
        setting.However, it has been observed that PD's presence in a human is
        related to its hand-writing as well as hand-drawn subjects. From that
        perspective, several techniques have been proposed by researchers to
        detect Parkinson's disease from hand-drawn images of suspected people.
        But the previous methods have their constraints.
      </p>
    </center>
    <h4>
      <center>
        <b class="pd"
```



```
</center>
<h4>
  <center>
    <b class="pd"
      ><font color="black" size="12" font-family="Comic Sans MS"
        >Causes and Symptoms of Parkinson's Disease</font
      >
    </b>
  </center>
</h4>
<span>
  <img
    src="https://www.narayanahealth.org/blog/wp-content/uploads/2015/04/parkinson.png"
    title="Disease"
  />
</span>
<span>
  <img
    src="https://stanfordmedicine25.stanford.edu/the25/parkinsondisease/_jcr_content/main/panel_builder_0/panel_0/panel_builder_0/panel_0/panel_builder/panel_0/image.img
    title="Symptoms"
/></span>
<span
  ><img
    src="https://www.verywellhealth.com/thmb/Aaqo8oM3QDHSNHCt_DlKCNeWoUk=/1500x0/filters:no_upscale():max_bytes(150000):strip_icc()/zhansen-5200700_Finaledit2-3e7eb00f1b
    title="Stages"
/></span>
<span
  ><img
```

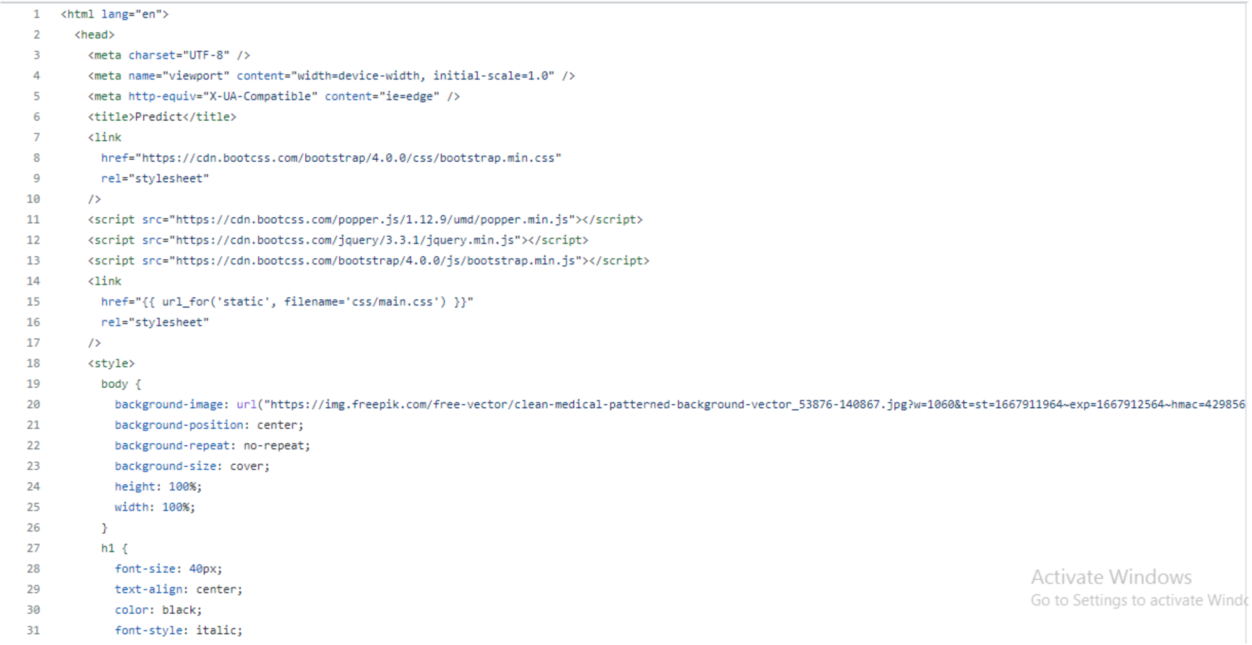


