SARANATHAN COLLEGE OF ENGINEERING

Venkateshwara Nagar, Panjappur, Trichy-12

NALAIYA THIRAN

Project Title	:	Detection of Parkinson’s disease using machine learning
Domain	:	Applied Data Science
Team id	:	PNT2022TMID32832
Team Leader	:	

Aswini Devi B - 813819205010@smartinternz.com

Team Members :

Harini S - 813819205018@smartinternz.com

Nishanthini S - 813819205043@smartinternz.com

Samyuktha C - 813819205052@smartinternz.com

GitHub : [Project Repository]

TABLE OF CONTENTS

TABLE OF CONTENTS

1. INTRODUCTION

The recent report of the World Health Organization shows a visible increase in the number and health burden of Parkinson's disease patient's increases rapidly. In China, this disease is spreading so fast and estimated that it reaches half of the population in the next 10 years. Classification algorithms are mainly used in the medical field for classifying data into different categories according to the number of characteristics. Parkinson's disease is the second most dangerous neurological disorder that can lead to shaking, shivering, stiffness, and difficulty walking and balance. It caused mainly due by the breaking down of cells in the nervous system. Parkinson's can have both motor and non-motor symptoms. The motor symptoms include slowness of movement, rigidity, balance problems, and tremors. If this disease continues, the patients may have difficulty walking and talking. The non-motor symptoms include anxiety, breathing problems, depression, loss of smell, and change in speech. If the above-mentioned symptoms are present in the person then the details are stored in the records. In this paper, the author considers the speech features of the patient, and this data is used for predicting whether the patient has Parkinson's disease or not.

1.1 PROJECT OVERVIEW

The researchers found that the drawing speed was slower and the pen pressure is lower among Parkinson's patients. One of the indications of Parkinson's is tremors and rigidity in the muscles, making it difficult to draw smooth spirals and waves.

It is possible to detect Parkinson's disease using the drawings alone instead of measuring the speed and pressure of the pen on paper. Our goal is to quantify the visual appearance (using HOG method) of these drawings and then train a machine learning model to classify them. In this project, We are using, Histogram of Oriented Gradients (HOG) image descriptor along with a Random Forest classifier to automatically detect Parkinson's disease in hand-drawn images of spirals and waves.

1.2 PURPOSE

Many of the people aged 65 or more do have a neurodegenerative disease, which has no cure. If we detect the disease in the early stages, then we can control it. Almost 30% of the patients are facing this incurable disease. Current treatment is available for patients who have minor symptoms. If these symptoms cannot be found at the early stages, it leads to death. The main cause for Parkinson's disease is the accumulation of protein molecules in the neuron which gets misfolded and hence causing Parkinson's disease. So till now, researchers got the symptoms and the root causes i.e. from where this disease had evolved. But very few symptoms have come to their cure and there are many symptoms that have no solution. So in this era where Parkinson's disease is increasing, it is very important to find the solution which can predict it in its early stages.

2. LITERATURE SURVEY

2.1 Existing problem

S.NO	TITLE	AUTHOR AND YEAR OF PUBLICATIONS	METHODOLOGY USED	LIMITATIONS
1.	Classification of handwritten drawings of people with Parkinson's disease by using histograms of oriented gradients and the random forest classifier	João Paulo Folador et al, 2 October 2019	The methods used are Random Forest Classifier and Histogram of Gradients (HOG) for achieving the accuracy of the model.	Training the model with more number of data sets rather than the existing ones (51) must be used for more specificity and accuracy
2.	Early Detection of Parkinson's Disease using Contrast Enhancement Techniques and CNN	Ishan Vatsaraj et al, 5 May 2021	Augmentation methods like rotation, vertical and horizontal flipping along with Support vector machines and HOG methods are applied.	This model does not provide the probability of the percentage that is affected in a person by the Parkinson's disease.
3.	Detection of Parkinson's disease using machine learning algorithm.	Shikha Singh et al, 22 April 2022	In this paper, they have used the machine learning ensemble method XGBoost which gives accurate results.	The research can be expanded by utilizing additional models and comparing the results to establish the most optimized and efficient models for disease detection and determining the degree of disease in the patient.

