

# **DETECTION OF PARKINSON'S DISEASE** **USING MACHINE LEARNING**

## **A PROJECT REPORT**

**Submitted by**  
**TEAM ID: PNT2022TMID24130**

### **TEAM MEMBERS:**

**R YASHASVI CHOWDARY**

**N ANKAIAH**

**P CHETHAN PRASAD**

**P PUSHKALA SAI**

## **1. INTRODUCTION**

### **1.1 Project Overview**

Parkinson's disease is a progressive disorder of the central nervous system affecting movement and inducing tremors and stiffness. It has 5 stages to it and affects more than 1 million individuals every year in India. This is chronic and has no cure yet. It is a neurodegenerative disorder affecting dopamine-producing neurons in the brain. For detecting PD, various machine learning models such as logistic regression, naive Bayes, KNN, and forest decision tree were used, with the features used here being minimum-redundancy maximum-relevance and recursive feature elimination. The accuracy obtained was 95.3% using data from the UCI machine learning repository. 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

By using machine learning techniques, the problem can be solved with minimal error rate. The voice dataset of Parkinson's disease from the UCI Machine learning library is used as input. Also our proposed system provides accurate results by integrating spiral drawing inputs of normal and Parkinson's affected patients. Machine learning also allows for combining different modalities, such as magnetic resonance imaging (MRI) and single-photon emission computed tomography (SPECT) data. in the diagnosis of PD. By using machine learning approaches, we may therefore identify relevant features that are not traditionally used in the clinical diagnosis of PD and rely on these alternative measures to detect PD in preclinical stages or atypical forms. In recent years, the number of publications on the application of machine learning to the diagnosis of PD has increased. Although previous studies have reviewed the use of machine learning in the diagnosis and assessment of PD, they were limited to the analysis of motor symptoms, kinematics, and wearable sensor data. Moreover, some of these reviews only included studies published between 2015 and 2016. In this study, we aim to comprehensively summarize all published studies that applied machine learning models to the diagnosis of PD for an exhaustive overview of data sources, data types, machine learning models, and associated outcomes, (b) assess and feasibility and efficiency of different machine learning methods in the diagnosis of PD, and (c) provide machine learning practitioners interested in the diagnosis of PD with an overview of previously used models and data modalities and the associated outcomes, and recommendations on how experimental protocols and results could be reported to facilitate reproduction. As a result, the application of machine learning to clinical and non-clinical data of different modalities has often led to high diagnostic accuracies in human participants, therefore may encourage the adaptation of machine learning algorithms and novel biomarkers in clinical settings to assist more accurate and informed decision making. While Parkinson's cannot be cured, early detection along with proper medication can significantly improve symptoms and quality of life.

## 2. LITERATURE SURVEY

### 2.1 Existing problem

More than 10 million people are living with Parkinson's Disease worldwide, according to the Parkinson's Foundation. While Parkinson's cannot be cured, early detection along with proper medication can significantly improve symptoms and quality of life. 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.

### 2.2 References

Ioannis G. Tsoulos, Georgia Mitsi, Athanassios Stavrakoudis and Spyros

Papapetropoulos(2019) proposed the application of Machine Learning in a Parkinson's Disease Digital Biomarker Dataset Using Neural Network Construction Methodology Discriminates Patient Motor Status where the objective is to provide preliminary evidence that artificial intelligence systems may allow one to discriminate PD patients from and determine different features of the disease. The recently introduced Neural Network Construction technique was used here to classify data collected by a mobile application into two categories. The NNC algorithm discriminated individual PD patients from HVs with 93.11% accuracy and ON vs OFF states with 76.5% accuracy.

Hakan Gunduz proposed Deep Learning-Based Parkinson's Disease Classification Using Vocal Feature Sets(2019) Parkinson's Disease (PD) is a progressive neurodegenerative disease with multiple motor and non-motor characteristics. PD patients commonly face vocal impairments during the early stages of the disease. So, diagnosis systems based on vocal disorders are at the forefront on recent PD detection studies. Our study proposes two frameworks based on Convolutional Neural Networks to classify Parkinson's Disease (PD) using sets of vocal (speech) features. Extracted deep features are not only successful at distinguishing PD patients from healthy individuals but also effective in boosting up the discriminative power of the classifiers.

Oliver Y. Chén , Florian Lipsmeier , Huy Phan , John Prince , Kirsten I. Taylor, Christian Gossens, Michael Lindemann, and Maarten de Vos(2020) proposed a Machine-Learning Framework to

Remotely Assess Parkinson's Disease Using Smartphones. Using smartphones, remote patient monitoring has the potential to obtain objective behavioral data semi-continuously, track disease fluctuations, and avoid rater dependency. Methods: Smartphones collect sensor data during various active tests and passive monitoring, including balance (postural instability), dexterity (skill in performing tasks using hands), gait (the pattern of walking), tremor (involuntary muscle contraction and relaxation), and voice. Data analysis results from 437 behavioral features obtained from 72 subjects (37 PD and 35 HC) sampled from 17 separate days during a period of up to six months suggest that this framework is potentially useful for the analysis of remotely collected smartphone sensor data in individuals with PD.

S. Sharanyaa, P N. Renjith and K. Ramesh(2020) proposed the Classification of Parkinson's Disease using Speech Attributes with Parametric and Nonparametric Machine Learning Techniques which evaluate the performance of state of art algorithms to detect Parkinson's disease with higher classification accuracy. The performance is evaluated by pre-processing the data based on speech attributes. Various performance metrics are computed for all four machine learning techniques and the results show that nonparametric models produce higher classification accuracy of 87.2% and 90.2% compared to parametric models.

Wu Wang, Junho Lee, Fouzi Harrou and Ying Sun(2020) proposed detecting Parkinson's disease (PD) at an early stage is certainly indispensable for slowing down its progress and providing patients the possibility of accessing to disease-modifying therapy. A comparison between the proposed deep learning model and twelve machine learning and ensemble learning methods based on relatively small data including 183 healthy individuals and 401 early PD patients shows the superior detection performance of the designed model, which achieves the highest accuracy, 96.45% on average. Besides detecting the PD, we also provide the feature importance on the PD detection process based on the Boosting method.

Julián D. Loaiza Duque, Antonio J. Sánchez Egea, Theresa Reeb, Andrés M. González Vargas(2020) proposed Angular Velocity Analysis Boosted by Machine Learning for Helping in the Differential Diagnosis of Parkinson's Disease and Essential Tremor. This work aims to develop Machine Learning models to improve the differential diagnosis between patients with Parkinson's Disease and Essential Tremor. For this purpose, we use a mobile phone's built-in gyroscope to record the angular velocity signals of two different arm positions during the patient's follow-up, more precisely, in rest and posture positions. The models developed reached an average accuracy of  $97.2 \pm 3.7\%$  (98.5% Sensitivity, 93.3% Specificity) to

differentiate between Healthy and Trembling subjects and an average accuracy of  $77.8 \pm 9.9\%$  (75.7% Sensitivity, 80.0% Specificity) to discriminate between Parkinson's Disease and Essential Tremor patients.

Luigi Borzì, Ivan Mazzetta, Alessandro Zampogna, Antonio Suppa, Gabriella Olmo and Fernanda Irrera(2021) proposed the Prediction of Freezing of Gait in Parkinson's Disease Using Wearables and Machine Learning the aim was to propose a wearable system able to catch the typical degradation of the walking pattern preceding FOG episodes, to achieve reliable FOG prediction using machine learning algorithms and verify whether dopaminergic therapy affects the ability of our system to detect and predict FOG. The classification model was trained with data from patients on (off) and tested on patients off (on) and found 84.0% (56.6%) sensitivity, 88.3% (92.5%) specificity and 87.4% (86.3%) accuracy.

Aleksandr Talitskii, Anna Anikin,Ekaterina Kovalenko, Aleksei Shcherbak ,Oscar Mayora , Olga Zimniakova,Ekaterina Bril , Maxim Semenov , Dmitry V. Dylov, and Andrey Somo(2021)proposed the research in the area focuses on how to detect, predict, or classify PD and similar diseases without addressing the point of what activities or exercises a subject should do to improve the performance of these tasks.we collect the data in a real clinical setting using a compact wearable wireless sensor node entailing a board gyroscope, accelerometer, and magnetometer.Application of ML methods to the collected data reveals three “most efficient” exercises to assist diagnosticians with the highest discriminating power (0.9 ROC AUC in each task). The proposed solution can be implemented as a medical decision support system for real-time PD diagnostics.

Trevor Exley(2022) proposed a prediction of UPDRS Motor Symptoms in Individuals With Parkinson's Disease From Force Plates Using Machine Learning.Parkinson's disease (PD) is a neurodegenerative disease that affects motor abilities with increasing severity as the disease progresses. Traditional methods for diagnosing PD include a section where a trained specialist scores qualitative symptoms using the motor subscale of the Unified Parkinson's Disease Rating Scale (UPDRS-III). Quiet standing can detect body bradykinesia (AUC-ROC = 0.924) and postural stability (AUC-ROC = 0.967) with high predictability.

Johann Faouzi , Samir Bekadar, Fanny Artaud , Alexis Elbaz , Graziella Mangone, Olivier Colliot, and Jean-Christophe Corvol(2022) proposed Machine Learning-Based Prediction of Impulse Control Disorders in Parkinson's Disease From Clinical and Genetic Data.Impulse control disorders (ICDs) are frequent

non-motor symptoms occurring during the course of Parkinson's disease (PD). The objective of this study was to estimate the predictability of the future occurrence of these disorders using longitudinal data, the first study using cross-validation and replication in an independent cohort. Methods: We used data from two longitudinal PD cohorts (training set: PPMI, Parkinson's Progression Markers Initiative; test set: DIGPD, Drug Interaction With Genes in Parkinson's Disease). Results: The recurrent neural network (PPMI: 0.85 [0.80 – 0.90], DIGPD: 0.802 [0.78 – 0.83]) was the only model to be significantly better than the trivial model (PPMI: ROC AUC = 0.75 [0.69 – 0.81]; DIGPD: 0.78 [0.75 – 0.80]) on both cohorts.

## 2.3 Problem Statement Definition

Parkinson's disease is a chronic progressive disease of the nervous system characterised by the cardinal features of rigidity, bradykinesia, tremor and postural instability. However there is no recognized test for PD for patients, particularly in the early stages. This results in increased mortality rate. Thus detection system of Parkinson's disease with easy steps and feasible one to detect parkinson's disease at the early stage is essential.

Who does the problem affect?	People who are men with minimization of nerve cells in primarily of village areas.
What are the boundaries of the problem?	People who are men with weak nerve cells and age over 50
What is the issue?	<p>In real time life of human, if the person is affected by Parkinson disease then it produces the side effect problems like dry skin and dandruff which majorly affects the quality of the life.</p> <p>As the age gets progresses, it causes the people to face major problem with the nerve cells in the brain.</p>

When does the issue occur?	During the age excess of over 50 as they will affect the people with loss of nerve cells in the brain.
----------------------------	--

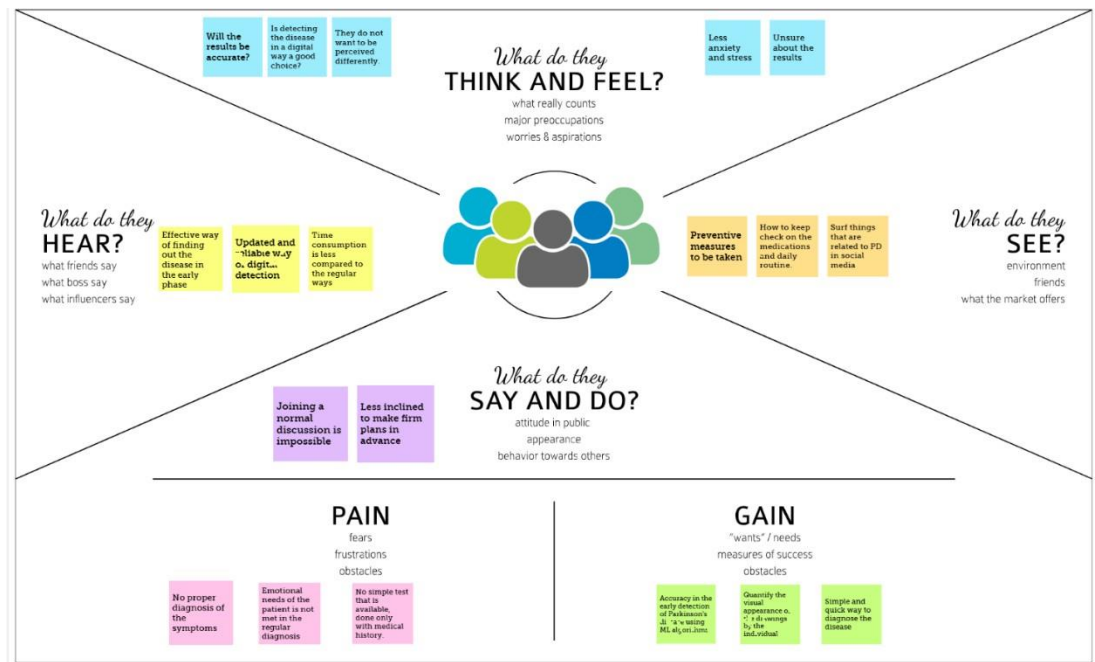
Where is the issue coming?	It majorly occurs due to the age getting over 50 and as maximum in village areas.
Why is it important that we fix the problem?	It is very crucial to develop a application that detects the disease at good prediction rate so that it helps to get a clear line of disease symptoms during the times.

Which solution can be used to address this issue?	An machine learning powered web application model with the strong building of algorithm that helps to identify and predicts the disease with the identification of symptoms. It processes the breathing signals using a neural network that infers whether the person has Parkinson's disease, and if they are identified then it assesses the severity of their disease in accordance with the Movement Disorder Society Unified Parkinson's Disease using ML algorithms.
What methodology used to solve the issue?	Supervised and Un-supervised machine learning, Data mining , Computer vision with OpenCV, Python web application interface – Flask , IBM Cloud.