```
    src="https://www.gutmicrobiotaforhealth.com/wp-content/uploads/2016/12/parkinson.jpg"
    title="Effect"
/></span>
<span
  ><img
    src="https://i.pinimg.com/originals/02/16/e4/0216e4b8a5db4d6e2a3f7043eaf7dc32.jpg"
    title="Cause"
/></span>
<span
  ><img
    src="https://jnnp.bmj.com/content/jnnp/91/8/795/F4.large.jpg"
    title="diagnosis"
/></span>
<h3>
  <center>
    <font color="black" size="12" font-family="Comic Sans MS"
      >Treatment for parkinson disease</font
    >
  </center>
</h3>
<span
  ><img
    src="https://www.mdpi.com/biomolecules/biomolecules-11-00612/article_deploy/html/images/biomolecules-11-00612-g001.png"
    title="diagnosis"
/></span>
<span
  ><img
    src="https://media.springernature.com/m685/springer-static/image/art%3A10.1038%2Fs41401-020-0365-y/MediaObjects/41401_2020_365_Fig1_HTML.png"
```

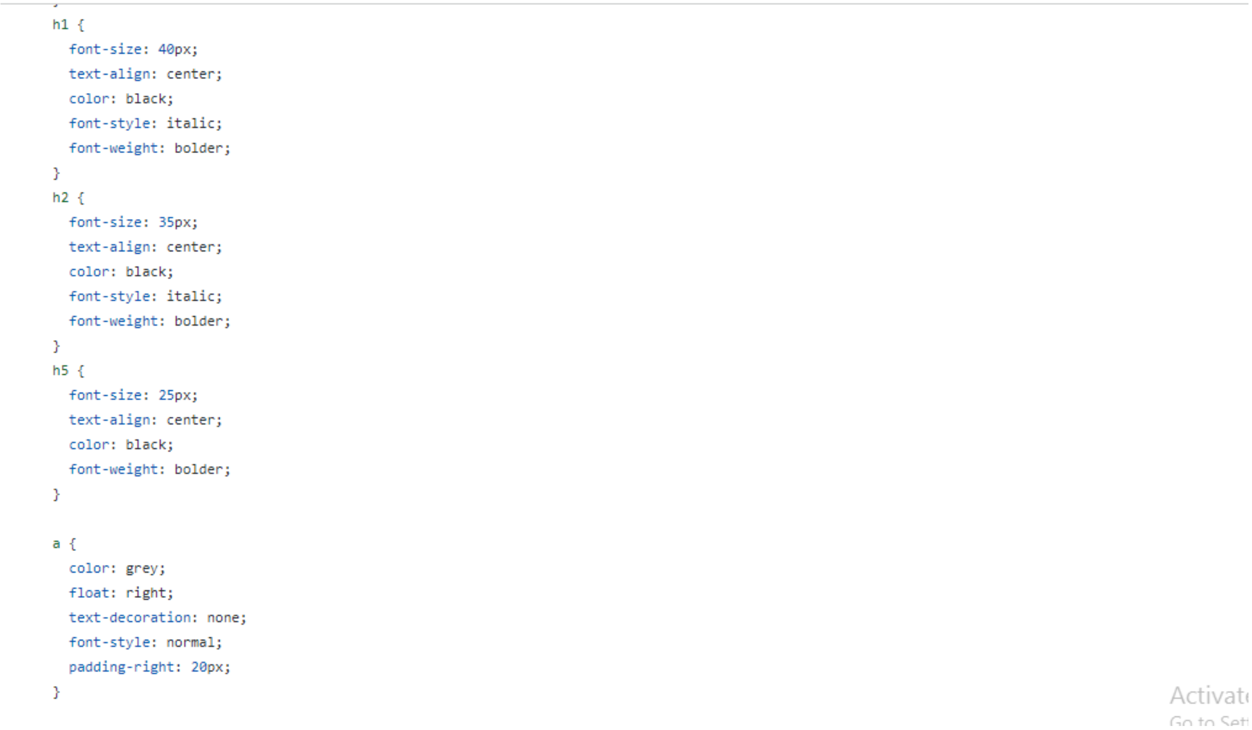


```
          title="diagnosis"
      /></span>

      <h3>
        <center>
          <font color="black" size="12" font-family="Comic Sans MS"
            >How brains looks during PD?</font
          >
        </center>
      </h3>