2.2 Reference

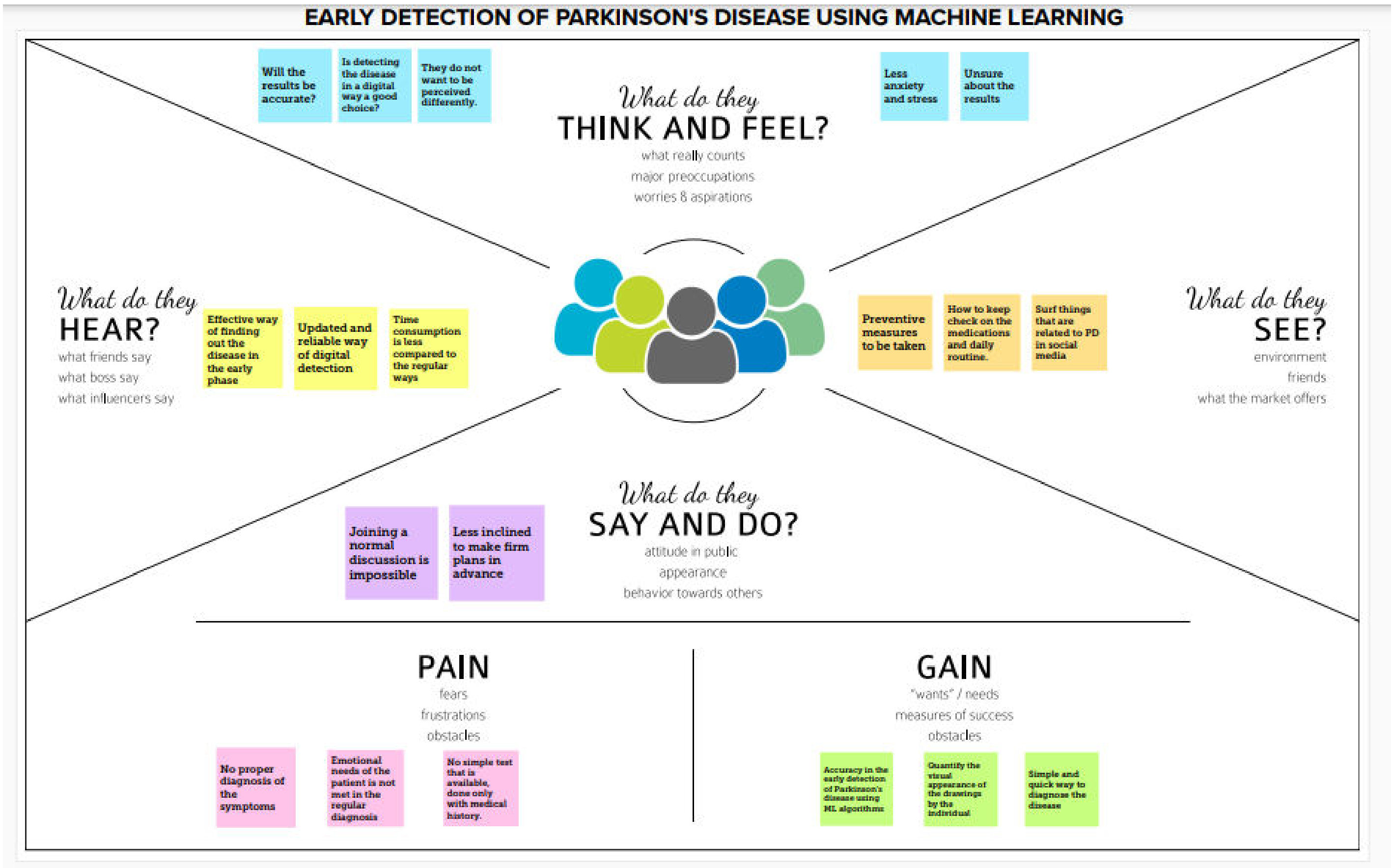
- João Paulo Folador et al, 2 October 2019
- Ishan Vatsaraj et al, 5 May 2021
- Shikha Singh et al, 22 April 2022

2.3 Problem Statement Definition

After researching and getting to know about the various approaches like logistic regression, XGBoost classifier that have been devised for the Parkinson's disease prediction using machine learning we have decided to propose our problem statement as prediction of Parkinson's disease using random forest classifier to accurately find out the predicted results on whether a person is healthy or not using Histogram of Gradients descriptor to automatically detect Parkinson's disease.

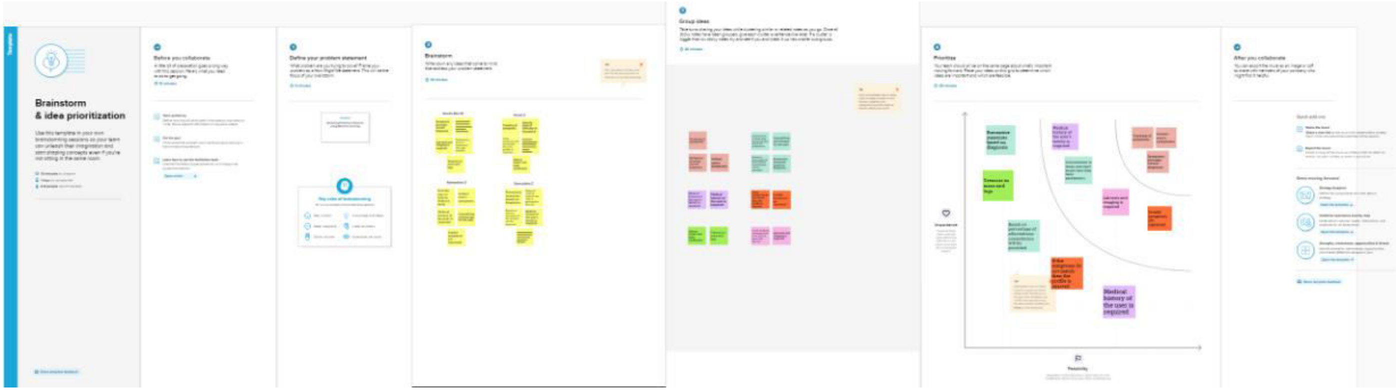
3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming :

