DETECTING PARKINSON’S DISEASE USING MACHINE LEARNING

IBM-Project-8946-1658939044

NALAIYA THIRAN PROJECT BASED LEARNING ON PROFESSIONAL READLINESS FOR INNOVATION, EMPLOYNMENT AND ENTERPRENEURSHIP

A PROJECT REPORT

Submitted By

C.KARTHIKEYAN	-	921319205057
M.MATHESH	-	921319205071
B.RAJESH KUMAR	-	921319205109
A.LOKHU PRASANTH	-	921319205065

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

in

INFORMATION TECHNOLOGY

PSNA COLLEGE OF ENGINEERING AND TECHNOLOGY

DINDIGUL 624622

TABLES OF CONTENTS

Parkinson's Disease Detection

Project Report

Introduction

Project Overview:

Parkinson’s disease (PD) is a neurodegenerative movement disease where the symptoms gradually develop to start with a slight tremor in one hand and a feeling of stiffness in the body and it becomes worse over time.

At present there is no conclusive result for this disease by non-specialist clinicians, particularly in the early stage of the disease where identification of the symptoms is very difficult. The disease is majorly said to be affecting the individuals who are living in village areas with their respective ages between 40 and 50.

Parkinson's disease detection system has been designed to detect the Parkinson's disease in a patient given their hand drawn spiral or wave images. The system is built using fundamental concepts of Data analytics and Computer Vision that are trained to differentiate between healthy and Parkinson hand drawn images.

Purpose:

Lack of adequate knowledge poses a barrier in the provision of appropriate treatment and care for individuals with Parkinson’s Disease. Parkinson's disease affects over 6 million people worldwide. There is no proper testing procedure defined to detect the disease as the disease examination varies at different instances of the medical operation. We propose our model to detect the disease at very less error rate.

Literature Survey:

Existing problem:

Due to insufficient resources and awareness in underdeveloped countries, proper and timely PD detection is highly challenged. Besides, all PD patients' symptoms are neither the same nor they all become pronounced at the same stage of the illness. Therefore, this work aims to combine more than one symptom by collecting data and detecting PD with the help of a cloud-based machine learning system for monitoring the PD patients in the developing countries.

References:

1. Anitha R, Nandhini T, Sathish Raj S, Nikitha V, "Early detection of Parkinson's Disease using Machine Learning", 2020

2. Md. SakiburRahman Sajal, Md. Tanvir Ehsan, Ravi VaidyaNathan, Shouyan Wang, Tpu Aziz and Khondaker Abdullah Al Mamun, "Telemonitoring

   Parkinson's Disease using Machine Learning by combining Tremor and Voice Analysis", 2020

3. Jaichandran R, Leelavathy S , Usha Kiruthika S , Goutham Krishna , Mevin John Mathew and Jomon Baiju, "Machine Learning technique