### **3.IDEATION & PROPOSED SOLUTION**

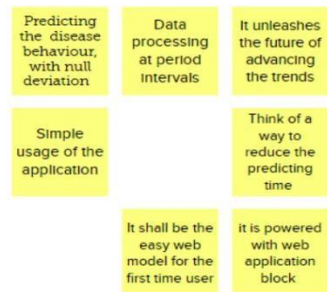
#### **3.1 Empathy Map Canvas**



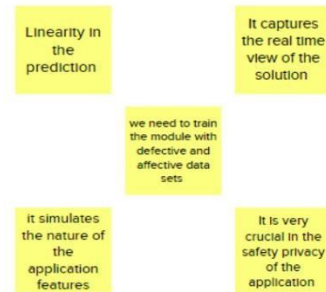


## 3.2 Ideation & Brainstorming

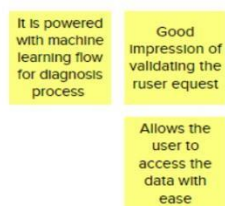
### V P Aswin Kumar (TL)



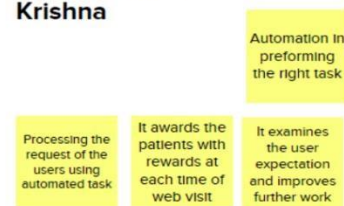
### S chethen

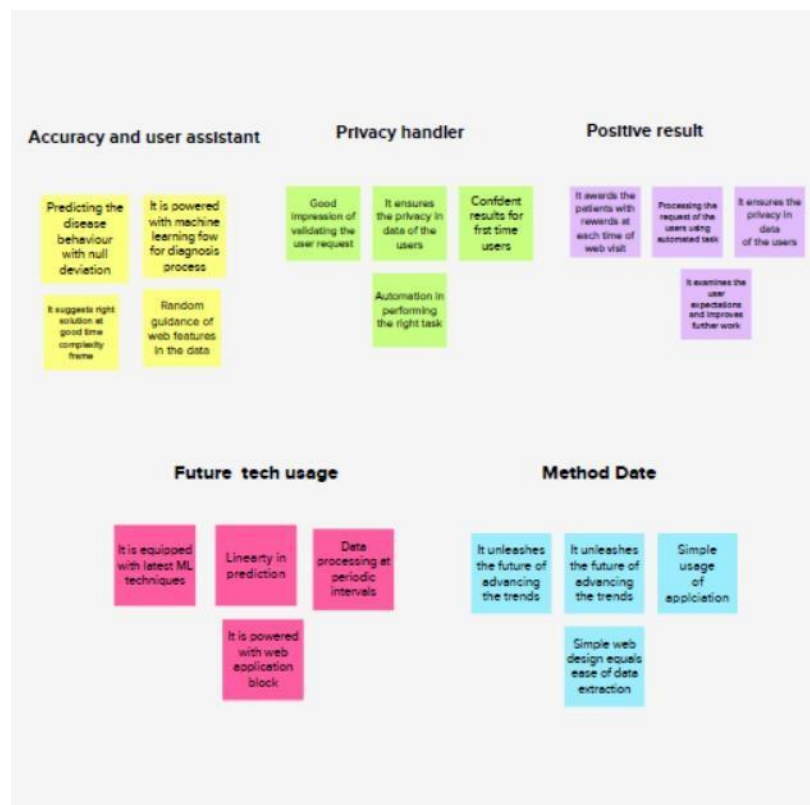
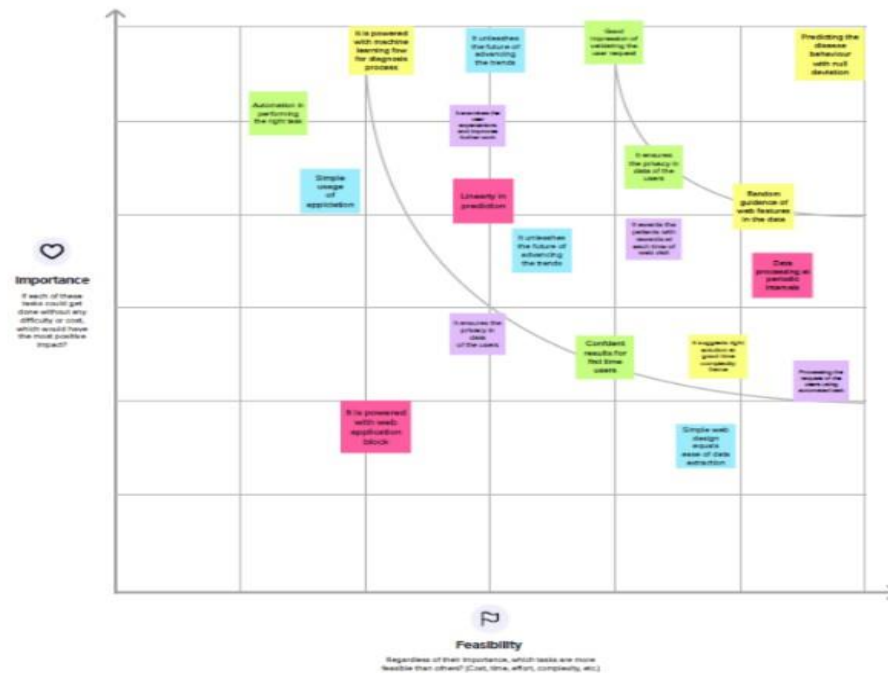


### Vishnu Teja Ramisetty



### Vujjini Vamisi Krishna





### 3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Early and Automatic detection of Parkinson's Disease in hand drawn images.
2.	Idea / Solution description	Detection is done by using Histogram of Oriented Gradients (HOG) image descriptor along with a Random Forest classifier.
3.	Novelty / Uniqueness	In this method, the approach to predict Parkinson's disease from hand-drawn wave and spiral images using computer vision and machine learning techniques has been recommended. The previous methods have their constraints.
4.	Social Impact / Customer Satisfaction	People can detect the disease at a very early stage and improve the quality of living. They can take proper precautions and lead a healthy and safe life.
5.	Business Model (Revenue Model)	It is cost efficient as it is a Software as a Service Platform. People need not spend much money to detect the disease.
6.	Scalability of the Solution	Better execution in accuracy, sensitivity, and specificity as well as in system design flexibility.

### 3.4 Problem Solution fit

Define CS, fit into CC	<div>1. CUSTOMER SEGMENT(S)<div>CS</div><div><div><div><div></div></div><div>Person with Parkinson Disease Symptoms</div></div><div><div><div></div></div><div>Those over 65 who are in the high-risk zone for the illness.</div></div></div></div>	<div>6. CUSTOMER CONSTRAINTS<div>CC</div><div><div><div><div></div></div><div>appropriate Internet connectivity is available.</div></div><div><div><div></div></div><div>Power supply when using a desktop computer</div></div><div><div><div></div></div><div>Does not require the cost for treatment</div></div><div><div><div></div></div><div>No requirement to visit a hospital for their checkup</div></div></div></div>	<div>5. AVAILABLE SOLUTIONS<div>AS</div><div><div><div><div></div></div><div><b>Solution available :</b> By using ML, we can detect early stage of PD and accurate result by checking with voice changes dataset and integrating with spiral drawing dataset</div></div><div><div><div></div></div><div><b>Fast techniques :</b> PD is generally diagnosed by MRI(it can't detect early stage of PD),SPECT(Titan reveal in brain chemistry, such as a decrease in dopamine) and PET(it detect brain regions involvement) Scan .This result in high Misdiagnosis rate</div></div><div><div><div></div></div><div><b>Pros :</b> As a result of using ML, the physician can determine whether anything is normal or aberrant and then provide medication in accordance with the afflicted stage.</div></div><div><div><div></div></div><div><b>Cons :</b> The current approach only detects PD at the secondary stage.Diagnosis suggestion also difficult</div></div></div></div>	Explore AS, differentiate
	<div>2. JOBS-TO-BE-DONE / PROBLEMS<div>J&amp;P</div><div><div><div><div></div></div><div>Spread Awareness about the Disease</div></div><div><div><div></div></div><div>Since L-DOPA and co-careldopa therapy are most effective when started when Parkinson's disease (PD) is still in its early stages, early diagnosis of PD is essential.</div></div><div><div><div></div></div><div>clinical evaluation is done in online</div></div><div><div><div></div></div><div>Treatment records are update according to their changes.</div></div><div><div><div></div></div><div>It will calculate the treatment date and set the alarm for remembering purpose.</div></div></div></div>	<div>9. PROBLEM ROOT CAUSE<div>RC</div><div><div><div><div></div></div><div>Lack of knowledge about the illness, which allowed people to ignore early signs. Additionally, this may result in people not knowing where to seek medical attention or what their best course of action is.</div></div><div><div><div></div></div><div>A lot of people are uncertain about the conditions.</div></div><div><div><div></div></div><div>People are reluctant to get testing because of the expense and travel considerations.</div></div><div><div><div></div></div><div>Many comparable symptoms can lead people to make faulty judgments.</div></div></div></div>	<div>7. BEHAVIOUR<div>BE</div><div><div><div><div></div></div><div>Participate in studies on the disease, its symptoms, and possible cures.</div></div><div><div><div></div></div><div>The topic or consumer could inquire of friends and family of any diagnosing facilities or request general assistance in finding the appropriate personnel.</div></div><div><div><div></div></div><div>After visiting multiple web pages online, the buyer could develop confirmation bias and begin to feel anxious.</div></div><div><div><div></div></div><div>Look for measures to stop the spread of the illness.</div></div><div><div><div></div></div><div>Prepare for a crisis situation.</div></div></div></div>	Focus on J&P, fit into BE, understand RC
Identify strong TR & EM	<div>3. TRIGGERS<div>TR</div><div><div><div><div></div></div><div>Publications on social media that promote awareness.</div></div><div><div><div></div></div><div>The platform's self-intuitive, simple design that is free and encourages interaction</div></div><div><div><div></div></div><div>Specified symptom-focused advertisements.</div></div></div></div>	<div>10. YOUR SOLUTION<div>SL</div><div><div><div><div></div></div><div>Machine learning methods can be used to tackle the issue with a low error rate. The input is the Parkinson's disease voice dataset from the UCI Machine learning library. Additionally, our suggested technique yields precise results by combining the spiral drawing inputs of Parkinson's patients and healthy individuals. We suggest a method that accurately analyses patient data from spiral drawings and voice recordings. Combining both data, the doctor can determine whether something is normal or abnormal and then give the appropriate medication for the affected stage, we are using the following analysis</div></div><div><div><div><div></div></div><div>1. Voice Dataset Analysis<div><div><div><div></div></div><div>K-means Clustering</div></div><div><div><div></div></div><div>Decision Tree Classification</div></div><div><div><div></div></div><div>Prediction of Output</div></div></div></div></div></div><div><div><div><div></div></div><div>2. Spiral Drawing Analysis(using HOG method)<div><div><div><div></div></div><div>Pre-Processing</div></div><div><div><div></div></div><div>Feature Extraction</div></div><div><div><div></div></div><div>random Forest classification</div></div><div><div><div></div></div><div>Prediction of output</div></div></div></div></div></div></div></div>	<div>8. CHANNELS of BEHAVIOR<div>CH</div><div><div><div><div></div></div><div><b>a. ONLINE</b><div><div><div><div></div></div><div>Browse physician profiles and schedule visits.</div></div><div><div><div></div></div><div>Online prediction is simple and cost-free.</div></div><div><div><div></div></div><div>A user-interactive website that is available at all times to anyone.</div></div></div></div></div></div><div><div><div><div></div></div><div><b>b. OFFLINE</b><div><div><div><div></div></div><div>Make an appointment and go to the doctor.</div></div><div><div><div></div></div><div>Consult with professionals and get recommendations for treatments.</div></div><div><div><div></div></div><div>Diagnosis Suggestion is difficult</div></div><div><div><div></div></div><div>Prediction of early stage of PD is impossible.</div></div></div></div></div></div></div>	Identify strong TR & EM
	<div>4. EMOTIONS: BEFORE / AFTER<div>EM</div><div><div><div><div></div></div><div>Following diagnosis in the absence of the condition: Joy, the elimination of confirmation bias, Peace, and Calmness.</div></div><div><div><div></div></div><div>After receiving a diagnosis: Fear, Overwhelmed, Vulnerable, and Depressed.</div></div><div><div><div></div></div><div>Before: Uncertainty, stress, and anxiety.</div></div></div></div>			

## 4 REQUIREMENT ANALYSIS

### 4.1 Functional requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Home Page	Short description about Parkinson's Disease, its different types, and symptoms along with possible comorbidity management techniques. If the user already has an account, they can log in. Otherwise, they are required to sign up.
FR-2	User Registration (Sign in) Page	User needs to sign up/ register by entering Name, Email address, Phone number and Password.
FR-3	User Confirmation & Verification	Verification will be done via Email or OTP.
FR-4	User Login Page	User can enter their credentials (Email and Password) and log in to their account.
FR-5	User Dashboard	The logged in user is led to a dashboard where the user is asked to upload the image in order to provide the diagnosis. The user is also asked for other parameters such as age, blood type, mobility issues etc for survey purposes. This information is optional and is collected only from willing users.

FR-6	Test input (Copy of handdrawn image)	The input to the prediction system is uploaded as an image. It can be uploaded either using a live drawing notepad or as the digital copy of an already drawn spiral/wave. Image quality evaluation is done in this step to determine whether the image quality is sufficient for processing.
FR-7	User authentication during login	User authentication is done using PHP via database in XAMPP server.
FR-8	Disease prediction by image processing	Classification is carried out using Digital image processing using Histogram of Oriented Gradients (HOG) image descriptor along with a random forest classifier.
FR-9	Recommendation	The prediction system provides a positive or negative diagnosis. It also suggests the specialization doctors that need to be consulted. The system arrives at the result by analysing the standards defined by Movement Disorder Society Unified Parkinson's Rating Scale and progression of the disease.

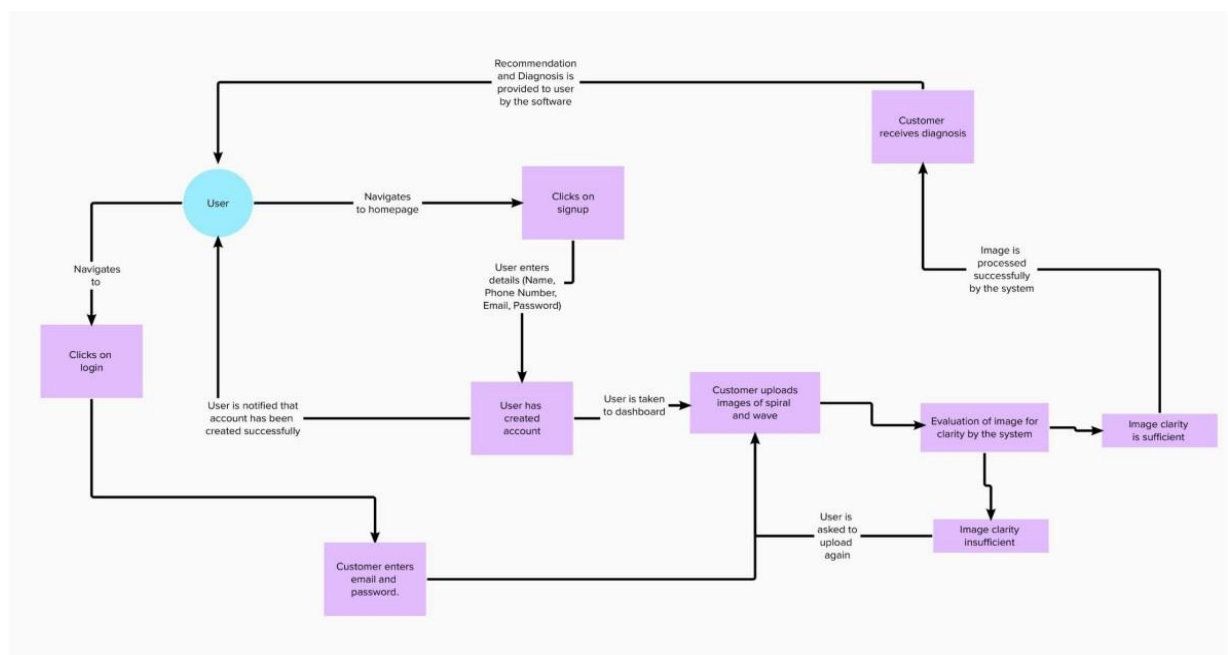
## 4.2 Non-Functional requirements

FR No.	Non-Functional Requirement	Description
NFR-1	<b>Usability</b>	The website can be easily navigated even by the uninitiated user and the functionality that the website provides is simple and easy to understand.
NFR-2	<b>Security</b>	The application is designed to safeguard against threats including unauthorized access and protects the patient's confidentiality by keeping patient details visible only to admin and the patient. Access permissions can only be changed by the system's data administrator.
NFR-3	<b>Reliability</b>	The software will work without failure. It does not have any security bugs.  The model is trained with different visuals for detecting the disease, which leads to a more accurate assessment of a disease, thereby making the system more reliable for its users.