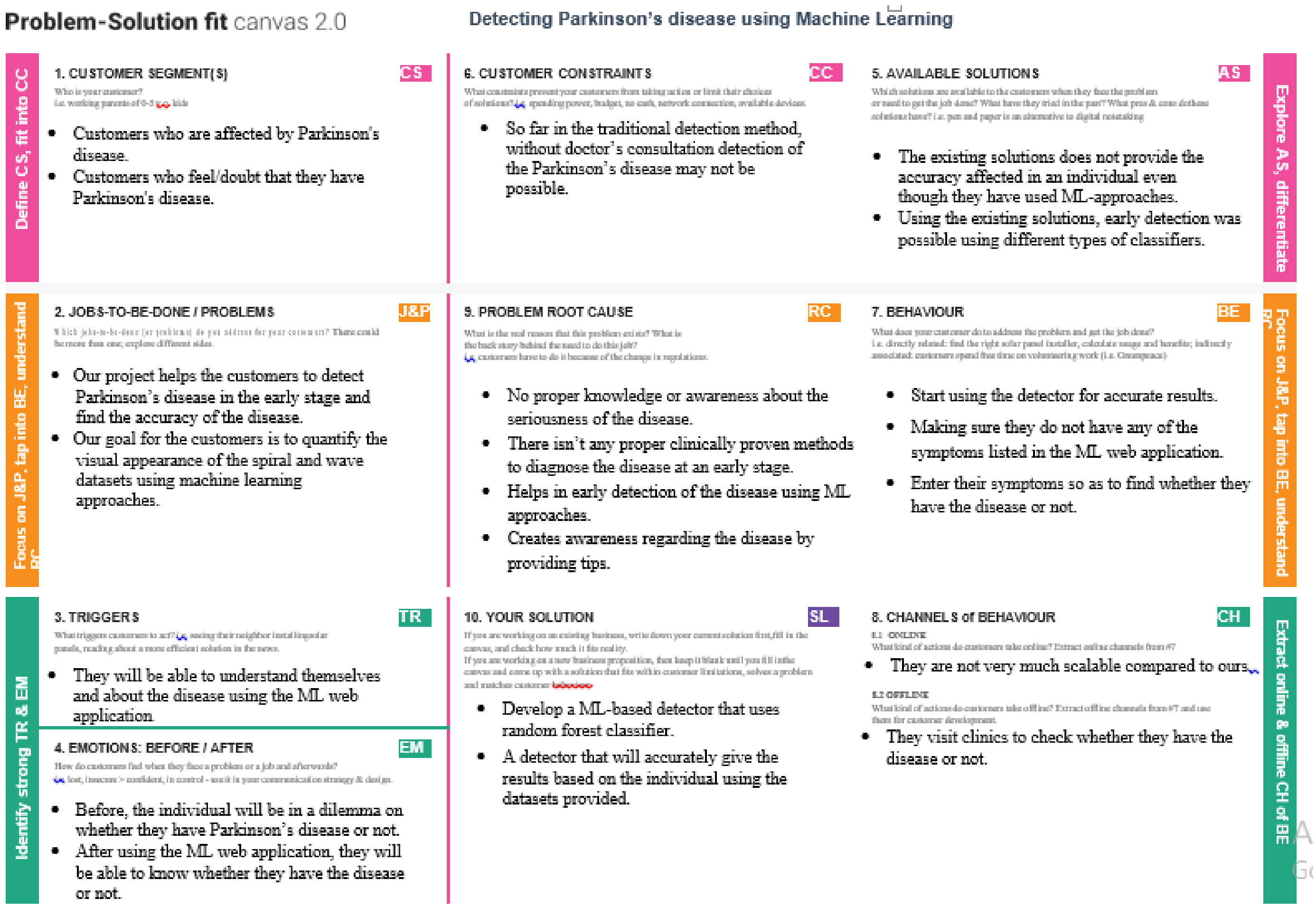
After researching and getting to know about the various approaches like logistic regression, XGBoost classifier that have been devised for the Parkinson's disease prediction using machine learning we have decided to propose our problem statement as prediction of Parkinson's disease using random forest classifier to accurately find out the predicted results on whether a person is healthy or not using Histogram of Gradients descriptor to automatically detect Parkinson's disease.



3.3 Proposed Solution

S.No	Parameter	Description
1.	Problem Statement (Problem to be solved)	Detection of Parkinson’s disease using the drawings made by the subjects instead of measuring the speed and pressure of the pen on paper to classify them. Our goal is to quantify the visual appearance of these drawings.
2.	Idea / Solution description	ML based application for detection of the symptoms of Parkinson’s disease at the early stage and to train a machine learning model to differentiate between healthy and the affected.
3.	Novelty / Uniqueness	By using random forest classifier and histogram of gradients we will be able to see the accuracy of the model which will automatically predict Parkinson’s disease.
4.	Social Impact / Customer Satisfaction	By using machine learning approaches, we may therefore identify relevant features that are not traditionally used in the clinical diagnosis of Parkinson’s disease and rely on these alternative measures to detect the disease during the early stage.
5.	Business Model (Revenue Model)	
6.	Scalability of the Solution	By using HOG and Random forest classifier we will be able to find a better accuracy of the model compared to existing ones as it predicts only if the individual has Parkinson’s disease or not which makes the project scalable. The project maybe more scalable if deep learning methods are implemented as it may give more accurate results.