      <span
        ><img
          id="im"
          src="https://ichef.bbci.co.uk/news/976/cpsprodpb/16161/production/_107456409_parkinsons.jpg"
          title="Stage"
      /></span>
      <span
        ><img
          id="im"
          src="https://img.parkinsonsinfoclub.com/wp-content/uploads/back-conditions-neck-conditions-london-back-pain-clinic-scaled.jpeg"
          title="Stage"
      /></span>
      <br /><br />
    </div>
  </body>
</html>
```

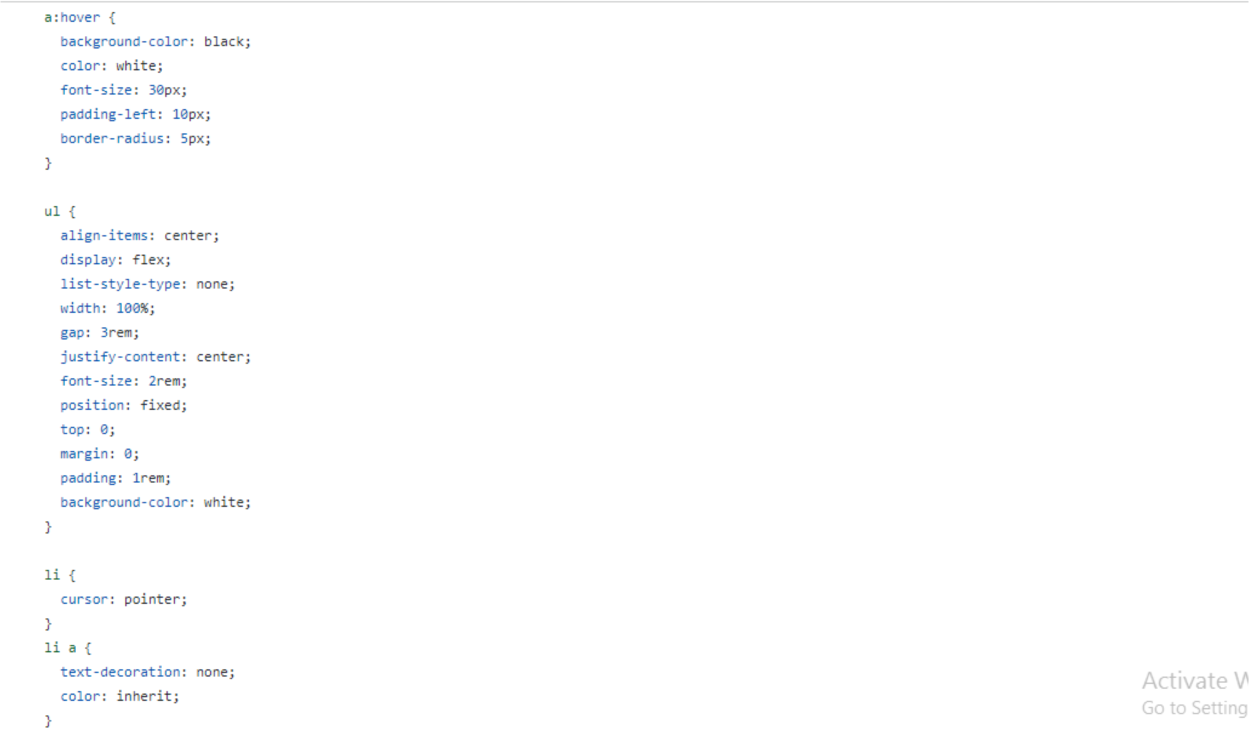
BASE PAGE(HTML)



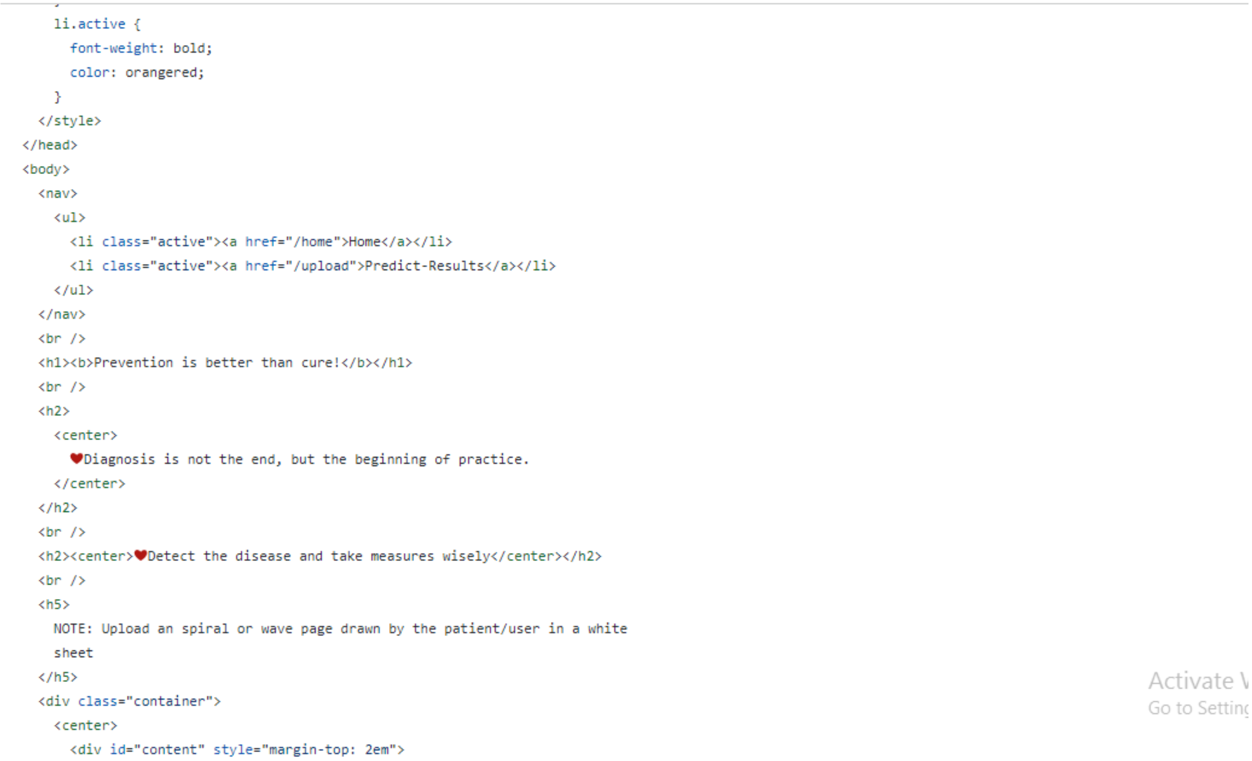
```
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <meta http-equiv="X-UA-Compatible" content="ie=edge" />
    <title>Predict</title>
    <link
      href="https://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.min.css"
      rel="stylesheet"
    />
    <script src="https://cdn.bootcss.com/popper.js/1.12.9/umd/popper.min.js"></script>
    <script src="https://cdn.bootcss.com/jquery/3.3.1/jquery.min.js"></script>