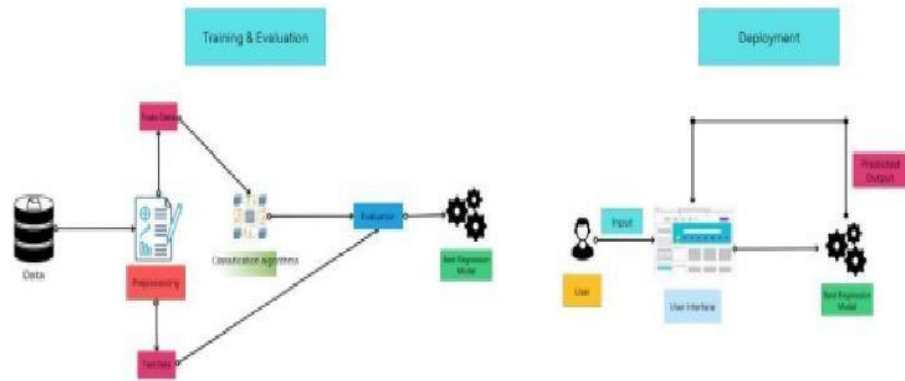
NFR-4	<b>Performance</b>	The system Is very responsive to user interactions with it and can handle a large traffic without getting overloaded. The user wait time is not prolonged,  including capturing and uploading to prediction and providing recommendations.
NFR-5	<b>Availability</b>	The software is always available to the user irrespective of the any new module development. If any backend work requires that the page be unavailable, then a notification is displayed to the user informing when it will up again for use. The software can also be utilized by anyone, regardless of the customer location or other network capabilities.
NFR-6	<b>Scalability</b>	The system has the ability to grow without any negative impact on its performance. The system is designed in a way it can withstand a large number of users at any given moment and if need be, can be scaled up to handle even more users.

## 5 PROJECT DESIGN

### 5.1 Data Flow Diagrams



## 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 (Web user)	Viewing Home Page	USN-1	As a user, I can view the home page which has a description of the disease as well as options to sign up or log in.	I can get to know about the disease and its symptoms as well as navigate to sign up page and log in page from there.	Low	Sprint-1
	Sign Up Page	USN-2	As a user, I can register for the application by entering my name, phone number, email, password, and confirming my password.	I can login with my credentials.	High	Sprint-1
	Authorization	USN-3	As a user, I will receive confirmation email once I have registered for the application.	I can receive confirmation email & click confirm.	High	Sprint-2

	Login	USN-4	As a user, I can log into the application by entering email & password.	I can access my account / dashboard after logging in successfully.	High	Sprint-1
	Dashboard	USN-5	As a user, I can upload images of spiral and wave to the website in order to receive a diagnosis.	I can successfully access the dashboard to upload the images.	High	Sprint-2
	Results	USN-6	As a user, I can receive a diagnosis in addition to recommendations on what I should do now.	I can access the diagnosis and possible available solutions.	High	Sprint-3
Administrator	Data Collection	USN-7	I need to collect data (images of spirals and waves drawn by healthy people and Parkinson's patients).	I have sizable amount of data to split into training set and testing set.	High	Sprint-2
	Data Pre-Processing	USN-8	I need to clean my data and prepare it for model building by doing pre-processing activities such as resizing, converting from RGB to grayscale etc.	I have the dataset ready for model building.	High	Sprint-3
	Model Building	USN-9	I need to build the model using Random Forest Classifier for spiral images and K Nearest Neighbour (KNN) for wave images.	The model is ready for deployment on testing data.	High	Sprint-4
	Model Deployment	USN-10	I need to deploy the Machine Learning model that was built.	The model has been deployed successfully.	Medium	Sprint-5
	Application Building	USN-11	I need to build the website for the application using HTML, CSS etc.	The website is functional.	High	Sprint-3

## 6. PROJECT PLANNING & SCHEDULING

### 6.1 Sprint Planning & Estimation



<b>Sprint</b>	<b>Functional Requirement (Epic)</b>	<b>User Story Number</b>	<b>User Story / Task</b>	<b>Story Points</b>	<b>Priority</b>	<b>Team Members</b>
Sprint-4	Viewing Home Page	USN-1	As a user, I can view the home page which has a description of the disease as well as options to sign up or log in.	2	Low	Yashasvi Chowdary Ankaiah Chethan Prasad Pushkala Sai
Sprint-4	Sign Up Page	USN-2	As a user, I can register for the application by entering my name, phone number, email, password, and confirming my password.	2	High	Yashasvi Chowdary Ankaiah Chethan Prasad Pushkala Sai
Sprint-4	Authorization	USN-3	As a user, I will receive confirmation email once I have registered for the application.	2	High	Yashasvi Chowdary Ankaiah Chethan Prasad Pushkala Sai
Sprint-4	Login	USN-4	As a user, I can log into the application by entering email & password.	2	High	Yashasvi Chowdary Ankaiah Chethan Prasad Pushkala Sai
Sprint-4	Dashboard	USN-5	As a user, I can upload images of spiral and wave to the website in order to receive a diagnosis.	2	High	Yashasvi Chowdary Ankaiah Chethan Prasad Pushkala Sai
Sprint-4	Results	USN-6	As a user, I can receive a diagnosis in addition to recommendations on what I should do now.	2	High	Yashasvi Chowdary Ankaiah Chethan Prasad Pushkala Sai

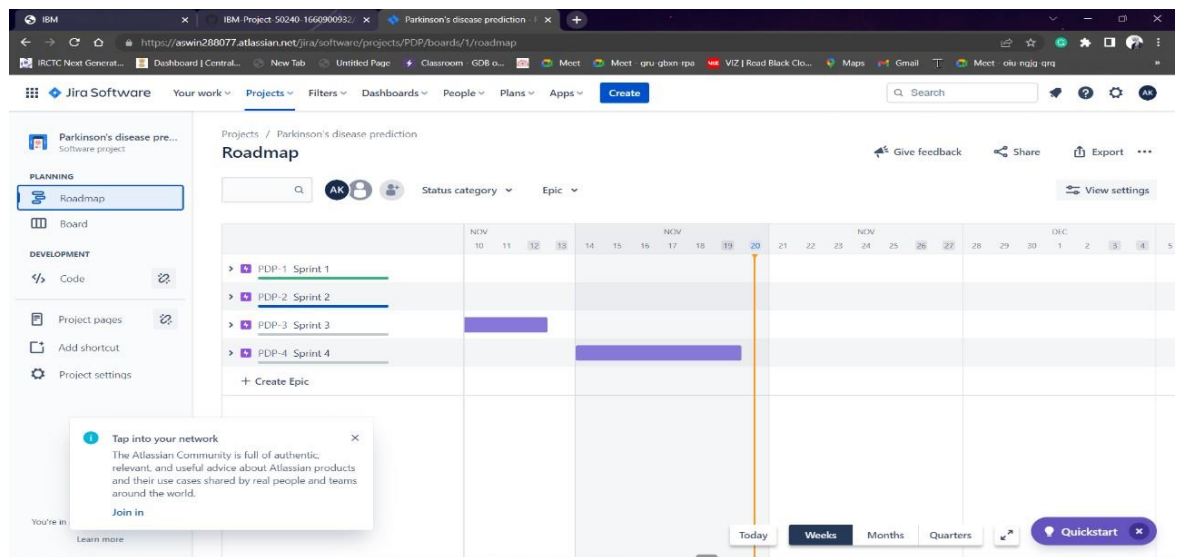
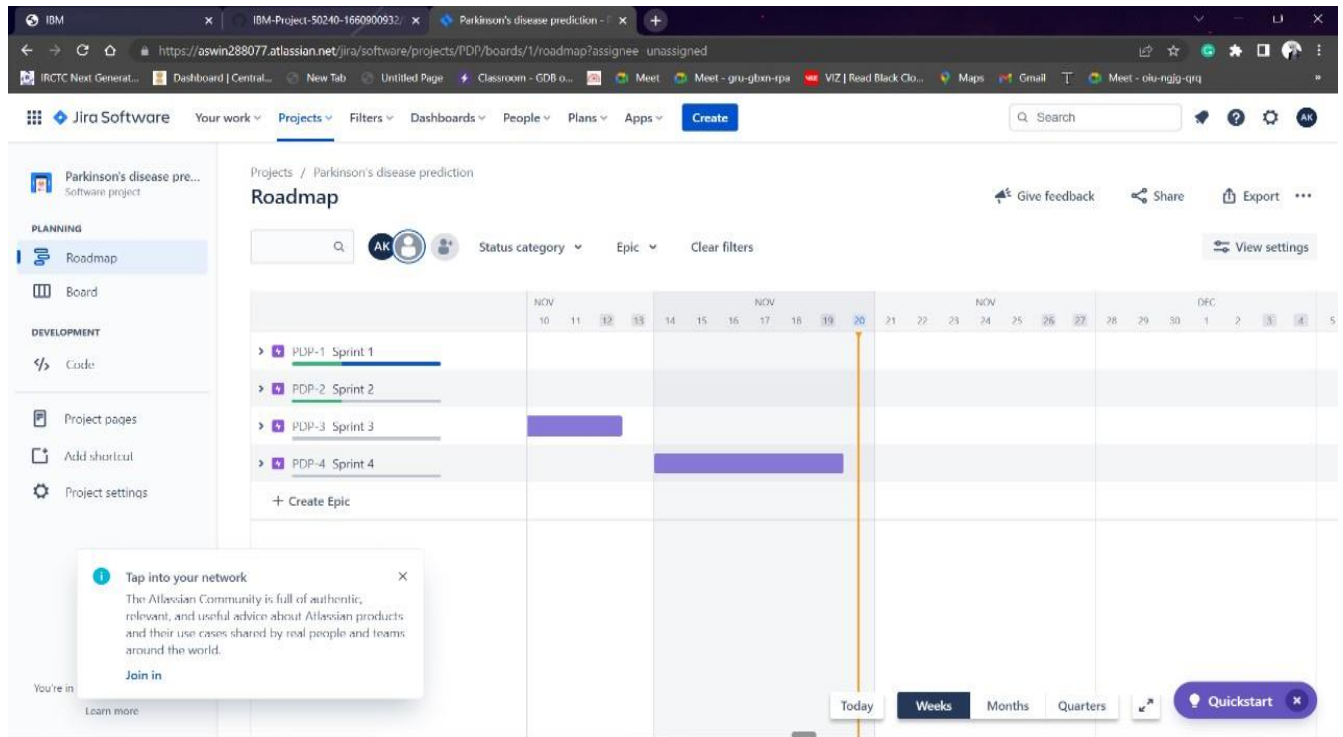
Sprint-1	Data Collection	USN-7	I need to collect data (images of spirals and waves drawn by healthy people and Parkinson's patients).	5	High	Yashasvi Chowdary Ankaiah Chethan Prasad Pushkala Sai
Sprint-1	Data Pre-Processing	USN-8	I need to clean my data and prepare it for model building by doing pre-processing activities such as resizing, converting from RGB to grayscale etc.	5	High	Yashasvi Chowdary Ankaiah Chethan Prasad Pushkala Sai