based

Parkinson’s Diseases Detection from Spiral and Voice Inputs”, 2020

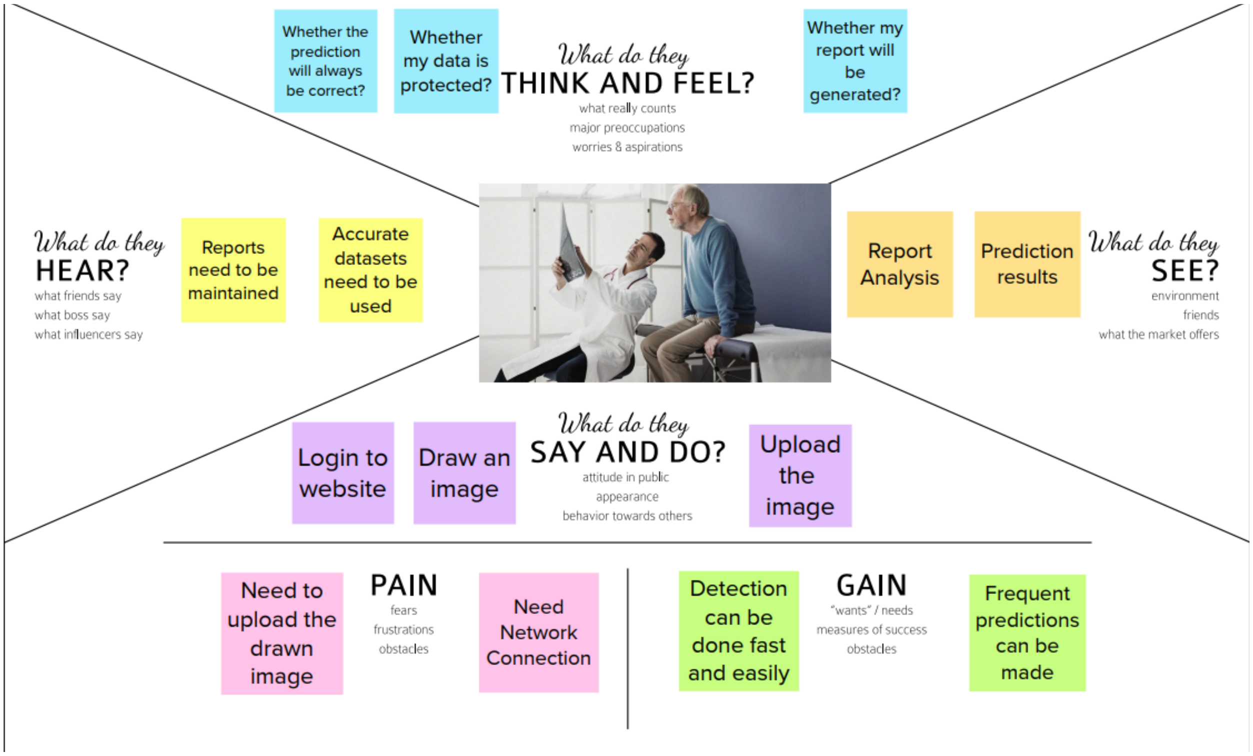
4. Radouani Laila, Lagdali Salwa, Rziza Mohammed, “Detection of voice impairment for Parkinson's disease using machine learning tools”, 2021
5. C K Gomathy, B.Varshini, B.Varsha, B.Dheeraj Kumar Reddy, “The Parkinson’s Disease Detection using Machine Learning Techniques.”, 2021

Problem Statement Definition:

By processing the handdrawn spiral and wave images of the patients we can create a model to learn the difference between healthy and Parkinson affected drawing patterns. The patients provides their handdrawn image and the our machine learrning model predicts whether the patient is affected by Parkinson's disease.

Ideation and Proposed Solution:

Empathy map canvas:



Whether the prediction will always be correct?

Whether my data is protected?

What do they
THINK AND FEEL?
what really counts
major preoccupations
worries & aspirations

Whether my report will be generated?