    <script src="https://cdn.bootcss.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
    <link
      href="{{ url_for('static', filename='css/main.css') }}"
      rel="stylesheet"
    />
    <style>
      body {
        background-image: url("https://img.freepik.com/free-vector/clean-medical-patterned-background-vector_53876-140867.jpg?w=1060&t=st=1667911964~exp=1667912564~hmac=429856
        background-position: center;
        background-repeat: no-repeat;
        background-size: cover;
        height: 100%;
        width: 100%;
      }
      h1 {
        font-size: 40px;
        text-align: center;
        color: black;
        font-style: italic;
```



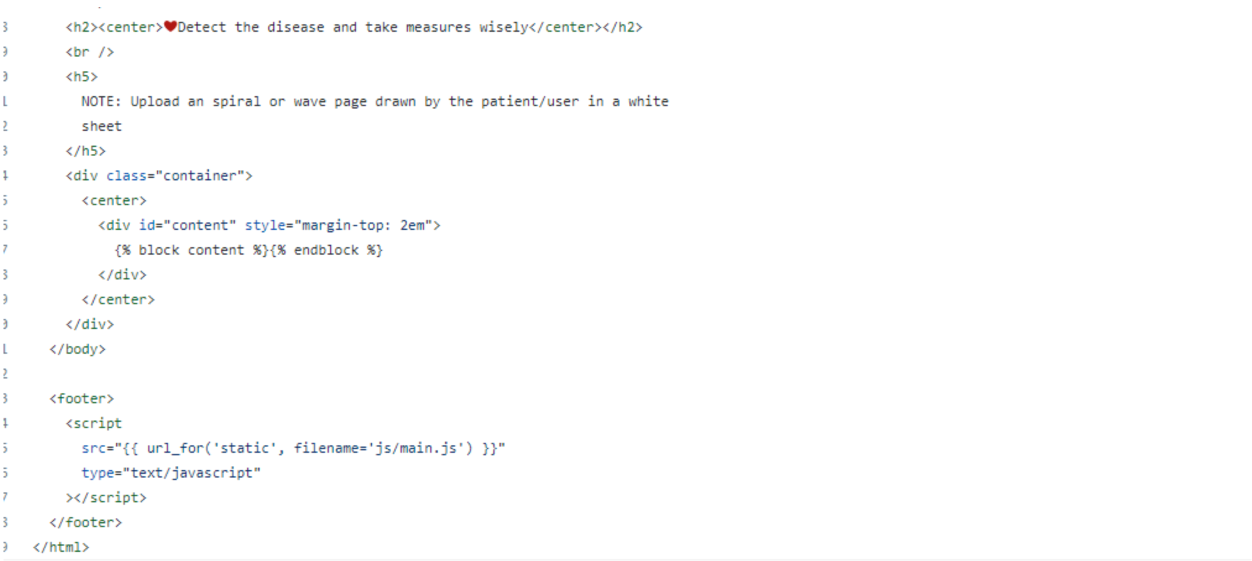
```
h1 {
  font-size: 40px;
  text-align: center;
  color: black;
  font-style: italic;
  font-weight: bolder;
}
h2 {
  font-size: 35px;
  text-align: center;
  color: black;
  font-style: italic;
  font-weight: bolder;
}
h5 {
  font-size: 25px;
  text-align: center;
  color: black;
  font-weight: bolder;
}