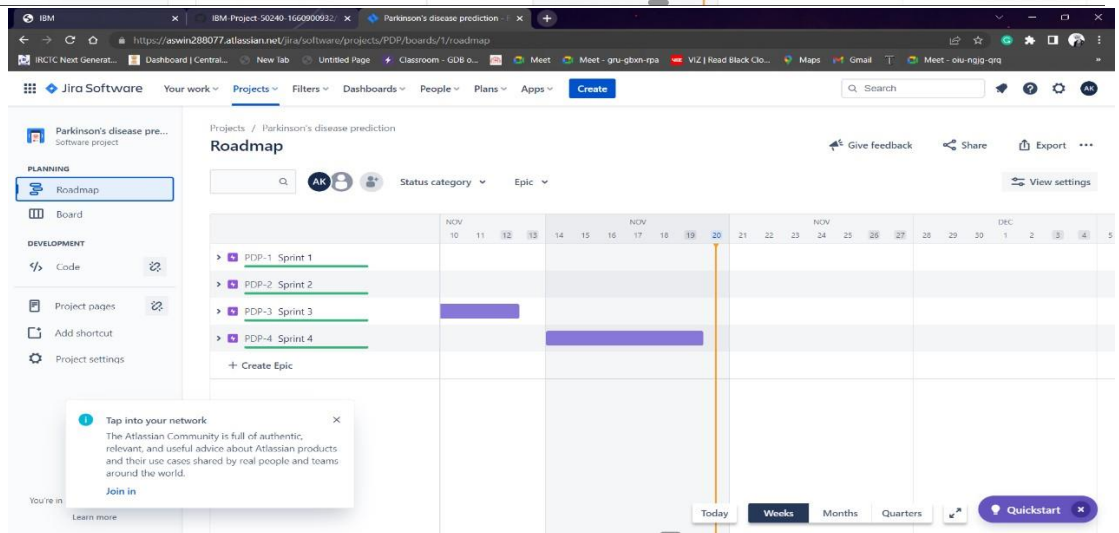
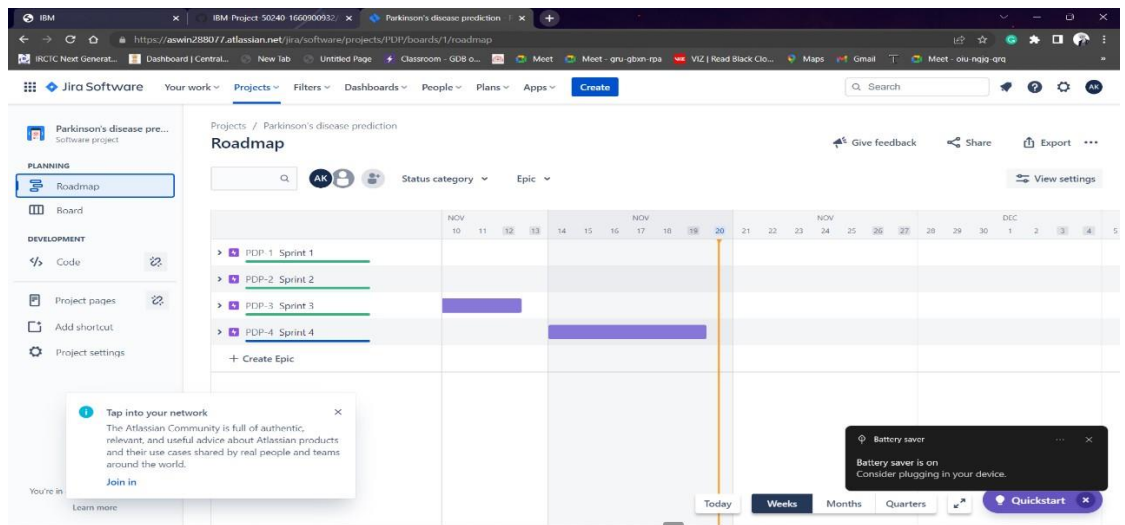
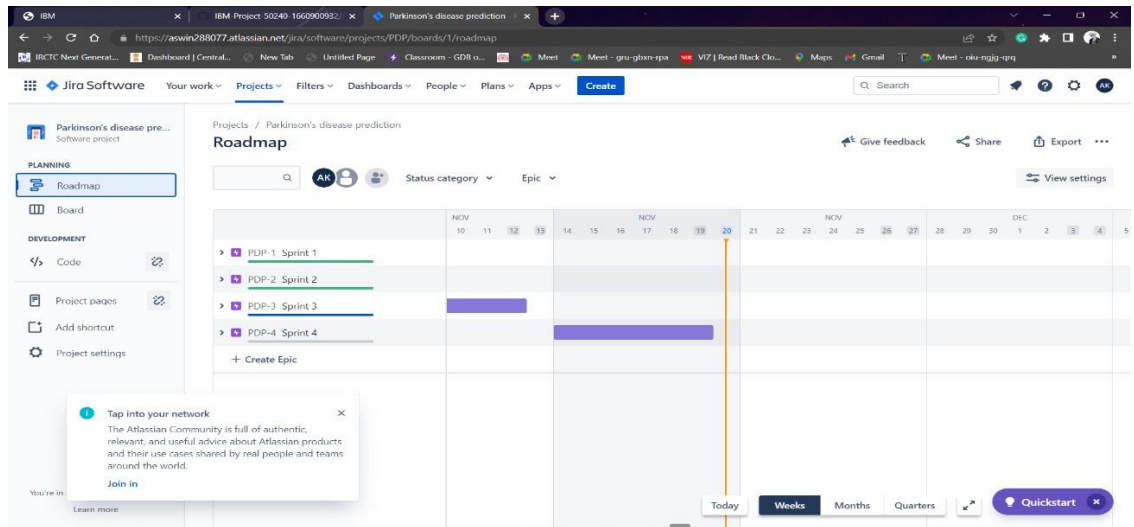
Sprint-2	Model Building 1	USN-9	I need to build the model using Random Forest Classifier for spiral images.	8	High	Yashasvi Chowdary Ankaiah Chethan Prasad Pushkala Sai
Sprint 2	Model Building 2	USN-10	I need to build the model using K Nearest Neighbour (KNN) for wave images.	8	High	Yashasvi Chowdary Ankaiah Chethan Prasad Pushkala Sai
Sprint-3	Model Deployment	USN-11	I need to deploy the Machine Learning model that was built.	13	Medium	Yashasvi Chowdary Ankaiah Chethan Prasad Pushkala Sai
Sprint-4	Application Building	USN-12	I need to build the website for the application using HTML, CSS and link it to the model.	8	High	Yashasvi Chowdary Ankaiah Chethan Prasad Pushkala Sai n

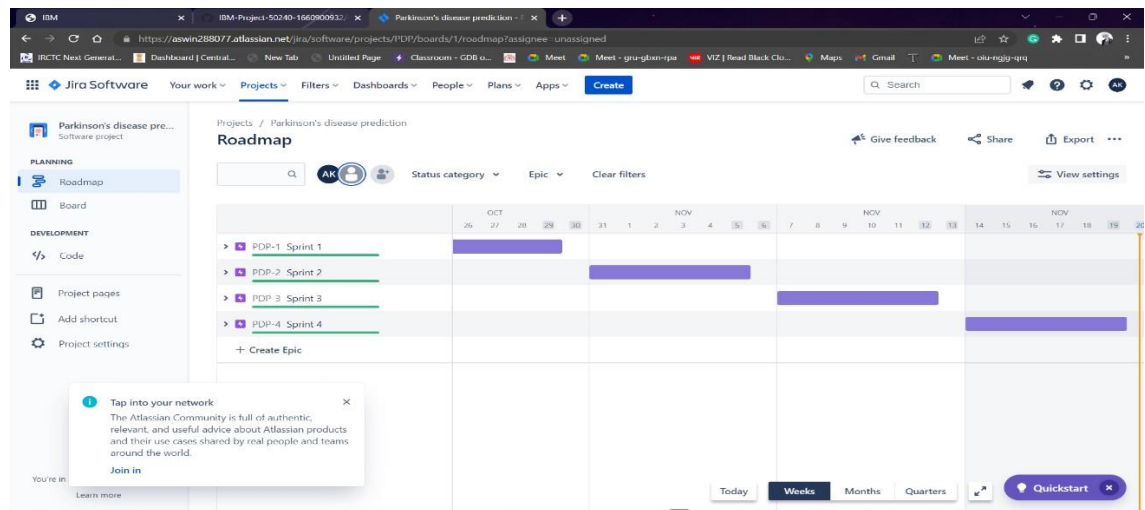
## 6.2 Sprint Delivery Schedule

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	vishun19 Nov 2022

## 6.3 Reports from JIRA







## 7 CODING & SOLUTIONING

### 7.1 Feature 1

i. We have trained the ML model for detecting Parkinsons Disease using RandomForestClassifier with the accuracy of 86.7%.

#### Image Pre-Processing

##### Import the necessary Libraries

Note: Download scikit-image for skimage

```
In [1]: from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import confusion_matrix, accuracy_score, classification_report
from skimage import feature
from imutils import build_montages
from imutils import paths
import numpy as np
import cv2
import os
import pickle
```

## Path for train and test data

```
In [2]: trainingpath=r"dataset/spiral/training"
testingpath=r"dataset/spiral/testing"
```

## Quantifying Images

```
In [3]: 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
```

## Loading Train Data and Test Data

```
In [4]: def load_split(path):
imagePaths = list(paths.list_images(path))
data = []
labels = []

for imagePath in imagePaths:
    label = imagePath.split(os.path.sep)[-2]

    image = cv2.imread(imagePath)
    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)

    data.append(features)
    labels.append(label)

return (np.array(data), np.array(labels))
```

## Model Evaluation

```
In [12]: predictions = model.predict(X_test)
predictions
```

```
Out[12]: array([[0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1,
1, 1, 1, 0, 0, 1, 1, 1], dtype=int64)
```

```
In [13]: cm = confusion_matrix(y_test, predictions)
cm
```

```
Out[13]: array([[14,  1],
[ 3, 12]], dtype=int64)
```

```
In [14]: accuracy = accuracy_score(y_test, predictions)
accuracy
```

```
Out[14]: 0.8666666666666667
```

```
In [15]: cr = classification_report(y_test, predictions)
print(cr)
```

	precision	recall	f1-score	support
0	0.82	0.93	0.87	15
1	0.92	0.80	0.86	15
accuracy			0.87	30
macro avg	0.87	0.87	0.87	30
weighted avg	0.87	0.87	0.87	30

## Testing The Model

```
In [8]: testingpath=list(paths.list_images(testingpath))
        idxs=np.arange(0,len(testingpath))
        idxs=np.random.choice(idxs,size=(25,),replace=False)
        images=[]

In [9]: for i in idxs:
        image=cv2.imread(testingpath[i])
        output=image.copy()

        # Load the input image,convert to grayscale and resize

        output=cv2.resize(output,(128,128))
        image=cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
        image=cv2.resize(image,(280,280))
        image=cv2.threshold(image,0,255,cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]

        #quantify the image and make predictions based on the extracted feature using last trained random forest
        features=quantify_image(image)
        preds=model.predict([features])
        label=le.inverse_transform(preds)[0]
        #the set of output images
        if label=="healthy":
            color=(0,255,0)
        else:
            color=(0,0,255)

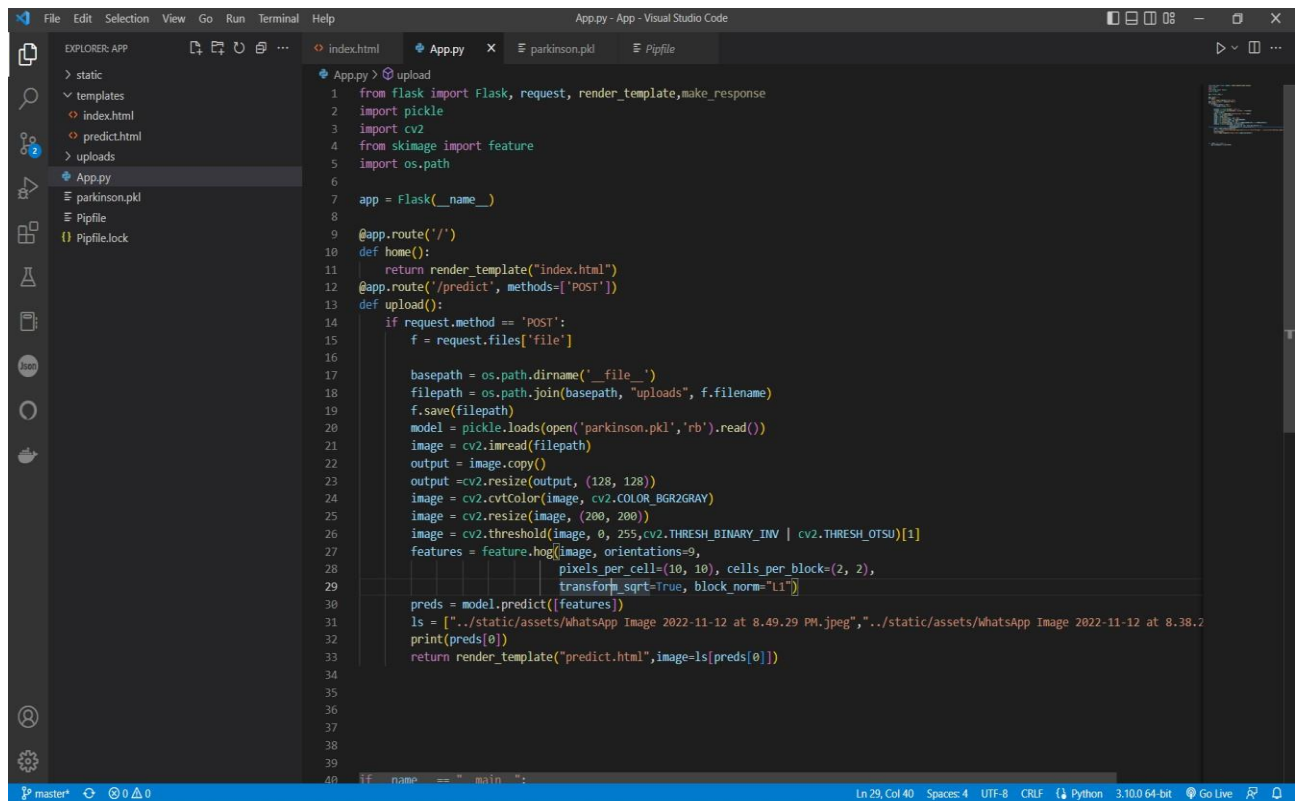
        cv2.putText(output,label,(3,20),cv2.FONT_HERSHEY_SIMPLEX,0.5,color,2)
        images.append(output)

        #creating a montage
        montage=build_montages(images,(128,128),(5,5))[0]
        cv2.imshow("Output",montage)
        cv2.waitKey(0)
```