What do they
HEAR?
what friends say
what boss say
what influencers say

Reports need to be maintained

Accurate datasets need to be used

Report Analysis

Prediction results

What do they
SEE?
environment
friends
what the market offers

Login to website

Draw an image

What do they
SAY AND DO?
attitude in public
appearance
behavior towards others

Upload the image

Need to upload the drawn image

PAIN
fears
frustrations
obstacles

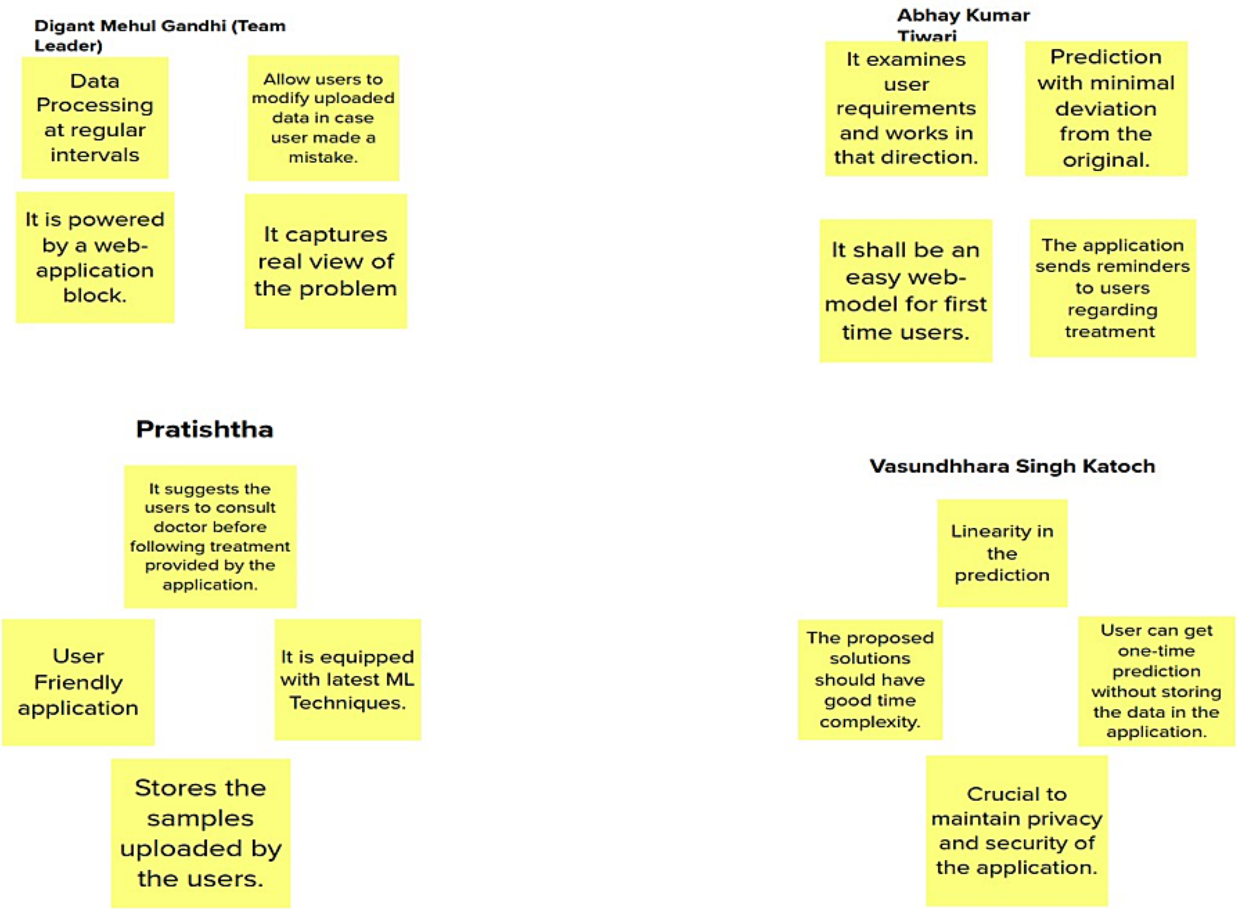
Need Network Connection

Detection can be done fast and easily

GAIN
"wants" / needs
measures of success
obstacles

Frequent predictions can be made

Ideation and Brainstorming:



Digant Mehul Gandhi (Team Leader)

Data Processing at regular intervals

Allow users to modify uploaded data in case user made a mistake.

It is powered by a web-application block.

It captures real view of the problem

Abhay Kumar Tiwari

It examines user requirements and works in that direction.

Prediction with minimal deviation from the original.

It shall be an easy web-model for first time users.

The application sends reminders to users regarding treatment

Pratishtha

It suggests the users to consult doctor before following treatment provided by the application.

User Friendly application

It is equipped with latest ML Techniques.

Stores the samples uploaded by the users.

Vasundhhara Singh Katoch

Linearity in the prediction

The proposed solutions should have good time complexity.