3.4 Problem Solution Fit



4. REQUIREMENT ANALYSIS

4.1 Functional requirements

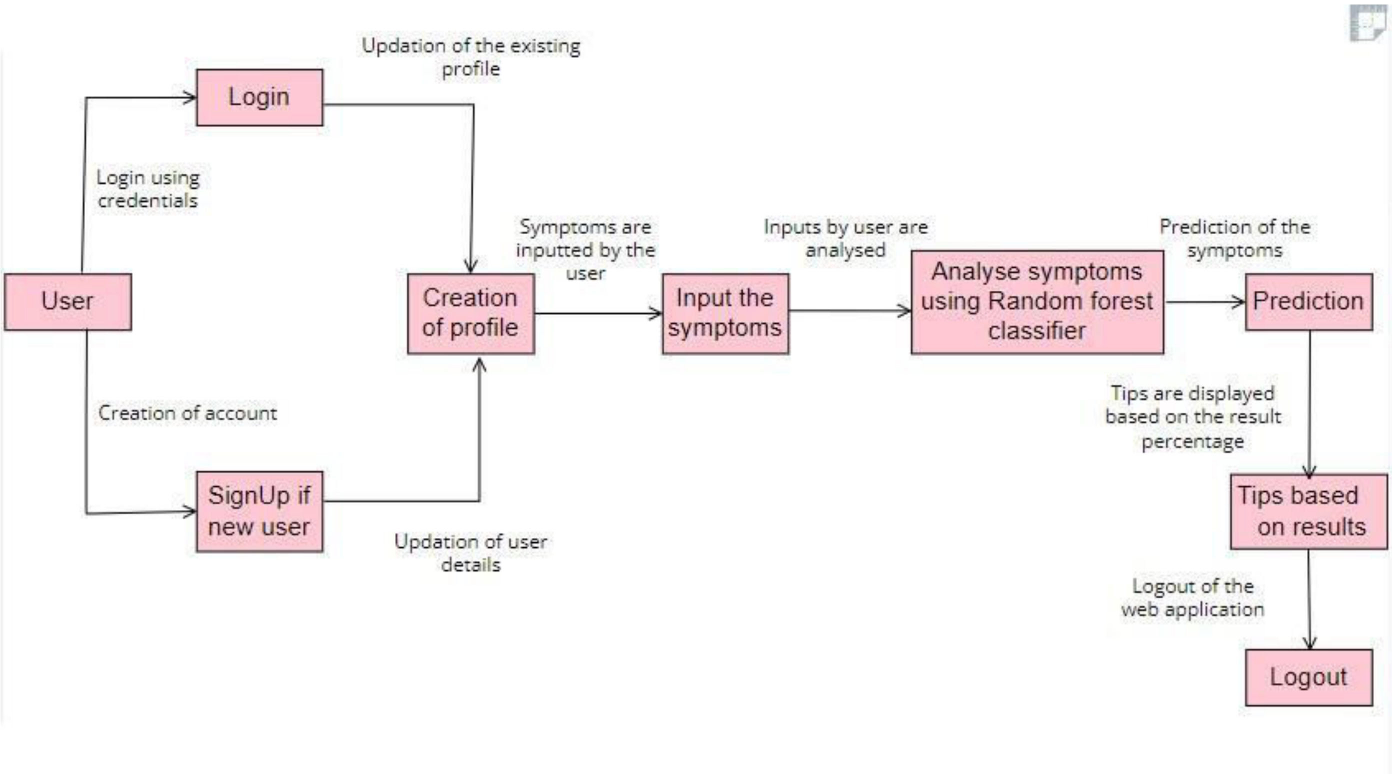
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Home Page	New users have to Register. Registered user login to access.
FR-2	User Registration	Registration through Gmail
FR-3	Login Page	User's enter their username and password
FR-4	Test Inputs	The user inputs the symptoms into the Machine Learning model.
FR-5	Result	Accurately, get the result as positive or negative with percentage affected in a person by the Parkinson's Disease.

4.2 NON-FUNCTIONAL REQUIREMENT

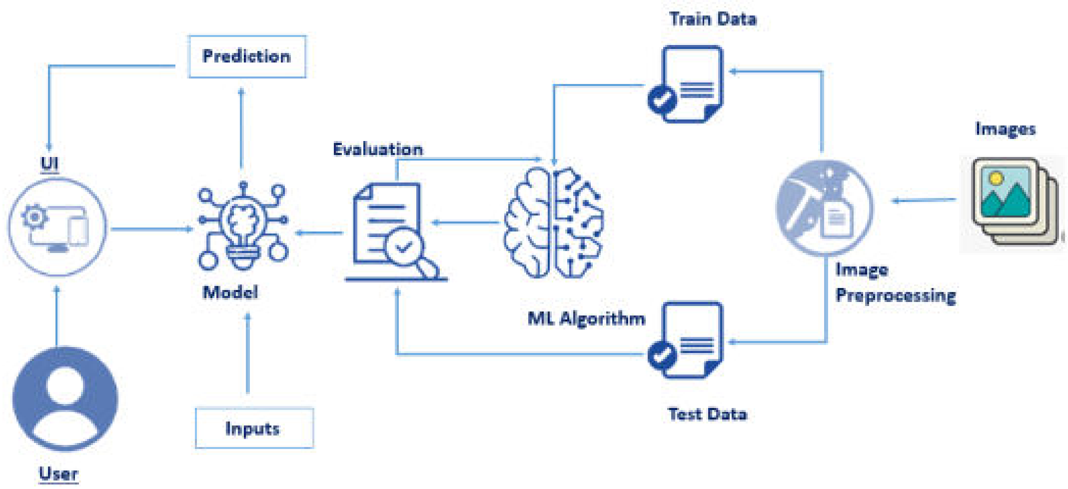
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The users who have signed up for the web application will have access to all the resources present in that website (for e.g.: tips to overcome the disease at an early stage).
NFR-2	**Security**	User information is protected for authenticated users.
NFR-3	**Reliability**	Since only authorized users have access to the contents of the page, the web application is reliable and authorized.
NFR-4	**Performance**	The web application makes use of HOG for image classification to quantify the image hence it gives accurate results.
NFR-5	**Availability**	The web application can be accessed 24/7 from anywhere when connected to the internet.
NFR-6	**Scalability**	The trained ML model can provide accurate results whenever the size of the dataset and the number of users are extended.

5. PROJECT DESIGN

5.1 DATA FLOW DIAGRAM



5.2 SOLUTION & TECHNICAL ARCHITECTURE



5.3 USER STORIES

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Webuser)	Home Page	USN-1	Description about the Parkinson's disease.	I can get an idea about the disease.	Low	Sprint-3
		USN-2	Details about the symptoms of the user is required. .		Low	Sprint-3
	Registration	USN-3	As a user, I can register to the web application by entering my username, email, and password.	I can access my account in a secured way .	Moderate	Sprint-3
	Login	USN-4	As a user, I can log in to the web application by entering my email id & password.	I can login successfully as I am authorised.	High	Sprint-2
	Main Page(Test vitals)	USN-5	As a user, I submit the symptoms required for the prediction.	I can access the the web application and can submit the inputs required.	Moderate	Sprint-4
	Results	USN-6	Results will be displayed along with their accuracyof the model.	I get my results accurately.	High	Sprint-4
Admin	Data collection	USN-7	Collect the required data for the detection of Parkinson's disease		High	Sprint-1