a {
  color: grey;
  float: right;
  text-decoration: none;
  font-style: normal;
  padding-right: 20px;
}
```



```
a:hover {
  background-color: black;
  color: white;
  font-size: 30px;
  padding-left: 10px;
  border-radius: 5px;
}

ul {
  align-items: center;
  display: flex;
  list-style-type: none;
  width: 100%;
  gap: 3rem;
  justify-content: center;
  font-size: 2rem;
  position: fixed;
  top: 0;
  margin: 0;
  padding: 1rem;
  background-color: white;
}

li {
  cursor: pointer;
}
li a {
  text-decoration: none;
  color: inherit;
}
```

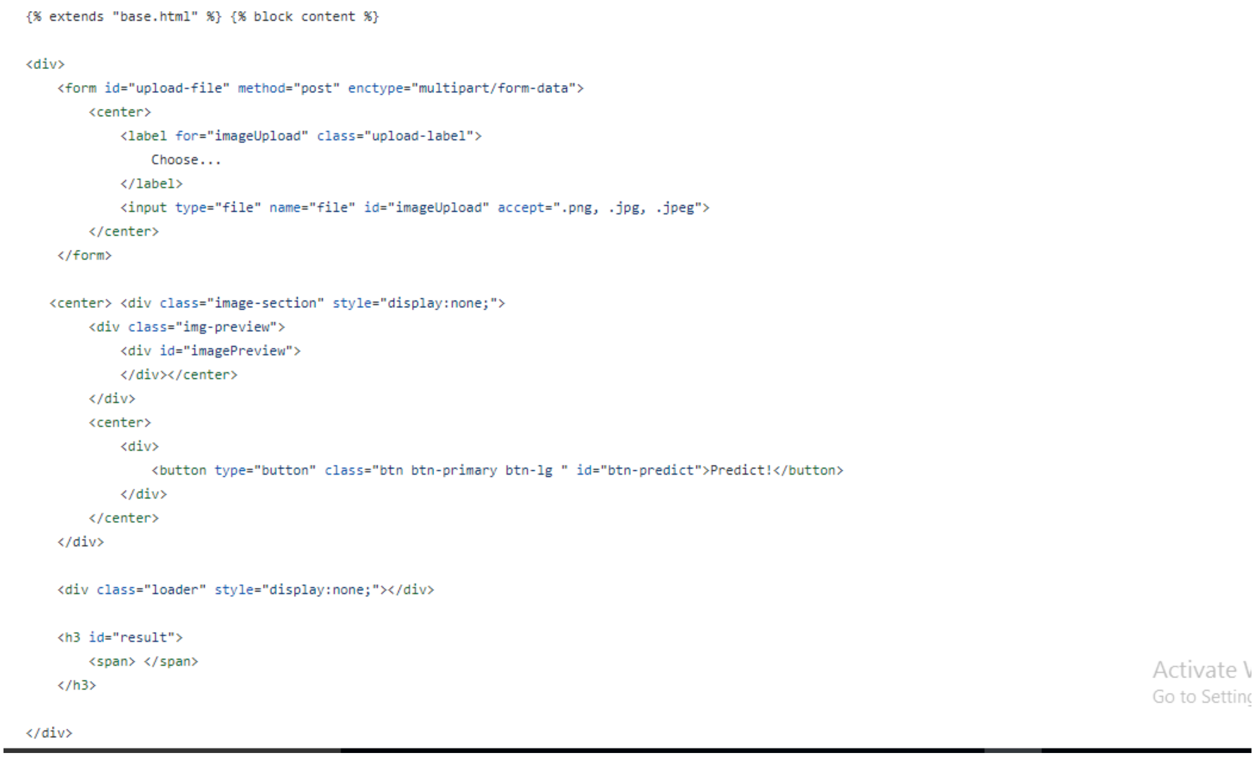


```
    li.active {
      font-weight: bold;
      color: orangered;
    }
  </style>
</head>
<body>
  <nav>
    <ul>
      <li class="active"><a href="/home">Home</a></li>
      <li class="active"><a href="/upload">Predict-Results</a></li>
    </ul>
  </nav>
  <br />
  <h1><b>Prevention is better than cure!</b></h1>
  <br />
  <h2>
    <center>
      ❤Diagnosis is not the end, but the beginning of practice.
    </center>
  </h2>
  <br />
  <h2><center>❤Detect the disease and take measures wisely</center></h2>
  <br />
  <h5>
    NOTE: Upload an spiral or wave page drawn by the patient/user in a white
    sheet
  </h5>
  <div class="container">
    <center>
      <div id="content" style="margin-top: 2em">
```



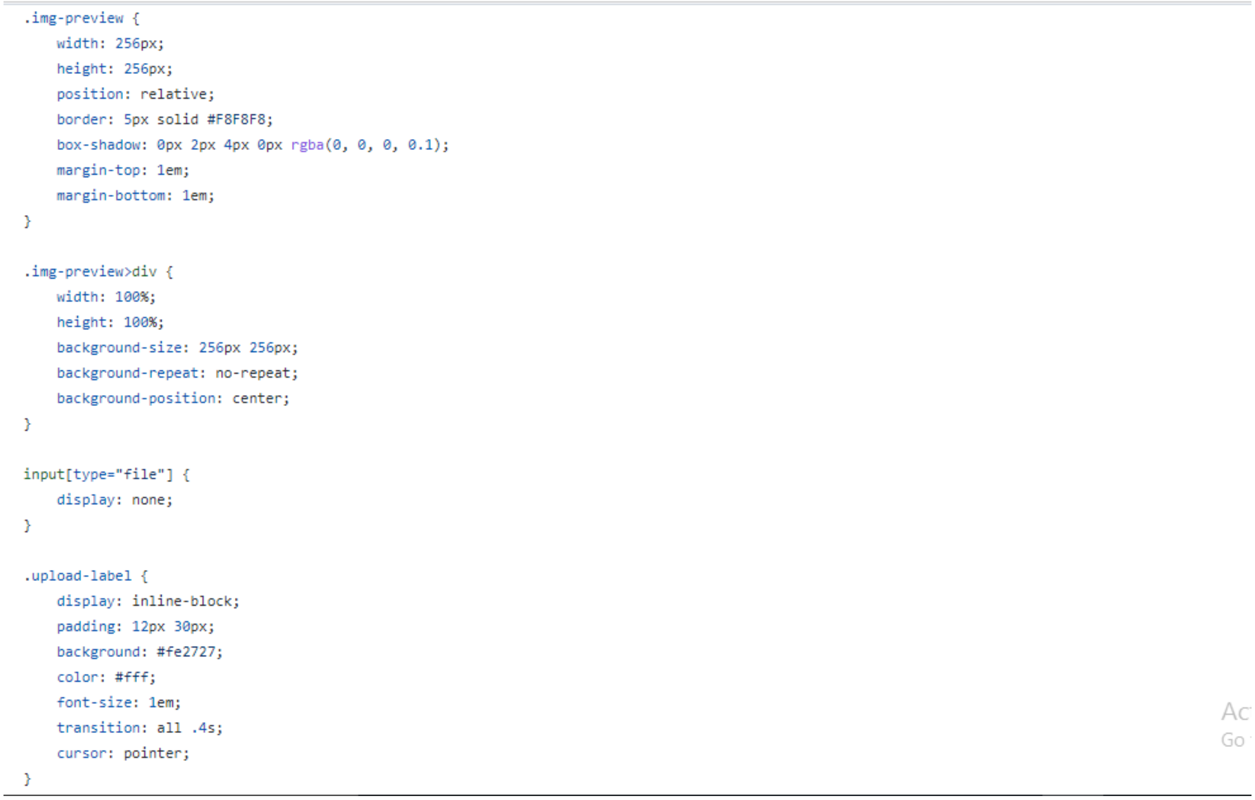
```
    <h2><center>❤Detect the disease and take measures wisely</center></h2>
    <br />
    <h5>
      NOTE: Upload an spiral or wave page drawn by the patient/user in a white
      sheet
    </h5>
    <div class="container">
      <center>
        <div id="content" style="margin-top: 2em">
          {% block content %}{% endblock %}
        </div>
      </center>
    </div>
  </body>