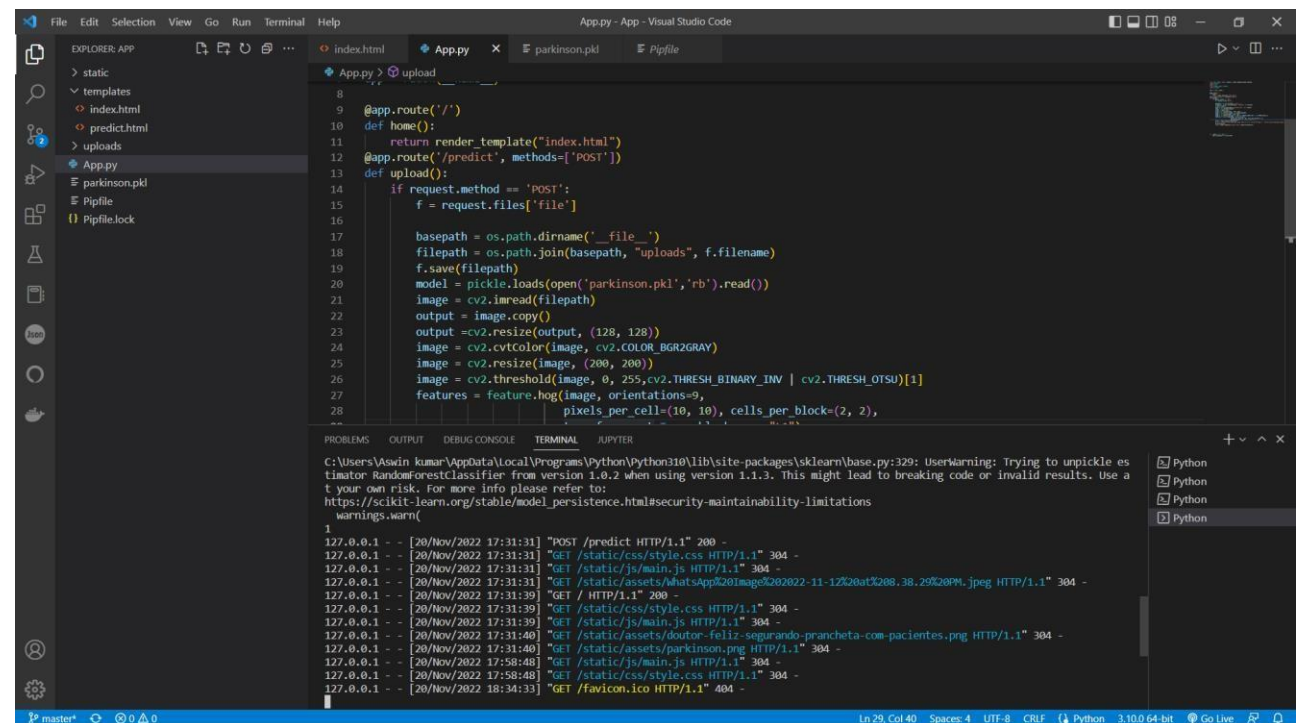
Out[9]: -1

## 7.2 Feature 2

**We have created an Application with Login/SignUp page, Home Page and Predict Page.**



```
1 from flask import Flask, request, render_template, make_response
2 import pickle
3 import cv2
4 from skimage import feature
5 import os.path
6
7 app = Flask(__name__)
8
9 @app.route('/')
10 def home():
11     return render_template("index.html")
12 @app.route('/predict', methods=['POST'])
13 def upload():
14     if request.method == 'POST':
15         f = request.files['file']
16
17         basepath = os.path.dirname(__file__)
18         filepath = os.path.join(basepath, "uploads", f.filename)
19         f.save(filepath)
20         model = pickle.loads(open('parkinson.pkl', 'rb').read())
21         image = cv2.imread(filepath)
22         output = image.copy()
23         output = cv2.resize(output, (128, 128))
24         image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
25         image = cv2.resize(image, (200, 200))
26         image = cv2.threshold(image, 0, 255, cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]
27         features = feature.hog(image, orientations=9,
28                               pixels_per_cell=(10, 10), cells_per_block=(2, 2),
29                               transform_sqrt=True, block_norm="L1")
30
31         preds = model.predict([features])
32         ls = ["../static/assets/WhatsApp Image 2022-11-12 at 8.49.29 PM.jpeg", "../static/assets/WhatsApp Image 2022-11-12 at 8.38.2"]
33         print(preds[0])
34         return render_template("predict.html", image=ls[preds[0]])
35
36
37
38
39
40 if __name__ == '__main__':
```



```
C:\Users\Awin kumar\AppData\Local\Programs\Python\Python310\lib\site-packages\sklearn\base.py:329: UserWarning: Trying to unpickle estimator RandomForestClassifier from version 1.0.2 when using version 1.1.3. This might lead to breaking code or invalid results. Use at your own risk. For more info please refer to: https://scikit-learn.org/stable/model_persistence.html#security-maintainability-limitations
  warnings.warn(
1
127.0.0.1 - - [20/Nov/2022 17:31:31] "POST /predict HTTP/1.1" 200 -
127.0.0.1 - - [20/Nov/2022 17:31:31] "GET /static/css/style.css HTTP/1.1" 304 -
127.0.0.1 - - [20/Nov/2022 17:31:31] "GET /static/js/main.js HTTP/1.1" 304 -
127.0.0.1 - - [20/Nov/2022 17:31:31] "GET /static/assets/WhatsAppImage0202022-11-12020at0200.38.29020PM.jpeg HTTP/1.1" 304 -
127.0.0.1 - - [20/Nov/2022 17:31:39] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [20/Nov/2022 17:31:39] "GET /static/css/style.css HTTP/1.1" 304 -
127.0.0.1 - - [20/Nov/2022 17:31:39] "GET /static/js/main.js HTTP/1.1" 304 -
127.0.0.1 - - [20/Nov/2022 17:31:40] "GET /static/assets/doutor-feliz-segurando-prancheta-com-pacientes.png HTTP/1.1" 304 -
127.0.0.1 - - [20/Nov/2022 17:58:48] "GET /static/assets/parkinson.png HTTP/1.1" 304 -
127.0.0.1 - - [20/Nov/2022 17:58:48] "GET /static/js/main.js HTTP/1.1" 304 -
127.0.0.1 - - [20/Nov/2022 17:58:48] "GET /static/css/style.css HTTP/1.1" 304 -
127.0.0.1 - - [20/Nov/2022 18:34:33] "GET /favicon.ico HTTP/1.1" 404 -
```




Home Page :

[Home](#) [Info](#) [Predict](#)

WELCOME TO MY IBM PROJECT 🧡

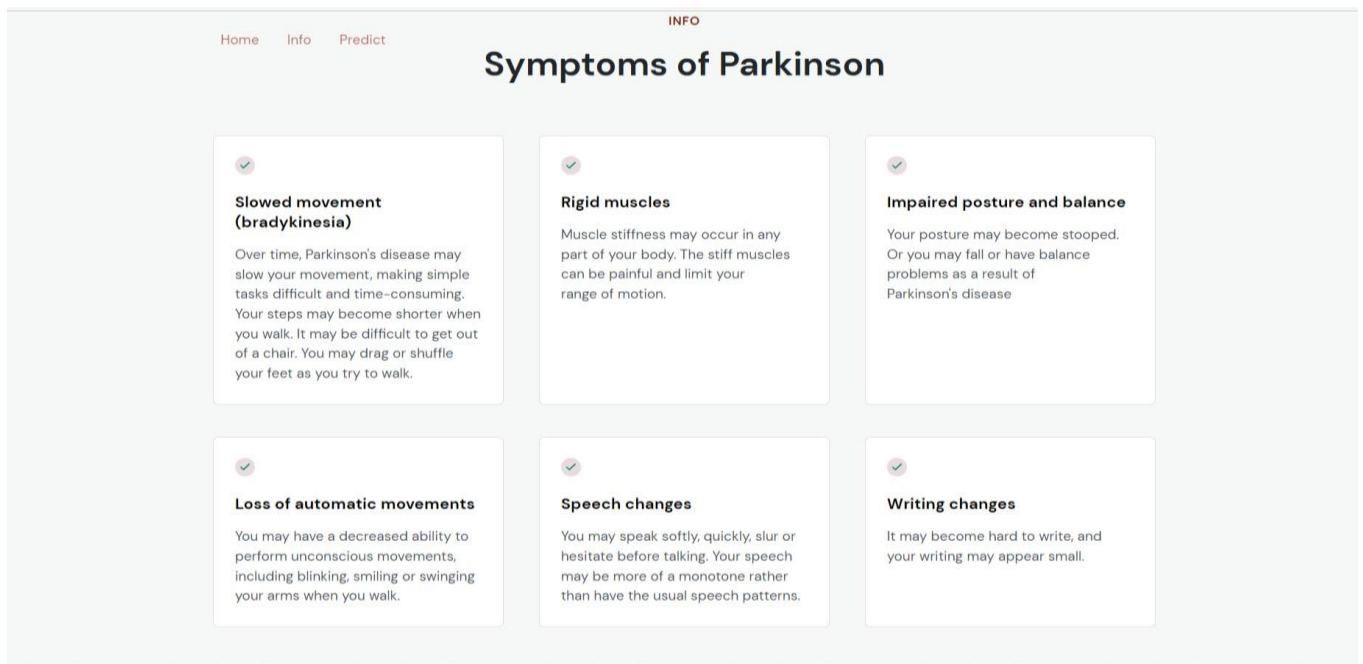
# Parkinson Disease prediction using ML

Parkinson's disease is a progressive disorder that affects the nervous system and the parts of the body controlled by the nerves. 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.

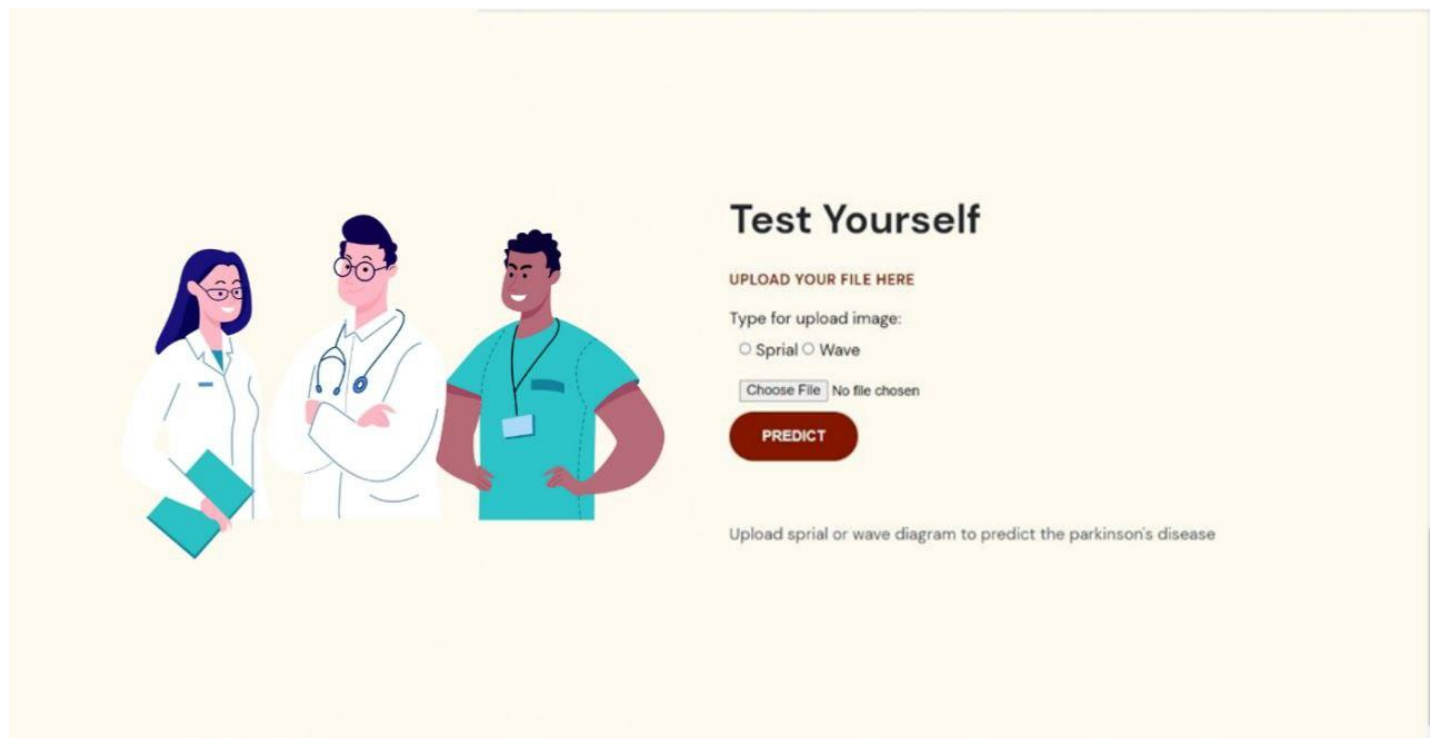


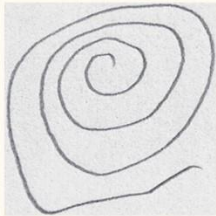
<b>9.4M PD</b> Parkinson's disease world 2017-22 percentage	<b>6.4M PD</b> Asia 2017-2022 percentage	<b>6.0PD</b> India 2017-2022 percentage
--	---	--

## INFO PAGE:



## Predict Page :





## Test Yourself

UPLOAD YOUR FILE HERE

V11HE01.png

**PREDICT**

Upload spiral or wave diagram to predict the parkinson's disease

## Your result



**No Parkinson Detected**



## Test Yourself

UPLOAD YOUR FILE HERE

V04PE01.png

**PREDICT**

Upload spiral or wave diagram to predict the parkinson's disease

## Your result



# 8.TESTING

## 8.1 Performance Metrics

### Classification Model : Confusion Matrix, Accuracy Score & Classification Report

#### Model Evaluation

```
In [12]: predictions = model.predict(X_test)
         predictions

Out[12]: array([0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1,
                1, 1, 1, 0, 0, 1, 1, 1], dtype=int64)

In [13]: cm = confusion_matrix(y_test, predictions)
         cm

Out[13]: array([[14,  1],
                [ 3, 12]], dtype=int64)

In [14]: accuracy = accuracy_score(y_test, predictions)
         accuracy

Out[14]: 0.8666666666666667

In [15]: cr = classification_report(y_test, predictions)
         print(cr)
```

	precision	recall	f1-score	support
0	0.82	0.93	0.87	15
1	0.92	0.80	0.86	15
accuracy			0.87	30
macro avg	0.87	0.87	0.87	30
weighted avg	0.87	0.87	0.87	30

## b. Hyperparameter Tuning

```
In [7]: from sklearn.model_selection import GridSearchCV

In [22]: model = RandomForestClassifier()

In [23]: parameters = {
    'max_depth' : [5,10,20,30,35],
    'random_state' : [0,1,2,3,4],
    'n_estimators' : [70,100,80,85,110]
}

In [24]: grid = GridSearchCV(model,parameters,cv=5)

In [25]: grid.fit(X_train, y_train)
Out[25]: GridSearchCV(cv=5, estimator=RandomForestClassifier(),
    param_grid={'max_depth': [5, 10, 20, 30, 35],
    'n_estimators': [70, 100, 80, 85, 110],
    'random_state': [0, 1, 2, 3, 4]})

In [26]: grid.best_params_
Out[26]: {'max_depth': 5, 'n_estimators': 100, 'random_state': 2}

In [27]: grid.best_estimator_
Out[27]: RandomForestClassifier(max_depth=5, random_state=2)

In [28]: grid.best_score_
Out[28]: 0.7923809523809524

In [29]: from sklearn.model_selection import RandomizedSearchCV
    rs = RandomizedSearchCV(model,parameters,cv=5)
    rs.fit(X_train, y_train)
Out[29]: RandomizedSearchCV(cv=5, estimator=RandomForestClassifier(),
    param_distributions={'max_depth': [5, 10, 20, 30, 35],
    'n_estimators': [70, 100, 80, 85, 110],
    'random_state': [0, 1, 2, 3, 4]})

In [30]: rs.best_params_
Out[30]: {'random_state': 2, 'n_estimators': 100, 'max_depth': 20}

In [31]: rs.best_score_
Out[31]: 0.7780952380952382
```

## 9.ADVANTAGES & DISADVANTAGES

### a. Advantages

- Less time consuming
- More accuracy in the model
- Easily implemented

### b. Disadvantages

- Packages to be installed
- Data collection is difficult

## 10. CONCLUSION

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. Here, we presented included studies in a high-level summary, providing access to information including (a) machine learning methods that have been used in the diagnosis of PD and associated outcomes, (b) types of clinical, behavioral and biometric data that could be used for rendering more accurate diagnoses, (c) potential biomarkers for assisting clinical decision making, and (d) 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. Machine learning approaches therefore have the potential to provide clinicians with additional tools to screen, detect or diagnose PD.

## 11. FUTURE SCOPE

The model can be trained with enormous amount of data to improve the accuracy. We can also merge the voice dataset and train the model accordingly for higher productivity.