6. PROJECT PLANNING & SCHEDULING

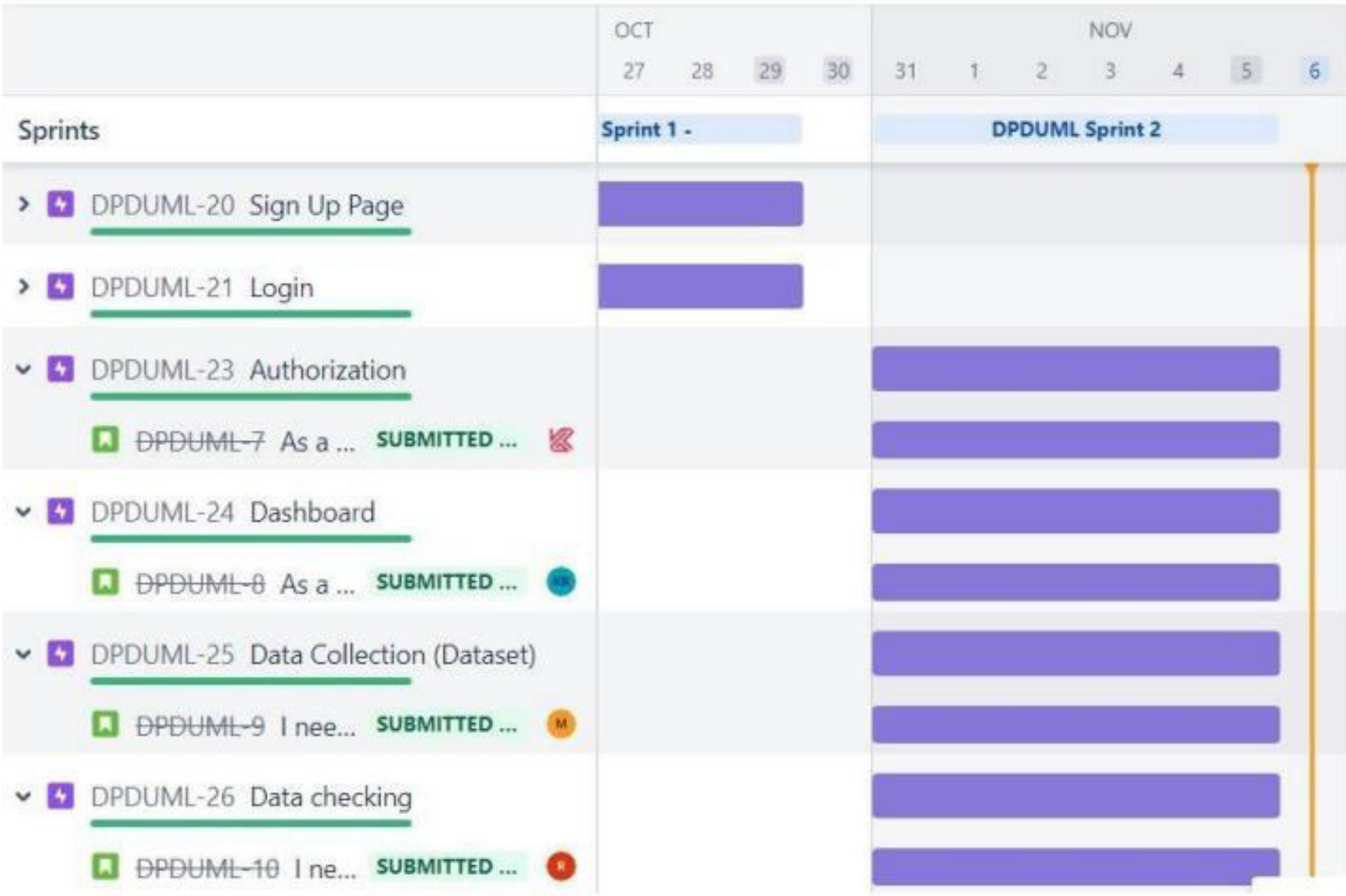
6.1 SPRINT PLANNING ESTIMATION

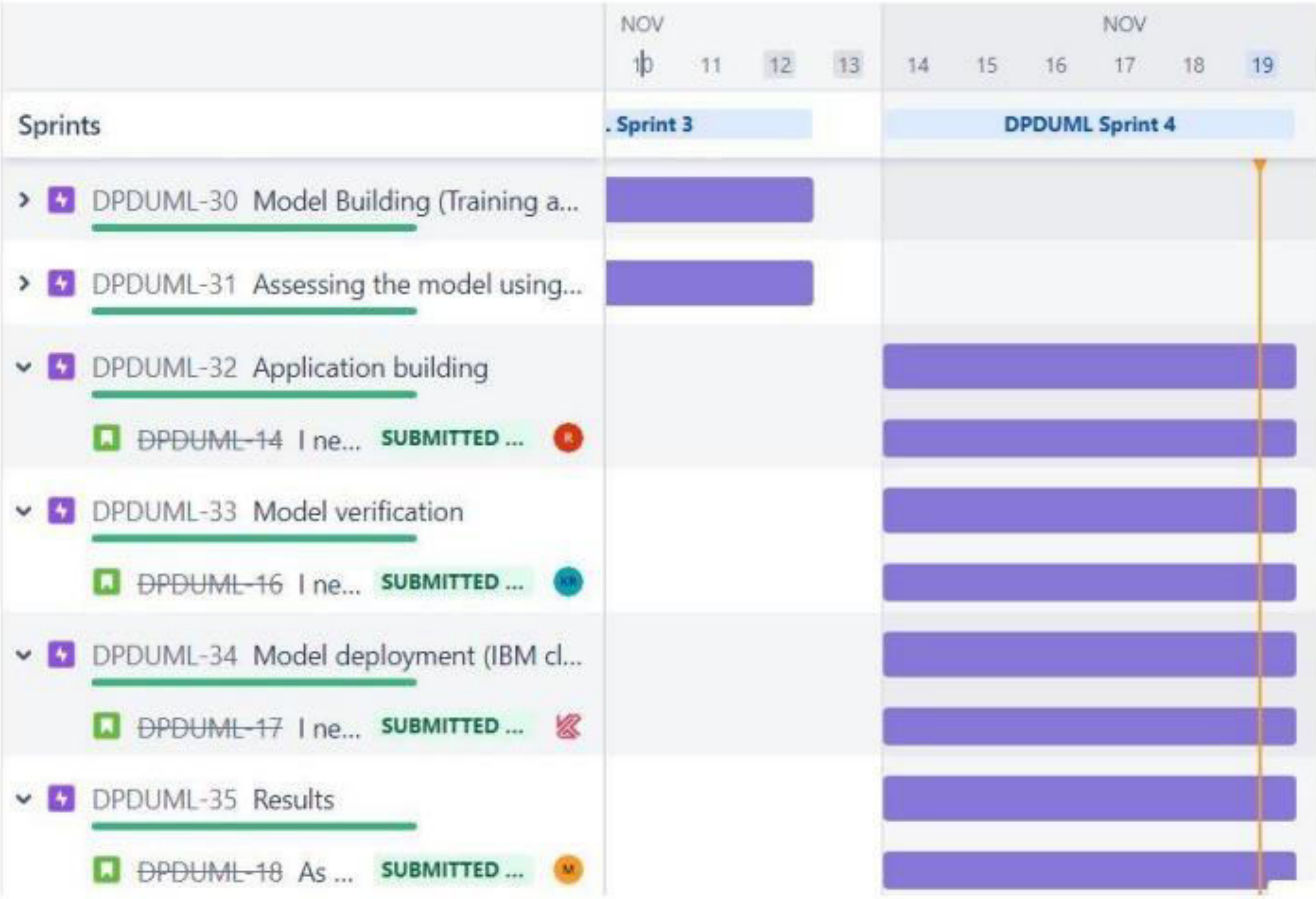
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-3	Home Page	USN-1	Description about the Parkinson's disease.	2	Low	Harini S
Sprint-3		USN-2	Details about the symptoms of the user is required.	1	Low	Samyuktha C
Sprint-3	Registration	USN-3	As a user, I can register to the web application by entering my username, email, , and password.	2	Moderate	Harini S
Sprint-2	Login	USN-4	As a user, I can log in to the web application by entering my username & password.	2	High	Samyuktha C
Sprint-4	Main Page(Test vitals)	USN-5	As a user, I submit the symptoms and the medical history required for the prediction.	2	Moderate	Aswini Devi B
Sprint-4	Results	USN-6	Results will be displayed whether the user is affected or not..	3	High	Nishanthini S
Sprint-1	Data collection	USN-7	Collect the required data for the detection of Parkinson's disease	1	High	Aswini Devi B
Sprint-1	Model Building	USN-8	Build the model using a Random forest classifier and HOG to classify the images.	2	High	Samyuktha C
Sprint-2	Train the model	USN-9	Training of ML model using IBM Watson.	2	High	Harini S
Sprint-2	Integrate the web app with the IBM model	USN-10	Usage of flask for the integration purpose	2	Moderate	Nishanthini S

6.2 Sprint Delivery Schedule

SPRINT	TOTAL STORY POINTS	DURATION	SPRINT START DATE	SPRINT END DATE (PLANNED)	STORY POINTS COMPLETED (AS ON PLANNED END DATE)	SPRINT RELEASE DATE (ACTUAL)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	19 Nov 2022

6.3 Reports From Jira





7. CODING & SOLUTIONING

7.1 INDEX.HTML

```
<html>
   <head>
      <title>PARKINSON'S DISEASE </title>
      <link rel = "stylesheet" href="{{url_for('static',filename='css/style.css')}}">
   </head>
   <body>
      <div class="hero">
         <div class="form-box">
            <div class="button-box">
               <div id="btn"></div>
               <button type="button" class="toggle-btn" onclick="login()">Log In</button>
               <button type="button" class="toggle-btn" onclick="register()">Register</button>
```

```
        </div>
        <form id="login" class="input-group" action="/form_login" method="post">