  <footer>
    <script
      src="{{ url_for('static', filename='js/main.js') }}"
      type="text/javascript"
    ></script>
  </footer>
</html>
```

PREDICTION PAGE(HTML)

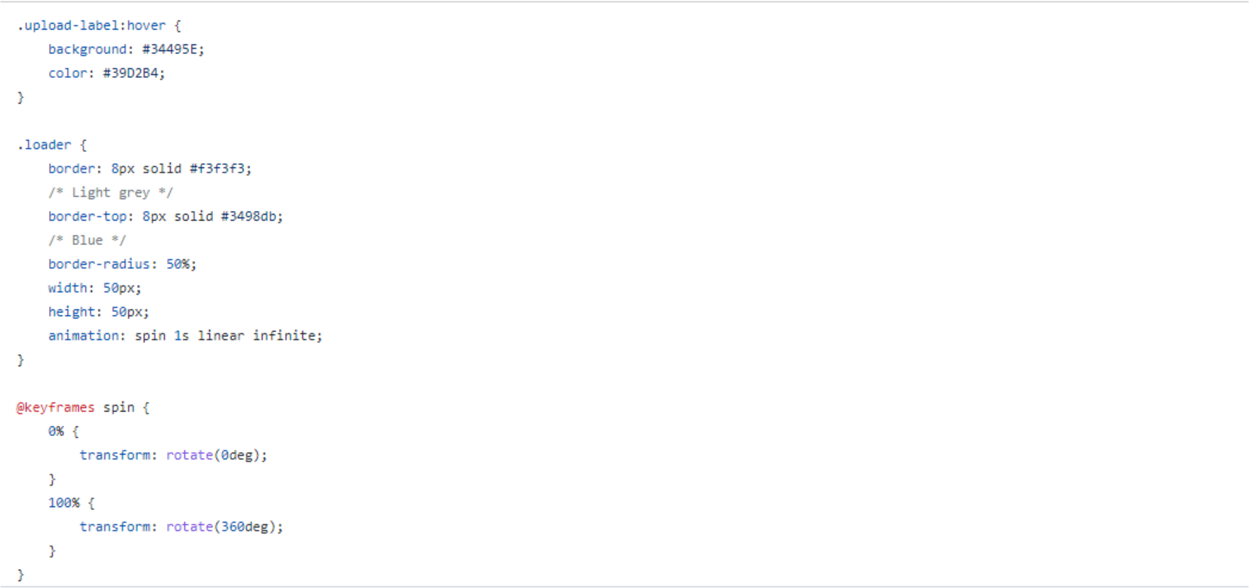


```
{% extends "base.html" %} {% block content %}

<div>
    <form id="upload-file" method="post" enctype="multipart/form-data">
        <center>
            <label for="imageUpload" class="upload-label">
                Choose...
            </label>
            <input type="file" name="file" id="imageUpload" accept=".png, .jpg, .jpeg">
        </center>
    </form>