## 12. APPENDIX

### a. Source Code

**Parkinson's\_Disease\_Prediction.ipynb :**

```
In [7]: 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
```



## Loading Train Data and Test Data

```
In [8]: def load_split(path):
        imagePaths = list(paths.list_images(path))
        data = []
        labels = []

        for imagePath in imagePaths:
            label = imagePath.split(os.path.sep)[-2]

            image = cv2.imread(imagePath)
            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)

            data.append(features)
            labels.append(label)

        return (np.array(data), np.array(labels))
```

## Load the train and test data

```
In [9]: print("[INFO] loading data...")
        (X_train, y_train) = load_split(trainingpath)
        (X_test, y_test) = load_split(testingpath)

        [INFO] loading data...
```

## Label Encoding

```
In [10]: le = LabelEncoder()
          y_train = le.fit_transform(y_train)
          y_test = le.transform(y_test)
          print(X_train.shape, y_train.shape)

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

## Model Building

### Training The Model

```
In [11]: print("[INFO] training model")
model = RandomForestClassifier(n_estimators=100)
model.fit(X_train, y_train)

[INFO] training model

Out[11]: RandomForestClassifier()
```

### Testing The Model

```
In [12]: testingpath=list(paths.list_images(testingpath))
idxs=np.arange(0,len(testingpath))
idxs=np.random.choice(idxs,size=(25,),replace=False)
images=[]

In [13]: for i in idxs:
          image=cv2.imread(testingpath[i])
          output=image.copy()

          # Load the input image,convert to grayscale and resize
```

## Model Evaluation

```
In [57]: predictions = model.predict(X_test)
predictions

Out[57]: array([0, 1, 1, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 1, 1, 0, 1,
        1, 0, 1, 1, 0, 1, 0, 1], dtype=int64)

In [58]: cm = confusion_matrix(y_test, predictions)
cm

Out[58]: array([[11,  4],
        [ 6,  9]], dtype=int64)

In [59]: accuracy = accuracy_score(y_test, predictions)
accuracy

Out[59]: 0.6666666666666666

In [60]: cr = classification_report(y_test, predictions)
print(cr)
```

	precision	recall	f1-score	support
0	0.65	0.73	0.69	15
1	0.69	0.60	0.64	15
accuracy			0.67	30
macro avg	0.67	0.67	0.67	30
weighted avg	0.67	0.67	0.67	30

## Save The Model

```
In [62]: pickle.dump(model,open('parkinson.pkl','wb'))
```

## Deployment

In [16]: !pip install -U ibm-watson-machine-learning

```
Requirement already satisfied: ibm-watson-machine-learning in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (1.0.257)
Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (0.8.9)
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (1.26.7)
Requirement already satisfied: packaging in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (21.3)
Requirement already satisfied: importlib-metadata in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (4.8.2)
Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (2.26.0)
Requirement already satisfied: pandas<1.5.0,>=0.24.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (1.3.4)
Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (2022.9.24)
Requirement already satisfied: ibm-cos-sdk==2.11.* in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (2.11.0)
Requirement already satisfied: lmond in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-watson-machine-learning) (0.3.3)
Requirement already satisfied: ibm-cos-sdk-core==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm-watson-machine-learning) (2.11.0)
Requirement already satisfied: ibm-cos-sdk-s3transfer==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm-watson-machine-learning) (2.11.0)
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm-watson-machine-learning) (0.10.0)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk-core==2.11.0->ibm-cos-sdk==2.11.*->ibm-watson-machine-learning) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas<1.5.0,>=0.24.2->ibm-watson-machine-learning) (2021.3)
Requirement already satisfied: numpy>=1.17.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas<1.5.0,>=0.24.2->ibm-watson-machine-learning) (1.20.3)
Requirement already satisfied: six>=1.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from python-dateutil<3.0.0,>=2.1->ibm-cos-sdk-core==2.11.0->ibm-cos-sdk==2.11.*->ibm-watson-machine-learning) (1.15.0)
Requirement already satisfied: charset-normalizer==2.0.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests->ibm-watson-machine-learning) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests->ibm-watson-machine-learning) (3.3)
Requirement already satisfied: zipp>=0.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from importlib-metadata->ibm-watson-machine-learning) (3.6.0)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from packaging->ibm-watson-machine-learning) (3.0.4)
```

In [17]: # Now connect notebook ml service with api key and url

```
from ibm_watson_machine_learning import APIClient
import json
import numpy as np
```

### Authenticate and Set Space

In [18]: wml\_credentials = {  
 "apikey": "RYa2JTvIsfgzBubvFmcYVUXLBDntmTWzC9KG5tjRtCS",  
 "url": "https://us-south.ml.cloud.ibm.com" #For Dallas region  
}

In [19]: wml\_client = APIClient(wml\_credentials)

In [20]: # Check the available deployments

```
wml_client.spaces.list()
```

Note: 'limit' is not provided. Only first 50 records will be displayed if the number of records exceed 50

ID	NAME	CREATED
efa48345-def9-4aa5-b19f-4dd7d5f766ce	ParkinsonDiseaseDetection	2022-11-06T10:09:49.894Z

In [21]: SPACE\_ID = "efa48345-def9-4aa5-b19f-4dd7d5f766ce"

Show desktop

```
In [22]: # Space id created default one

wml_client.set.default_space(SPACE_ID)
```

```
Out[22]: 'SUCCESS'
```

```
In [23]: # To check the environment
```

```
wml_client.software_specifications.list()

autoai-k01-c22-z-py3.9 1230009a-5011-5e8d-972a-0291688cc140 base
runtime-22.1-py3.9 12b83a17-24d8-5082-900f-0ab31fbfd3cb base
scikit-learn_0.22-py3.6 154010fa-5b3b-4ac1-82af-4d5ee5abbc85 base
default_r3.6 1b70aec3-ab34-4b87-8aa0-a4a3c8296a36 base
pytorch-onnx_1.3-py3.6 1bc6029a-cc97-56da-b8e0-39c3880dbbe7 base
kernel-spark3.3-r3.6 1c9e5454-f216-59dd-a20e-474a5cdf5988 base
pytorch-onnx_rt22.1-py3.9-edt 1d362186-7ad5-5b59-8b6c-9d0880bde37f base
tensorflow_2.1-py3.6 1eb25b84-d6ed-5dde-b6a5-3fbd16656666 base
spark-mllib_3.2 20047f72-0a98-58c7-9ff5-a77b012eb8f5 base
tensorflow_2.4-py3.8-horovod 217c16f6-178f-56bf-824a-b19f20564c49 base
runtime-22.1-py3.9-cuda 26215f05-08c3-5a41-a1b0-da66306ce658 base
do_py3.8 295addb5-9ef9-547e-9bf4-92ae3563e720 base
autoai-ts_3.8-py3.8 2aa0c932-798f-5ae9-abd6-15e0c2402fb5 base
tensorflow_1.15-py3.6 2b73a275-7cbf-420b-a912-eae7f436e0bc base
kernel-spark3.3-py3.9 2b7961e2-e3b1-5a8c-a491-a82c8368839a base
pytorch_1.2-py3.6 2c8ef57d-2687-4b7d-acce-01f94976dac1 base
spark-mllib_2.3 2e51f700-bca0-4b0d-88dc-5c6791338875 base
pytorch-onnx_1.1-py3.6-edt 32983cea-3f32-4400-8965-dde874a8d67e base
spark-mllib_3.0-py37 36507ebe-8770-55ba-ab2a-eafe787600e9 base
spark-mllib_2.4 390d21f8-e58b-4fac-9c55-d7ceda621326 base
```

```
In [22]: # Space id created default one
```

```
wml_client.set.default_space(SPACE_ID)
```

```
Out[22]: 'SUCCESS'
```

```
In [23]: # To check the environment
```

```
wml_client.software_specifications.list()

autoai-k01-c22-z-py3.9 1230009a-5011-5e8d-972a-0291688cc140 base
runtime-22.1-py3.9 12b83a17-24d8-5082-900f-0ab31fbfd3cb base
scikit-learn_0.22-py3.6 154010fa-5b3b-4ac1-82af-4d5ee5abbc85 base
default_r3.6 1b70aec3-ab34-4b87-8aa0-a4a3c8296a36 base
pytorch-onnx_1.3-py3.6 1bc6029a-cc97-56da-b8e0-39c3880dbbe7 base
kernel-spark3.3-r3.6 1c9e5454-f216-59dd-a20e-474a5cdf5988 base
pytorch-onnx_rt22.1-py3.9-edt 1d362186-7ad5-5b59-8b6c-9d0880bde37f base
tensorflow_2.1-py3.6 1eb25b84-d6ed-5dde-b6a5-3fbd16656666 base
spark-mllib_3.2 20047f72-0a98-58c7-9ff5-a77b012eb8f5 base
tensorflow_2.4-py3.8-horovod 217c16f6-178f-56bf-824a-b19f20564c49 base
runtime-22.1-py3.9-cuda 26215f05-08c3-5a41-a1b0-da66306ce658 base
do_py3.8 295addb5-9ef9-547e-9bf4-92ae3563e720 base
autoai-ts_3.8-py3.8 2aa0c932-798f-5ae9-abd6-15e0c2402fb5 base
tensorflow_1.15-py3.6 2b73a275-7cbf-420b-a912-eae7f436e0bc base
kernel-spark3.3-py3.9 2b7961e2-e3b1-5a8c-a491-a82c8368839a base
pytorch_1.2-py3.6 2c8ef57d-2687-4b7d-acce-01f94976dac1 base
spark-mllib_2.3 2e51f700-bca0-4b0d-88dc-5c6791338875 base
pytorch-onnx_1.1-py3.6-edt 32983cea-3f32-4400-8965-dde874a8d67e base
spark-mllib_3.0-py37 36507ebe-8770-55ba-ab2a-eafe787600e9 base
spark-mllib_2.4 390d21f8-e58b-4fac-9c55-d7ceda621326 base
```

## Save and Deploy the Model

```
In [24]: import sklearn
sklearn.__version__

Out[24]: '1.0.2'
```

```
In [25]: MODEL_NAME = "ParkinsonDiseaseDetection_DeployedModel"
DEPLOYMENT_NAME = "ParkinsonDiseaseDetection"
```

```
In [26]: # Set Python default version
```

```
software_spec_uid = wml_client.software_specifications.get_id_by_name('runtime-22.1-py3.9')
```

## Create Model Properties to deploy the model

```
In [27]: # Setup Model Meta
```

```
model_props = {
    wml_client.repository.ModelMetaNames.NAME: MODEL_NAME,
    wml_client.repository.ModelMetaNames.TYPE: 'scikit-learn 1.0',
    wml_client.repository.ModelMetaNames.SOFTWARE_SPEC_UID: software_spec_uid
}
```

In [28]: # Save Model

```
model_details = wml_client.repository.store_model(  
    model = model,  
    meta_props = model_props,  
    training_data = X_train,  
    training_target = y_train  
)
```

In [29]: model\_details

```
{  
  'schemas': {'input': [{'fields': [{'name': 'f0', 'type': 'float'},  
    {'name': 'f1', 'type': 'float'},  
    {'name': 'f2', 'type': 'float'},  
    {'name': 'f3', 'type': 'float'},  
    {'name': 'f4', 'type': 'float'},  
    {'name': 'f5', 'type': 'float'},  
    {'name': 'f6', 'type': 'float'},  
    {'name': 'f7', 'type': 'float'},  
    {'name': 'f8', 'type': 'float'},  
    {'name': 'f9', 'type': 'float'},  
    {'name': 'f10', 'type': 'float'},  
    {'name': 'f11', 'type': 'float'},  
    {'name': 'f12', 'type': 'float'},  
    {'name': 'f13', 'type': 'float'},  
    {'name': 'f14', 'type': 'float'},  
    {'name': 'f15', 'type': 'float'},  
    {'name': 'f16', 'type': 'float'},  
    {'name': 'f17', 'type': 'float'},  
    {'name': 'f18', 'type': 'float'},  
    {'name': 'f19', 'type': 'float'}]}]}
```

In [30]: model\_id = wml\_client.repository.get\_model\_id(model\_details)  
model\_id

Out[30]: '7d936b97-a55f-403a-9624-5ad06e18e6b0'

### Deploy in props

In [31]: # Set meta

```
deployment_props = {  
    wml_client.deployments.ConfigurationMetaNames.NAME : DEPLOYMENT_NAME,  
    wml_client.deployments.ConfigurationMetaNames.ONLINE : {}  
}
```

In [32]: # Deploy

```
deployment = wml_client.deployments.create(  
    artifact_uid = model_id,  
    meta_props = deployment_props  
)
```

```
#####  
Synchronous deployment creation for uid: '7d936b97-a55f-403a-9624-5ad06e18e6b0' started  
#####  
  
initializing  
Note: online_url is deprecated and will be removed in a future release. Use serving_urls instead.  
ready  
  
-----  
Successfully finished deployment creation, deployment_uid='cbe26007-da09-4ca5-919f-3b00aa88f433'
```

**App.py :**

```

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

@app.route('/') def
home():
    return
render_template("index.html")
@app.route('/predict',
methods=['POST']) def upload():    if
request.method == 'POST':        f =
request.files['file']
        basepath = os.path.dirname('__file__')
filepath = os.path.join(basepath, "uploads", f.filename)
        f.save(filepath)        model =
pickle.loads(open('parkinson.pkl','rb').read())        image = cv2.imread(filepath)
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 = 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])
        ls = ["../static/assets/WhatsApp Image 2022-11-12 at 8.49.29
PM.jpeg","../static/assets/WhatsApp Image 2022-11-12 at 8.38.29 PM.jpeg"]
print(preds[0])
        return render_template("predict.html",image=ls[preds[0]])

if __name__ == "__main__":
    app.run(debug=False,port=5555)

```

**style.css :**

```
/*=== GERAL =====*/
```

```
}

:root {
  --hue: 10;
  --primary-color: hsl(var(--hue), 100%, 26%);
  --headline: hsl(210, 11%, 15%);
  --paragraph: hsl(210, 9%, 31%);

  --brand-beige: hsl(39, 100%, 97%);
  --brand-light: hsl(calc(var(--hue) - 22), 23%, 89%);
  --brand-light-2: hsl(calc(var(--hue) + 10), 14%, 97%);
  --brand-dark: hsl(var(--hue), 100%, 14%);
  --bg-light: hsl(180, 14%, 97%);
  font-size: 62.5%; /* 1rem = 10px
*/
  --nav-height: 7.2rem;
}
html
{
  /* chrome://flags/#smooth-scrolling */  scroll-
behavior: smooth;
}
html, body
{
  width:
100%;
height: 100%;
}
body {  font-family:
'DM Sans';
```



```
* {  margin: 0;
```

```
padding: 0;  box-sizing:
```

```
border-box;  font-size:
```

```
1.6rem;
```







```
text-align: center;

overflow: overlay;
background-color: var(--bg-
light);
}
.wrapper { width:
min(50rem, 100%);
margin-inline: auto;
padding-inline: 2.4rem;
```



```
list-style: none;
} img { max-
width: 100%;
} section {
padding-block: 10rem;
} section header h4 {
font-size: 1.4rem; font-
weight: 700; line-height:
150%; letter-spacing:
0.08rem; color: var(--
primary-color);
text-transform:
uppercase;
margin-bottom:
1.6rem;
} section header h2
{ font-size: 3rem;
line-height: 3.9rem;

color: var(--headline);
} section header h1 {
font-size: 3.4rem;
color: var(--headline);
```