            <input type="text" class="input-field" placeholder="User Id" name ="userid" required>
            <input type="password" class="input-field" placeholder="Password" name="pwd" required>
            <button type="submit" class="submit-btn" value="Login">Login</button>
        </form>
        <h6 class="err">{{info}}</h6>
        <form id="register" class="input-group" action="/form_reg" method="post">
            <input type="email" class="input-field" placeholder="Email Id">
            <input type="text" class="input-field" placeholder="User Id" name ="userid" required>
            <input type="password" class="input-field" placeholder="Password" name="pwd" required>
            <button type="submit" id = "sub" class="submit-btn" >Register</button>
        </form>
        <h6 class="err">{{info}}</h6>
    </div>
</div>
<script>
    var x = document.getElementById("login")
    var y = document.getElementById("register")
    var z = document.getElementById("btn")
    function register(){
        x.style.left = "-400px";
        y.style.left = "50px";
        z.style.left = "110px";
    }
    function login(){
        x.style.left = "50px";
```

```
            y.style.left = "450px";
            z.style.left = "0px";
          }
      </script>
   </body>
</html>
```

7.2 HOME.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{
            bgcolor="#800000";
            position: relative;
            background-size: cover;
            background-repeat: no-repeat;
            height: 100%;
            width: 100%;
         }
         h3{
            text-align:center;
            color:white;
         }
```

```
.main{
   margin-top:100px;
}
p{
   color:green;
   text-indent:10px;
   margin:10px;
   font-size:20px;
}
.navbar{
margin: 0px;
padding:20px;
background-color:#eeebdd;
opacity:0.6;
color:black;
font-family:'Roboto',sans-serif;
font-style: italic;
border-radius:20px;
font-size:25px;
}
a{
color:rgb(11, 3, 21);
float:right;
text-decoration:none;
font-style:normal;
padding-right:20px;
}
a:hover{
```

```
        background-color:black;
        color:white;
        border-radius:15px;
        font-size:30px;
        padding-left:10px;
        }
        img{
        width:450px;
        height:400px;
        padding:25px;