   <center> <div class="image-section" style="display:none;">
        <div class="img-preview">
            <div id="imagePreview">
            </div></center>
        </div>
        <center>
            <div>
                <button type="button" class="btn btn-primary btn-lg " id="btn-predict">Predict!</button>
            </div>
        </center>
    </div>

    <div class="loader" style="display:none;"></div>

    <h3 id="result">
        <span> </span>
    </h3>

</div>
```

HOME PAGE(CSS)

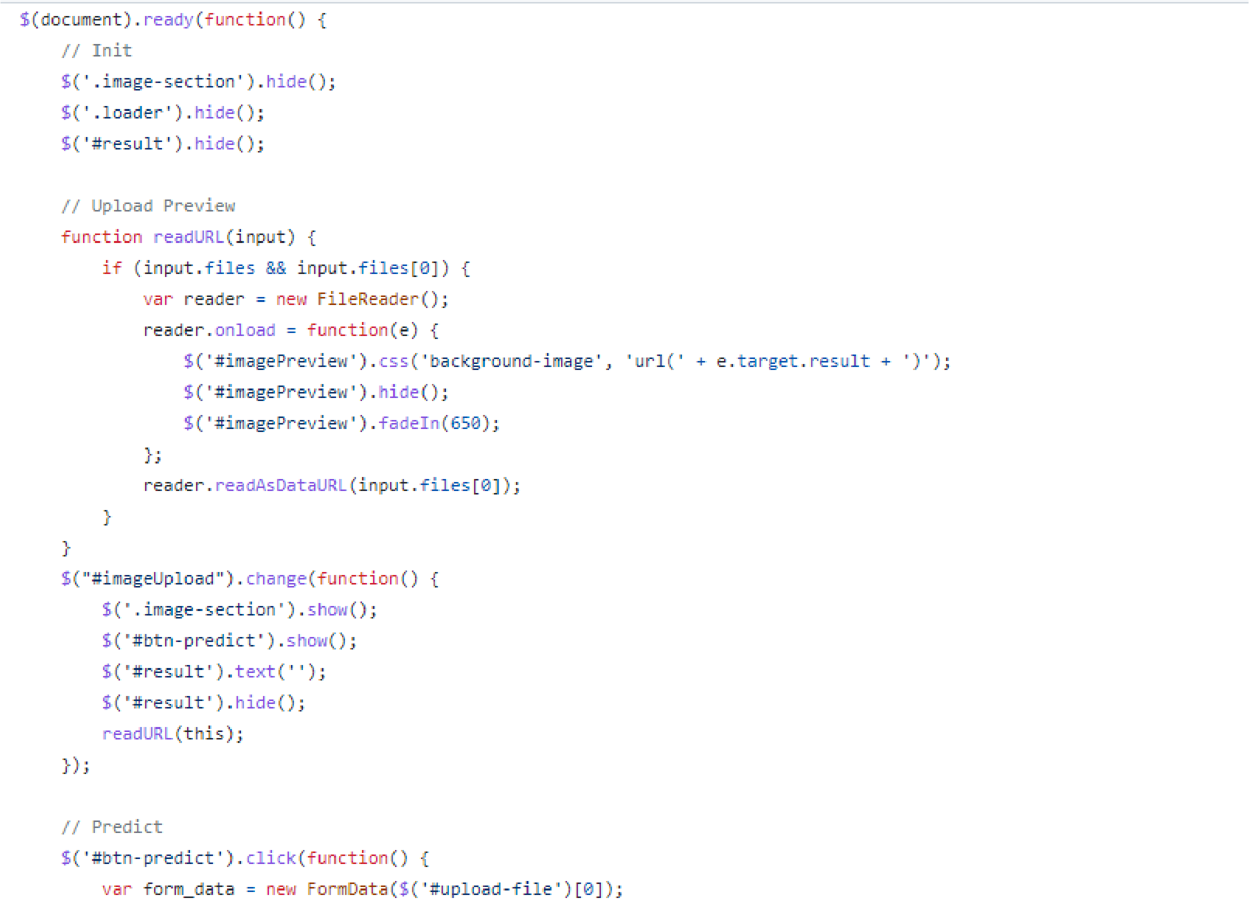


```
.img-preview {
    width: 256px;
    height: 256px;
    position: relative;
    border: 5px solid #F8F8F8;
    box-shadow: 0px 2px 4px 0px rgba(0, 0, 0, 0.1);
    margin-top: 1em;
    margin-bottom: 1em;
}