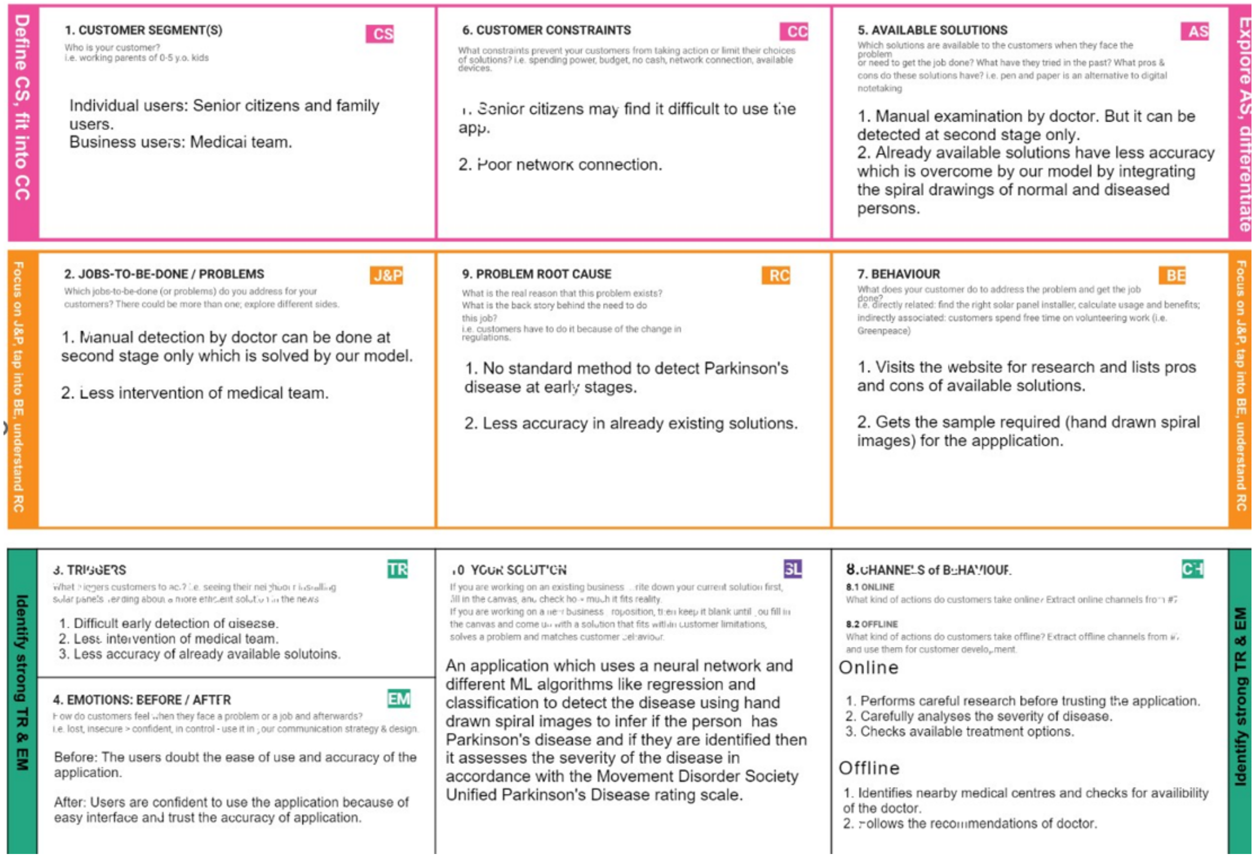
User can get one-time prediction without storing the data in the application.

Crucial to maintain privacy and security of the application.

Proposed Solution:

Idea / Solution description	Our model processes the hand drawn spiral and wave images using a neural network that infers whether the patient has Parkinson's disease.
Novelty / Uniqueness	This application offers medical advice and solutions as the next step after user is confirmed based on the presence of Parkinson’s disease. This can be used direct by medical team for analysing and offering the solutions at much positive scaling time.

Problem Solution fit:



Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

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

Individual users: Senior citizens and family users.
Business users: Medical team.

6. CUSTOMER CONSTRAINTS CC

What constraints prevent your customers from taking action or limit their choices of solutions? i.e. spending power, budget, no cash, network connection, available devices.