        }
        img:hover{

        border-color:grey;
        }
        #im{
        width:1450px;
        height:700px;
        padding:25px;
        }
    </style>
</head>
<body bgcolor="#AEEB62">
    <div class="navbar">
        <a href="/logout" >Logout</a>
        <a href="/upload" >Predict</a>
        <a href="/home">Home</a>
```

```
<a>Welcome {{name}} !</a>

<br>

</div>

<br>

<center><b class="pd"><font color="green" size="15" font-family="Open Sans" >Detecting Parkinson's Disease using ML</font></b></center>

<div>

<br>

<br>

<center>

<p>Parkinson's disease is a progressive disorder that affects the nervous system and the
parts of the body controlled by the nerves. Diagnosis of Parkinson's disease (PD) is commonly
based on medical observations and assessment of clinical signs, including the characterization
of a variety of motor symptoms. However, traditional diagnostic approaches may suffer from
subjectivity as they rely on the evaluation of movements that are sometimes subtle to human
eyes and therefore difficult to classify, leading to possible misclassification. In the
meantime, early non-motor symptoms of PD may be mild and can be caused by many other
conditions. Therefore, these symptoms are often overlooked, making diagnosis of PD at an
early stage challenging. To address these difficulties and to refine the diagnosis and
assessment procedures of PD, machine learning methods have been implemented for the
classification of PD and healthy controls or patients with similar clinical presentations.
 Symptoms start slowly.The first symptom may be a barely noticeable tremor in just one hand.
 Tremors are common, but the disorder may also cause stiffness or slowing of movement.</p>

</center>

<span><img
src="https://i.pinimg.com/564x/9d/bb/40/9dbb40520cee5eb80f6358447f1d02e4.jpg"/></span>

<span><img
src="https://i.pinimg.com/564x/ee/fe/a6/eefea6145e3991d708fc18d769a20394.jpg"></span>

<span><img
src="https://i.pinimg.com/564x/5d/4a/77/5d4a778e85780ac4ac950aa4d4aed535.jpg"></span>
```

```
        <span><img
src="https://i.pinimg.com/564x/ff/57/e6/ff57e677da997e1fd6910355083a73fc.jpg"></span>

        <span><img
src="https://i.pinimg.com/564x/18/64/98/18649858f2bf2fc7112fca79d56dde32.jpg"></span>

          <span><img
src="https://i.pinimg.com/564x/4b/4c/6a/4b4c6aa15b1d64e715027340afc01d34.jpg"></span>

      <br><br><br><br><br>

        </div>

    </body>

</html>
```

7.3 BASE. 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>
            .bar
            {
                margin: 0px;
                padding:20px;
                background-color:#e2e2e2;
                opacity:0.6;
```

```
    color:black;
    font-family:'Roboto',sans-serif;
    font-style: italic;
    border-radius:20px;
    font-size:25px;
}
a{
    color:dodgerblue;
    float:right;
    text-decoration:none;
    font-style:normal;
    padding-right:20px;
}
a:hover{
    background-color:darkturqouise;
    color:white;
    border-radius:15px;
    font-size:30px;
    padding-left:5px;
}
body{
    background-color: #e2e2e2;
    position: relative;
    background-size: cover;
    background-repeat: no-repeat;
    height: 100%;
    width: 100%;
}
```

```
        h1{
          font-size:40px;
          text-align:center;
          color:rgb(20, 176, 204);
          font-style:italic;
          font-weight:bolder;
        }
        h2{
          font-size:35px;
          text-align:center;
          color:rgb(17, 196, 227);
          font-style:italic;
          font-weight:bolder;
        }
        h5{
          font-size:25px;
          text-align:center;
          color:rgb(53, 134, 152);
          font-weight:bolder;
        }
    </style>
</head>
<body>
    <div class="bar">
        <a href="/logout" >Logout</a>
        <a href="/upload" >Predict</a>
        <a href="/home">Home</a>
        <br>
```

```
        </div>

        <h1>Prevention is better than cure!</h1>

        <h2><center>Parkinson disease prediction</center></h2>

        <h5>NOTE: Upload a spiral or wave page drawn by the 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>
```

7.4 PRED. 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">
```

```html
        <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>
{% endblock %}
```

7.5 MAIN.CSS

```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: #39D2B4;
    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); }
}
```

7.6 STYLE.CSS

```
*{
    margin: 0;
    padding: 0;
    font-family: sans-serif;
}
.hero{
    height: 100%;
    width: 100%;
    background-image: linear-gradient(rgba(0,0,0,0.4), rgba(0,0,0,0.4)),url(bg.jpg);
    background-position: center;
    background-size: cover;
    position: absolute;
}
.form-box{
    height: 380px;
    width: 360px;
    position: relative;
```

```
    margin: 6% auto;
    background: #fff;
    padding: 5px;
    overflow: hidden;
}
.button-box{
    width: 220px;
    margin: 35px auto;
    position: relative;
    box-shadow: 0 0 20px 9px #5f97e51f;
    border-radius: 40px;
}
.toggle-btn{
    padding: 10px 30px;
    cursor: pointer;
    background: transparent;
    border: 0;
    outline: none;
    position: relative;
}
#btn{
    top: 0;
    left: 0;
    position: absolute;
    width: 110px;
    height: 100%;
    background: linear-gradient(to right,  #7369ca,#11b1c3);
    border-radius: 30px;
```

```css
    transition: 0.5s;
}
.input-group{
    top: 120px;
    position: absolute;
    width: 280px;
    transition: .5s;
}
.input-field{
    width: 100%;
    padding: 10px 0;
    margin: 5px 0;
    border-left: 0;
    border-top: 0;
    border-right: 0;
    border-bottom: 1px solid #999;
    outline: none;
    background: transparent;
}
.submit-btn{
    width: 85%;
    padding: 10px 30px;
    cursor: pointer;
    display: block;
    margin: auto;
    background: linear-gradient(to right, #4e4888,#7bc0c8);
    border: 0;
    outline: none;
```