.img-preview>div {
    width: 100%;
    height: 100%;
    background-size: 256px 256px;
    background-repeat: no-repeat;
    background-position: center;
}

input[type="file"] {
    display: none;
}

.upload-label {
    display: inline-block;
    padding: 12px 30px;
    background: #fe2727;
    color: #fff;
    font-size: 1em;
    transition: all .4s;
    cursor: pointer;
}
```

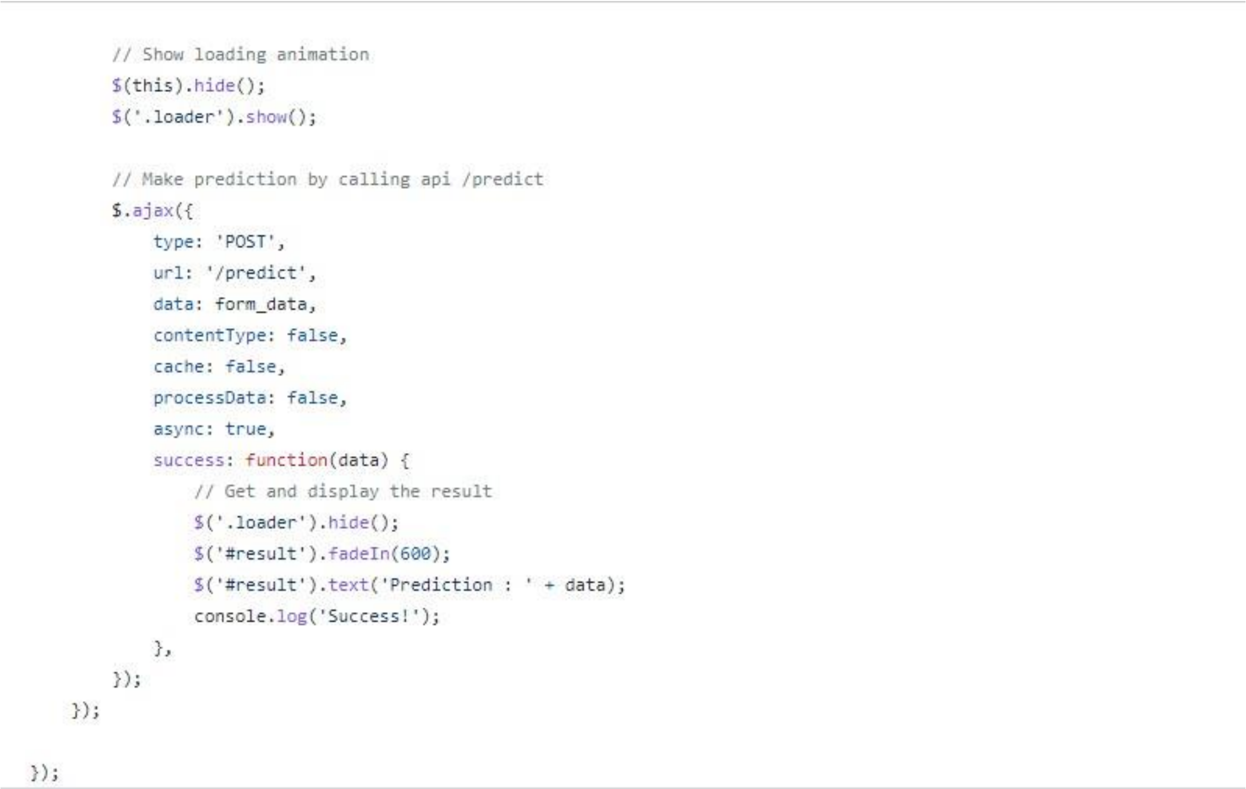


```
.upload-label:hover {
    background: #34495E;
    color: #39D2B4;
}

.loader {
    border: 8px solid #f3f3f3;
    /* Light grey */
    border-top: 8px solid #3498db;
    /* Blue */
    border-radius: 50%;
    width: 50px;
    height: 50px;
    animation: spin 1s linear infinite;
}

@keyframes spin {
    0% {
        transform: rotate(0deg);
    }
    100% {
        transform: rotate(360deg);
    }
}
```

HOME PAGE(JS)



```
$(document).ready(function() {
    // Init
    $('.image-section').hide();
    $('.loader').hide();
    $('#result').hide();

    // Upload Preview
    function readURL(input) {
        if (input.files && input.files[0]) {
            var reader = new FileReader();
            reader.onload = function(e) {
                $('#imagePreview').css('background-image', 'url(' + e.target.result + ')');
                $('#imagePreview').hide();
                $('#imagePreview').fadeIn(650);
            };
            reader.readAsDataURL(input.files[0]);
        }
    }
    $("#imageUpload").change(function() {
        $('.image-section').show();
        $('#btn-predict').show();
        $('#result').text('');
        $('#result').hide();
        readURL(this);
    });

    // Predict
    $('#btn-predict').click(function() {
        var form_data = new FormData($('#upload-file')[0]);
```



```
            // Show loading animation
            $(this).hide();
            $('.loader').show();

            // Make prediction by calling api /predict
            $.ajax({
                type: 'POST',
                url: '/predict',
                data: form_data,
                contentType: false,
                cache: false,
                processData: false,
                async: true,
                success: function(data) {
                    // Get and display the result
                    $('.loader').hide();
                    $('#result').fadeIn(600);
                    $('#result').text('Prediction : ' + data);
                    console.log('Success!');
                },
            });
        });

    });
```

GITHUB

https://github.com/IBM-EPBL/IBM-Project-21667-1659787516.

PROJECT DEMO LINK

https://drive.google.com/drive/folders/1PaUZlLN3936fErHy4NUXODHT6owWMSN2?usp=sharing.