```
}  ul {    line-height:
```

```
130%;    margin-bottom:
```

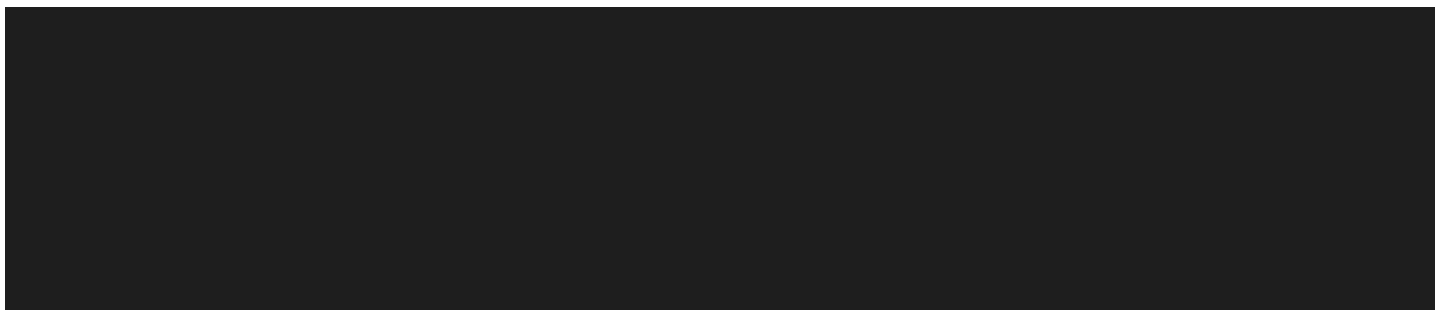
```
2.4rem;
```

```
}  section .content  
p {    font-weight:  
400;  font-size:  
1.6rem;  line-  
height: 150%;  
  
    color: var(--paragraph);  
}
```





```
.button  
{  
  background: var(--primary-color);
```



```
border: none; border-
radius: 4rem;
margin-bottom:
6rem;
padding: 1.6rem
3.2rem; width: fit-
content;
color: white;
font-size: 1.4rem;
font-weight: 700;
text-transform: uppercase;
display: flex;
align-items: center;
justify-content: center;
gap: 1.6rem;
text-decoration:
none;
transition: background
200ms;
}

.button:hover {
background-color: var(--brand-dark);
}

.social-links {
display: flex; align-
items: center; justify-
content: center; gap:
3.2rem;
}
```



```
/* custom colors */
#services .card circle {
  fill: var(--brand-light);
}

#navigation .logo path[fill*='#00856F'],
#backToTopButton circle {
  fill: var(--primary-color);
} button.open-menu
path[stroke*='#00856F'],
#contact li path { stroke:
var(--primary-color);
}
```



```
body.menu-expanded #navigation .logo path,  
#navigation.scroll .logo path {  
fill: white;  
}  
  
#navigation.scroll button.open-menu path[stroke*='#00856F'] {  
stroke: white;  
}  
  
/*=== NAVIGATION  
=====*/ nav {  
display: flex;  
height: var(--nav-  
height);  
position:  
fixed; top: 0;  
width: 100vw;  
z-index:  
100;  
} nav .wrapper {  
display: flex;  
align-items: center;  
justify-content: space-between;  
}  
nav.scroll { background: var(--  
primary-color);  
} nav  
button {
```



```
background: none;  
border: none;  cursor:  
pointer;  
}  nav .menu, nav  
.close-menu {  
position: fixed;
```







}





/\*=== MENU-EXPANDED =====\*/



```
body.menu-expanded {
overflow: hidden;
} body.menu-expanded >
:not(nav) { visibility:
hidden;
}

.menu, .close-menu, body.menu-
expanded .open-menu {
opacity: 0;
visibility: hidden;
} body.menu-expanded .menu,
body.menu-expanded .close-menu
{ opacity: 1;
visibility: visible;
}
.menu { transform:
translateY(100%);
} body.menu-expanded .menu {
top: 0; left: 0; background:
var(--primary-color);

width: 100vw;
height: 100vh;

padding-top: var(--nav-height);
transition: transform
300ms; transform:
translateY(0); }
```

```
.menu ul:nth-child(1) {
display: flex;  flex-
direction: column;  gap:
4.8rem;  margin-top:
6rem; font-weight: 700;
  font-size: 2.4rem;  line-
height: 3.1rem;
}
.menu ul li a {
color: white;  text-
decoration: none;
```

```
}

.menu .button {
background: white;
border-radius: 4rem;
font-weight: 700; font-
size: 1.8rem; line-
height: 2.3rem; text-
transform: uppercase;
text-decoration: none;
  color: var(--primary-
color);
  display: inline-
block; padding: 16px
32px;
  margin-top:
4.8rem; margin-
bottom: 8rem;
}

.menu .button:hover { background-
color: var(--primary-color); color:
white;

  filter: brightness(1.3);
}
body.menu-expanded .logo,
body.menu-expanded button {
position: relative; z-
index: 100;
}
```

```
body.menu-expanded .logo path {
  fill: white;
}
body.menu-expanded button
path {  stroke: white;
}
/*=== #HOME =====*/
#home {  padding-block: 0;  padding-top:
calc(4.1rem + var(--nav-height));
}
#home::before
{  content:
'';
```

```
width: 100%; height: calc(76% +
var(--nav-height)); background-color:
var(--brand-light); display: block;
position:
absolute; top: 0;
left: 0; z-index:
-1;
}
```

```
#home .button { margin-
inline: auto;
}
```

```
#home p { font-size:
1.8rem; line-height:
150%; font-weight: 400;
color: var(--paragraph);
margin-bottom:
3.2rem;
}
```

```
#home img { width:
26.4rem; display:
block; margin-inline:
auto; object-position:
0 2rem;
}
```

```
#home .stats {
width: 100%;
```

```
padding-block: 4rem; margin-
inline: auto;
background-color: var(--brand-
beige); border: 1px solid var(--
brand-light); border-radius: 0.6rem;
display: flex;
flex-direction:
column;
justify-content: center;
gap: 6rem;
}
#home .stat h3 { font-
size: 4.8rem; color:
var(--headline);
```



```
    line-height: 130%;
    margin-bottom:
0.4rem;
}

#home .stat p {  margin: 0;
color: var(--primary-color);
font-size: 1.6rem;  line-
height: 150%;
}

/*=== SERVICES =====*/
#services header h2 {  margin-
bottom: 6rem;
}

#services .cards {
display: flex;  flex-
direction: column;  gap:
3.2rem;
}

#services .card {
padding: 2.4rem;  text-
align: left;

    background: white;
    border: 1px solid var(--brand-
light);  border-radius: 0.6rem;
}
```

```
#services .card h3 {  margin-
block: 1.6rem;
}

/*=== ABOUT =====*/
#about { text-align: left; background-
color: var(--brand-beige);
}
#about header h2 {
margin-bottom: 2.4rem;
}
#about .content p {
margin-top: 2.4rem;
```

```
    margin-bottom: 6rem;
}

/*=== CONTACT =====*/
#contact {
    text-align: left;
}

#contact header {
    margin-bottom: 3.2rem;
}

#contact ul {
    display: flex;
    flex-direction: column;
    gap: 1.6rem;
    margin-bottom: 3.2rem;
}

#contact ul li {
    display: flex;
    align-items: center;
    gap: 0.8rem;
}

/*=== FOOTER =====*/
footer {
    background-color: var(--primary-color);
    padding-block: 6rem;

    text-align: left;
```



```
    color: white; }  
footer .logo { display:  
inline-block; margin-  
bottom: 2.4rem;  
}  
footer .logo svg {  
width: 23.6rem; height:  
3.1rem;  
} footer .logo  
path { fill:  
white; }
```



```
    footer p {    color: var(--
brand-beige);    line-
height: 2;
        margin-bottom:
3.2rem;
    }    footer .social-
links {
        justify-content: flex-start;
    }
```

```
/* BACK TO TOP */
#backToTopButton {
position: fixed;
bottom: 1rem;    right:
2.5rem;
    opacity:
0;
    visibility: hidden;
    transform:
translateY(100%);
transition: 200ms;
}
```

```
#backToTopButton.show {
opacity: 1;    visibility:
visible;    transform:
translateY(0);
}
```

```
/*=== RESPONSIVO =====*/
@media (min-width: 1024px) {
```

```
/*=== GERAL
=====*/ body {
overflow: auto; }

.wrapper { width:
min(112rem, 100%);
display: grid;
}

.col-a { grid-
area: A; }

.col-b { grid-
area: B;
```

```

    }    section {
padding-block: 16rem;
    }    section header
h2 {      font-size:
4rem;      line-height:
5.2rem;
    }

/*=== NAVIGATION =====*/
/* reset */
nav#navigation .wrapper * {
margin: 0;      padding: 0;
visibility: initial;
display: initial;
opacity: initial;      flex-
direction: initial;
position: initial;      font-
size: initial;      font-
weight: initial;
transform: initial;
color: initial;
    background-color:      initial;
filter: initial;
    }    nav#navigation .close-
menu,    nav#navigation .open-
menu,    nav#navigation .social-
links {      display: none;

```



```
}
```

```
nav#navigation .menu {  display:
flex;  align-items: center;
justify-content: space-between;
width: 60%;
}
nav#navigation .menu ul:nth-child(1) {
display: flex;  gap: 3.2rem; }

nav#navigation .menu ul li a {
color: var(--primary-color);
```





```
        opacity: 0.7;
    }    nav#navigation .menu
a.button {    display: flex;
justify-content: center;
align-items: center;
padding: 1rem 2.4rem;
        border: 1px solid var(--primary-
color);    border-radius: 4rem;
        color: var(--primary-
color);
        font-weight: 700;
font-size: 1.4rem;    line-
height: 1.8rem;    text-
transform: uppercase;
    }    nav#navigation .menu a.button:hover
{    background-color: var(--primary-
color);    border: none;    color:
white;
    }    nav#navigation.scroll .menu ul
li a {    color: var(--brand-light-
2);    opacity: 1;
    }
    nav#navigation.scroll .menu a.button {    border-
color: white;
```

```
        color: white;
    }

    nav#navigation .menu li a {
transition: opacity 0.4s;
    } nav#navigation .menu li

a.active, nav#navigation .menu

li a:hover {

    opacity: 1;    font-
weight: 700;
    }

    nav#navigation .menu li a::after {
content: '';    width: 0%;
```

```

        height: 2px;    background-color: var(--primary-color);
    position:
relative;    bottom:
-2rem;    left: -
0.5rem;
        display:
block;
        transition: width
0.5s;
    }    nav#navigation.scroll .menu li
a::after {    background-color: white;
    }    nav#navigation .menu li
a.active::after,    nav#navigation .menu
li a:hover::after {    padding-inline:
0.8rem;    width: 100%;
    }    nav#navigation.scroll .menu li
a.active,    nav#navigation.scroll .menu
li a:hover {    opacity: 1;
    }
    nav#navigation.scroll .menu
a.button.active,    nav#navigation.scroll
.menu a.button:hover {    background-color:
var(--primary-color);    filter:
brightness(1.3);    border: none;
    }

```

```
/*=== HOME =====*/
#home::before { height: calc(96% -
var(--nav-height)); }

#home { padding-top: var(--nav-
height); }

#home .wrapper { grid-template-
columns: 60.5rem 1fr; grid-
template-areas:
    'A B'
    'C C';
}
```

```
#home .col-a {      text-align: left;      align-self: center;
}

#home h1 {
  font-size: 5.2rem;
}

#home .content p {      font-size: 1.8rem;
}

#home .stats {      grid-area: C;
  flex-direction: row;
  padding: 6rem;
  gap: 0;
}

#home .stats .stat + .stat {      border-left: 1px solid var(--primary-color);
}

#home .stats .stat {
  flex: 1;
}

#home .button {
  margin: 0;
}
```

```
#home img {
width: 42rem;
}

/*=== SERVICES =====*/
#services h2 {
width: 47rem;
margin-inline: auto;
}

#services .cards {
flex-direction: row;
flex-wrap: wrap;
gap: 4rem; }
```

```
#services .card {
width: 30%;    flex-
grow: 1;
}

/*=== ABOUT =====*/
#about .wrapper {    grid-
template-columns: 48rem 1fr;
grid-template-areas: 'B A';    gap:
6.7rem;
}

#about .col-a {    align-
self: center;
}

#about .content p {    margin-
bottom: 0;
}

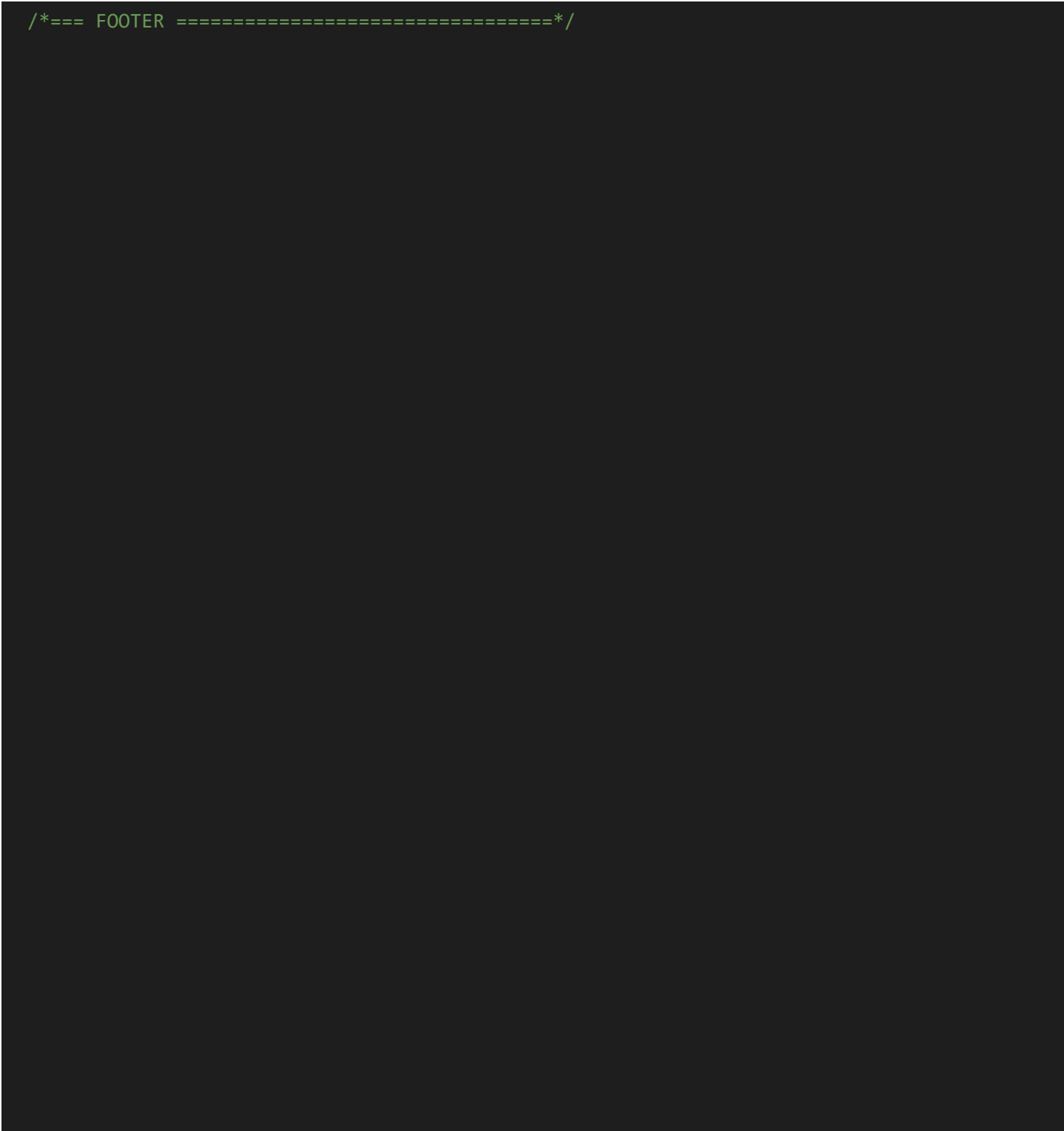
/*=== CONTACT =====*/

#contact .wrapper {
    grid-template-columns: 1fr 57.5rem;    grid-template-
areas: 'A B';
}