```
    border-radius: 30px;
}
.check-box{
    margin: 30px 10px 30px 0;
}
span{
    color: #777;
    font-size: 12px;
    bottom: 68px;
    position: absolute;


}
#login{
    left: 50px;
}
#register{
    left: 450px;
}
.err{
    color:rgb(198, 156, 243);
    margin: 265px 0 0 145px;
}
```

7.7 MAIN.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!');
            },
        });
    });
});
```

7.8 APP.PY

```
import os.path
import pickle
import cv2
from flask import Flask, render_template, request
from skimage import feature
app = Flask(__name__)
```

```
@app.route('/')
def hello_world():
    return render_template("index.html")
class my_dictionary(dict):
  def __init__(self):
    self = dict()
  def add(self, key, value):
    self[key] = value
database=my_dictionary()


@app.route('/form_reg',methods=['POST','GET'])
def reg():
    name2=request.form['userid']
    pwd1=request.form['pwd']
    if name2 in database:
        return render_template('index.html',info='UserName Already Taken!!')
    else:
        database.add(name2,pwd1)
        return render_template("index.html")


@app.route('/form_login',methods=['POST','GET'])
def login():
    name1=request.form['userid']
    pwd=request.form['pwd']
    if name1 not in database:
                return render_template('index.html',info='Invalid User!!')
    else:
        if database[name1]!=pwd:
            return render_template('index.html',info='Invalid Password!!')
        else:
```

```
            return render_template('home.html',name=name1)


@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("index.html")


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


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

```
        # 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]

        # 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]]
        return result
    return None

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

7.9 IMAGE_PRE_PROCESSING.IPYNB

```
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
sns.set()
os.getcwd()
from google.colab import files
uploaded = files.upload()
handle_spiral = zf.ZipFile(r'dataset1.zip')
handle_spiral.extractall('dataset1')
handle_spiral.close()
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/'
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
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]
```

```
  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
trainY
testX
testY
le = LabelEncoder()
trainY = le.fit_transform(trainY)
testY = le.transform(testY)
print(trainX.shape,trainY.shape)
trainY
testY
print("Training model....")
model = RandomForestClassifier(n_estimators=100)
model.fit(trainX,trainY)
preds = model.predict(testX)
```

```
preds
cnf = confusion_matrix(testY,preds)
cnf
plt.figure(figsize=(5,5))
sns.heatmap(cnf , annot=True , cmap="coolwarm" , cbar=False)
plt.show()
acc = metrics.accuracy_score(testY,preds)
acc
indexes = np.random.randint(0,30,25)
indexes
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)
```

```
'''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)
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)
pickle.dump(model,open('parkinson.pkl','wb'))
```

8. TESTING

8.1 TEST CASES

Test case ID	Test Scenario	Expected Result	Status
Login_TC_001	Verify the login details of the user, if invalid details are uploaded, Inform that it is invalid.	If valid, user enters the dashboard	Pass
ImageUpload_TC_002	When the user uploads the image the HOG quantifies it and gives the result	The Hand-drawn image should be processed and predicted by the model.	Pass
Prediction_TC_003	Process the uploaded image with high accuracy	Display whether the user is affected or not by the disease.	Pass
Information_TC_004	Check if the UI of the web application looks appealing.	Information Page with good look and feel is displayed	Pass

Activate W

9. RESULTS

9.1 PERFORMANCE METRICS

S.NO	PARAMETERS	VALUES	SCREENSHOT
1.	Metrics	Classification model : Confusion matrix Accuracy score Classification report	[ ] 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 5 10] 0.8
2.	Tune the model	Hyper-parameter tuning validation method	print("Training model....") model = RandomForestClassifier(n_estimators=100) model.fit(trainX,trainY) Training model.... RandomForestClassifier()

10. ADVANTAGES & DISADVANTAGES

Advantages :

- Our machine learning model has used Random forest classifier method which has not been implemented in other Parkinson's disease prediction model and we have used histogram of gradients which will quantify the images that are uploaded for prediction.
- Our web application has a separate page that displays the information about the disease, the symptoms and the treatments which can be done to overcome it and it also overcomes human error rates.

Disadvantages :

- The observational data lacked some potential predictors of Parkinson's disease, for example, speech impairment.
- It produces fake results if the input data is entered wrong

11. CONCLUSION

It is possible to detect Parkinson`s disease using the drawings alone instead of measuring the speed and pressure of the pen on paper. Here, we have included studies in a high-level summary, providing access to information including machine learning methods that have been used in the diagnosis of PD and associated outcomes, which includes types of clinical, behavioral, and biometric data that could be used for rendering more accurate diagnosis, potential biomarkers for assisting clinical decision making, and other highly relevant information, including databases that could be used to enlarge and enrich smaller datasets. In summary, realization of machine learning-assisted diagnosis of PD yields high potential for a more systematic clinical decision-making system, while adaptation of novel biomarkers may give rise to easier access to PD diagnosis at an earlier stage.

12. FUTURE SCOPE

With a single test value, we can't predict whether the user have Parkinson's disease or not. Hence in future, we are planning to integrate more predicting symptoms like speech impairment, medical history collection and also make a prediction based on the past test reports by making this model real we can make more accurate prediction. The future work includes the speech dataset, various other symptoms of the Parkinson's people can also be collected to detect PD in a very early stage. The dataset of various motor and nonmotor symptoms could also be collected and analyzed in PD detection

13. APPENDIX

- Classification of handwritten drawings of people with Parkinson's disease by using histograms of oriented gradients and the random forest classifier - João Paulo Folador et al, 2 October 2019
- Early Detection of Parkinson's Disease using Contrast Enhancement Techniques and CNN - Ishan Vatsaraj et al, 5 May 2021
- Detection of Parkinson's disease using machine learning algorithm - Shikha Singh et al, 22 April 2022