#contact h2 {
width: 40.4rem;
}
```



/\*== FOOTER =====\*/







```
footer {  
  padding-block: 8rem;  
} footer  
.wrapper {  
  grid-template-columns: 1fr 1fr;  
  grid-template-areas: 'A B';  
}
```



```
    footer .col-a {  
align-self: center;  
    }  
  
    footer .col-b {      align-  
self: center;      justify-  
self: flex-end;  
    }  
footer p {  
    margin-bottom: 0  
    }  
}
```

Main.js

```
window.addEventListener('scroll', onScroll)
onScroll() function
onScroll() {
  showNavOnScroll()
  showBackToTopButtonOnScroll()
  activateMenuAtCurrentSection(home)
  activateMenuAtCurrentSection(services)
  activateMenuAtCurrentSection(about)
  activateMenuAtCurrentSection(contact)
} function
activateMenuAtCurrentSection(section) {  const
targetLine = scrollY + innerHeight / 2
  // verificar se a seção passou da linha
  // quais dados vou precisar?
  const sectionTop = section.offsetTop  const sectionHeight =
section.offsetHeight  const sectionTopReachOrPassedTargetline =
targetLine >= sectionTop
  // verificar se a base está abaixo da linha alvo

  const sectionEndsAt = sectionTop + sectionHeight  const
sectionEndPassedTargetline = sectionEndsAt <= targetLine
  // limites da seção  const sectionBoundaries =
sectionTopReachOrPassedTargetline && !sectionEndPassedTargetline
```

```

    const sectionId = section.getAttribute('id')    const menuElement =
document.querySelector(`.menu a[href*=${sectionId}]`)

menuElement.classList.remove('active')
if (sectionBoundaries) {
menuElement.classList.add('active')
}
} function showNavOnScroll() {    if
(scrollY > 0) {
navigation.classList.add('scroll')
    } else {
navigation.classList.remove('scroll')
    }
} function
showBackToTopButtonOnScroll() {    if
(scrollY > 550) {
backToTopButton.classList.add('show')
    } else {
backToTopButton.classList.remove('show')
    }
} function openMenu() {
document.body.classList.add('menu-expanded')
} function closeMenu() {
document.body.classList.remove('menu-expanded')
}

ScrollReveal({
origin: 'top',
distance: '30px',
duration: 700
}).reveal(`
    #home,
    #home img,
    #home .stats,
    #services,
    #services header,
    #services .card
#about,
    #about header,
    #about .content`)

```

```
function imageChange(event){    let element =
document.getElementById("image")
    element.src = URL.createObjectURL(event.target.files[0])
}
```

```
1  $(document).ready(function () {
2      // Init
3      $('.image-section').hide();
4      $('.loader').hide();
5      $('#result').hide();
6
7      // Upload Preview
8      function readURL(input) {
9          if (input.files && input.files[0]) {
10             var reader = new FileReader();
11             reader.onload = function (e) {
12                 $('#imagePreview').css('background-image', 'url(' + e.target.result + ')');
13                 $('#imagePreview').hide();
14                 $('#imagePreview').fadeIn(650);
15             }
16             reader.readAsDataURL(input.files[0]);
17         }
18     }
19     $("#imageUpload").change(function () {
20         $('.image-section').show();
21         $('#btn-predict').show();
22         $('#result').text('');
23         $('#result').hide();
24         readURL(this);
25     });
26
27     // Predict
28     $('#btn-predict').click(function () {
29         var form_data = new FormData($('#upload-file')[0]);
30
31         // Show loading animation
32         $(this).hide();
33         $('.loader').show();
34     });
```

```

67     .submit-btn{
68         width: 85%;
69         padding: 10px 30px;
70         cursor: pointer;
71         display: block;
72         margin: auto;
73         background: linear-gradient(to right, #4e4888,#7bc0c8);
74         border: 0;
75         outline: none;
76         border-radius: 30px;
77     }
78     .check-box{
79         margin: 30px 10px 30px 0;
80     }
81     span{
82         color: #777;
83         font-size: 12px;
84         bottom: 68px;
85         position: absolute;
86     }
87 }
88 #login{
89     left: 50px;
90 }
91 #register{
92     left: 450px;
93 }
94 .err{
95     color:rgb(198, 156, 243);
96     margin: 265px 0 0 145px;
97 }
25     width: 220px;
26     margin: 35px auto;
27     position: relative;
28     box-shadow: 0 0 20px 9px #5f97e51f;
29     border-radius: 40px;
34
35     // Make prediction by calling api /predict
36     $.ajax({
37         type: 'POST',
38         url: '/predict',
39         data: form_data,
40         contentType: false,
41         cache: false,
42         processData: false,
43         async: true,
44         success: function (data) {
45             // Get and display the result
46             $('#loader').hide();
47             $('#result').fadeIn(600);
48             $('#result').text('Prediction : '+data);
49             console.log('Success!');
50         },
51     });
52 });
53
54 });

```

**In TEMPLATE**

**folder**

**home.html :**

```

<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Parkinson Prediction using ML</title>

  <link rel="stylesheet" href="../static/css/style.css" />
  <link rel="preconnect" href="https://fonts.googleapis.com" />
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />
  <link href="https://fonts.googleapis.com/css2?family=DM+Sans:wght@400;700&display=swap"
rel="stylesheet" />
</head>

<body>
  <nav id="navigation">
    <div class="wrapper">

      <div class="menu">
        <ul>
          <li>
            <a onclick="closeMenu()" href="#home">Home</a>
          </li>
          <li><a onclick="closeMenu()" href="#services">Info</a></li>
          <li><a onclick="closeMenu()" href="#about">Predict</a></li>
        </ul>

        <!-- <a class="button" onclick="closeMenu()" href="#contact">Agende sua consulta</a>
-->

      </div>

      <button aria-expanded="false" aria-label="Abrir menu" onclick="openMenu()"
class="open-menu">
        <svg width="40" height="40" viewBox="0 0 40 40" fill="none"
xmlns="http://www.w3.org/2000/svg">
          <path d="M10 20H30" stroke="#00856F" stroke-width="2" stroke-linecap="round"
stroke-linejoin="round" />
          <path d="M10 12H30" stroke="#00856F" stroke-width="2" stroke-linecap="round"
stroke-linejoin="round" />

```



```

        <path d="M18 28L30 28" stroke="#00856F" stroke-width="2" stroke-linecap="round"
stroke-linejoin="round" />
    </svg>
</button>
<button aria-expanded="true" aria-label="Fechar menu" onclick="closeMenu()"
class="close-menu">
    <svg width="40" height="40" viewBox="0 0 40 40" fill="none"
xmlns="http://www.w3.org/2000/svg">
        <path d="M30 10L10 30M10 10L30 30" stroke="#FFFAF1" stroke-width="2"
stroke-linecap="round" stroke-linejoin="round" />
    </svg>
</button>
</div>
</nav>

<section id="home">
    <div class="wrapper">
        <div class="col-a">
            <header>
                <h4>Welcome to my IBM Project👋</h4>
                <h1>Parkinson Disease prediction using ML</h1>
            </header>

            <div class="content">
                <p>
                    Parkinson's disease is a progressive disorder that affects the nervous
system and the parts of the body controlled by the nerves. 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>

            </div>
        </div>

        <div class="col-b">
            
        </div>

        <div class="stats">
            <div class="stat">
                <h3>9.4M PD</h3>
                <p>Parkinson's disease world 2017-22 percentage</p>

```





</div>

```

        <div class="stat">
            <h3>6.4M PD</h3>
            <p>Asia 2017-2022 percentage</p>
        </div>

        <div class="stat">
            <h3>6.0PD</h3>
            <p>India 2017-2022 percentage</p>
        </div>
    </div>
</div>
</section>

<section id="services">
    <div class="wrapper">
        <header>
            <h4>Info</h4>
            <h2>Symptoms of Parkinson</h2>
        </header>

        <div class="content">
            <div class="cards">
                <div class="card">
                    <svg width="24" height="24" viewBox="0 0 24 24" fill="none"
xmlns="http://www.w3.org/2000/svg">
                        <circle cx="12" cy="12" r="12" fill="#DCE9E2" />
                        <path d="M17.091 8.18182L10.091 15.1818L6.90918 12" stroke="#00856F"
strokewidth="1.5" stroke-linecap="round" stroke-linejoin="round" />
                    </svg>

                    <h3>Slowed movement (bradykinesia)</h3>
                    <p>
                        Over time, Parkinson's disease may slow your movement, making simple tasks
                        difficult and time-consuming.
                        Your steps may become shorter when you walk. It may be difficult to get out
                        of a chair. You may drag or shuffle your feet as you try to walk.
                    </p>
                </div>

                <div class="card">
                    <svg width="24" height="24" viewBox="0 0 24 24" fill="none"
xmlns="http://www.w3.org/2000/svg">
                        <circle cx="12" cy="12" r="12" fill="#DCE9E2" />
                        <path d="M17.091 8.18182L10.091 15.1818L6.90918 12" stroke="#00856F"
strokewidth="1.5"

```



```
stroke-linecap="round" stroke-linejoin="round" />
```

```

</svg>

<h3>Rigid muscles</h3>
<p>
    Muscle stiffness may occur in any part of your body. The stiff muscles can
be painful and limit your                range of motion.
</p>
</div>

<div class="card">
    <svg width="24" height="24" viewBox="0 0 24 24" fill="none"
xmlns="http://www.w3.org/2000/svg">
        <circle cx="12" cy="12" r="12" fill="#DCE9E2" />
        <path d="M17.091 8.18182L10.091 15.1818L6.90918 12" stroke="#00856F"
strokewidth="1.5"                stroke-linecap="round" stroke-linejoin="round" />
    </svg>

    <h3>Impaired posture and balance</h3>
    <p>
        Your posture may become stooped. Or you may fall or have balance problems as a
result of
        Parkinson's disease
    </p>
</div>

<div class="card">
    <svg width="24" height="24" viewBox="0 0 24 24" fill="none"
xmlns="http://www.w3.org/2000/svg">
        <circle cx="12" cy="12" r="12" fill="#DCE9E2" />
        <path d="M17.091 8.18182L10.091 15.1818L6.90918 12" stroke="#00856F"
strokewidth="1.5"                stroke-linecap="round" stroke-linejoin="round" />
    </svg>

    <h3>Loss of automatic movements</h3>
    <p>
        You may have a decreased ability to perform unconscious movements,
including blinking, smiling or swinging                your arms when you walk.
    </p>
</div>

<div class="card">
    <svg width="24" height="24" viewBox="0 0 24 24" fill="none"
xmlns="http://www.w3.org/2000/svg">
        <circle cx="12" cy="12" r="12" fill="#DCE9E2" />

```



```
<path d="M17.091 8.18182L10.091 15.1818L6.90918 12" stroke="#00856F" stroke-
```

```

width="1.5"                stroke-linecap="round" stroke-
linejoin="round" />      </svg>

    <h3>Speech changes</h3>
    <p>
        You may speak softly, quickly, slur or hesitate before talking. Your
speech may be more of a monotone                rather than have the usual speech
patterns.
    </p>
</div>

    <div class="card">
        <svg width="24" height="24" viewBox="0 0 24 24" fill="none"
xmlns="http://www.w3.org/2000/svg">
            <circle cx="12" cy="12" r="12" fill="#DCE9E2" />
            <path d="M17.091 8.18182L10.091 15.1818L6.90918 12" stroke="#00856F"
strokewidth="1.5"                stroke-linecap="round" stroke-linejoin="round" />
        </svg>

        <h3>Writing changes</h3>
        <p>
            It may become hard to write, and your writing may appear small.
        </p>
    </div>
</div>
</section>

<section id="about">
    <div class="wrapper">
        <div class="col-a">
            <header>
                <h2>Test Yourself</h2>

                <h4> Upload your file here</h4>
                <!-- <form id="form-data" method="post" enctype="multipart/form-data"
action="/predict">

                    <legend>Type for upload image:</legend>
                    <div>
                        <input type="radio" id="imageChoice1" name="imageType" value="Spril" />
                        <label for="contactChoice1">Sprial</label>

                        <input type="radio" id="imageChoice2" name="imageType" value="Wave" />
                    <label for="contactChoice2">Wave</label>

```





```

        </div>

        <input type="file" id="file-input" onchange="imageChange(event)"/>
        <input type="submit" />
    </form> -->

    <!-- <button type="button" id="btn-predict" onclick="predict()">Predict</button>
->
    <form action = "http://localhost:5555/predict" method = "POST" id="form-data"
enctype = "multipart/form-data">
        <input type = "file" name = "file" onchange="imageChange(event)"
style="padding:10px"/>
        <input type = "submit" value="Predict" class="button"/>
    </form>

</header>

<div class="content">

    <p>
        Upload sprial or wave diagram to predict the parkinson's disease
    </p>
</div>
</div>

<div class="col-b">
    
</div>
</div>
</section>

<!-- ScrollReaveal Lib -->
<script src="https://unpkg.com/scrollreveal"></script>
<script src="../static/js/main.js"></script>
</body>
</html>

```

**prediction.html :**

```

<!DOCTYPE html>
<html lang="en">

<head>
  <meta charset="UTF-8" />
  <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  <title>Parkinson Prediction using ML</title>

  <link rel="stylesheet" href="../static/css/style.css" />
  <link rel="preconnect" href="https://fonts.googleapis.com" />
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin />
  <link href="https://fonts.googleapis.com/css2?family=DM+Sans:wght@400;700&display=swap"
rel="stylesheet" />
</head>

<body>
  <nav id="navigation">
    <div class="wrapper">

      <div class="menu">
        <ul>
          <li>
            <a onclick="closeMenu()" href="/">Home</a>
          </li>
          <li><a onclick="closeMenu()" href="/">Info</a></li>
          <li><a onclick="closeMenu()" href="/">Predict</a></li>
        </ul>
      </div>
    </div>
  </nav>
  <section>
    <div>
      <h2>Your result</h2>
      

    </div>
  </section>
  <!-- ScrollReveal Lib -->
  <script src="https://unpkg.com/scrollreveal"></script>
  <script src="../static/js/main.js"></script>

</body>
</html>

```

## **13. Reference Link**

- a. Github Link

[Git repository link](#)

- b. Project Demo Link

[Project Demo Video link](#)