1. Senior citizens may find it difficult to use the app.

2. Poor network connection.

5. AVAILABLE SOLUTIONS AS

Which solutions are available to the customers when they face the problem
or need to get the job done? What have they tried in the past? What pros & cons do these solutions have? i.e. pen and paper is an alternative to digital notetaking

1. Manual examination by doctor. But it can be detected at second stage only.
2. Already available solutions have less accuracy which is overcome by our model by integrating the spiral drawings of normal and diseased persons.

Explore AS, differentiate

Focus on J&P, tap into BE, understand RC

2. JOBS-TO-BE-DONE / PROBLEMS J&P

Which jobs-to-be-done (or problems) do you address for your customers? There could be more than one; explore different sides.

1. Manual detection by doctor can be done at second stage only which is solved by our model.

2. Less intervention of medical team.

9. PROBLEM ROOT CAUSE RC

What is the real reason that this problem exists?
What is the back story behind the need to do this job?
i.e. customers have to do it because of the change in regulations.

1. No standard method to detect Parkinson's disease at early stages.

2. Less accuracy in already existing solutions.

7. BEHAVIOUR BE

What does your customer do to address the problem and get the job done?
i.e. directly related: find the right solar panel installer, calculate usage and benefits; indirectly associated: customers spend free time on volunteering work (i.e. Greenpeace)

1. Visits the website for research and lists pros and cons of available solutions.

2. Gets the sample required (hand drawn spiral images) for the appplication.

Focus on J&P, tap into BE, understand RC

Identify strong TR & EM

3. TRIGGERS TR

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

1. Difficult early detection of disease.
2. Less intervention of medical team.
3. Less accuracy of already available solutoins.

4. EMOTIONS: BEFORE / AFTER EM

How do customers feel when they face a problem or a job and afterwards?
i.e. lost, insecure > confident, in control - use it in your communication strategy & design.

Before: The users doubt the ease of use and accuracy of the application.

After: Users are confident to use the application because of easy interface and trust the accuracy of application.

10. YOUR SOLUTION SL

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

An application which uses a neural network and different ML algorithms like regression and classification to detect the disease using hand drawn spiral images to infer if the person has Parkinson's disease and if they are identified then it assesses the severity of the disease in accordance with the Movement Disorder Society Unified Parkinson's Disease rating scale.

8. CHANNELS of BEHAVIOUR CH

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

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

Online

1. Performs careful research before trusting the application.
2. Carefully analyses the severity of disease.
3. Checks available treatment options.

Offline

1. Identifies nearby medical centres and checks for availibility of the doctor.
2. Follows the recommendations of doctor.

Identify strong TR & EM

Requirement Analysis:

Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	**Functional Requirement (Epic)**	**Sub Requirement (Story / Sub-Task)**
FR-1	User Authentication	Registration through Gmail, Login to the application, Confirmation via mail and OTP
FR-2	Data management	Web server has access to change/edit data and update it to server.
FR-3	Input data upload	Data is uploaded for analysis and prediction
FR-4	Testing	Applying the algorithms on the test data
FR-5	Prediction	Prediction is made by the model
FR-5	Result	Results of presence of Parkinson or not is displayed

Non-functional Requirements:

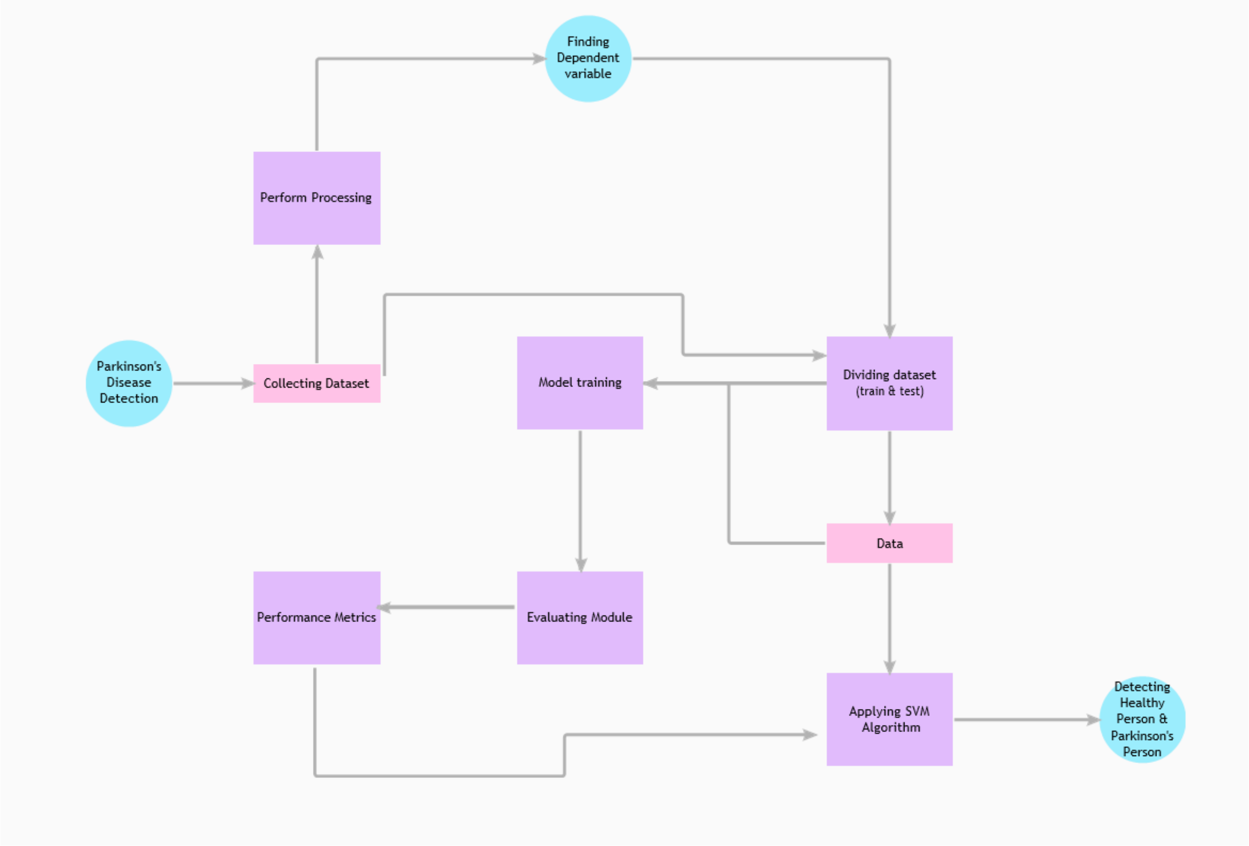
Following are the non-functional requirements of the proposed solution.

FR No.	**Non-Functional Requirement**	**Description**
NFR-1	**Usability**	The UI of the application must be user-friendly and easy to use. The input loading should be enabled faster.
NFR-2	**Security**	The image and voice records should be secure and must not be accessible by everyone.
NFR-3	**Reliability**	The prediction of the system must be

		with higher accuracy so that it will be trusted by users.
NFR-4	**Performance**	The XGBoost algorithm used for detecting PD should incorporate a sparsity-aware split finding algorithm to handle different types of sparsity patterns in the data. Out-of-core computing feature of the XGBoost algorithm should optimize the available disk space and maximizes its usage.
NFR-5	**Availability**	The application should be available to all groups of people all the time.
NFR-6	**Scalability**	XGBooster should not only able to keep up with all those other algorithms but exceeds them in performance. XGBoost should be able to solve real- world scale problems using a minimal number of resources.

Project Design:

Data flow diagrams:


Finding Dependent variable
Perform Processing
Parkinson's Disease Detection
Collecting Dataset
Model training
Dividing dataset (train & test)
Data
Performance Metrics
Evaluating Module
Applying SVM Algorithm
Detecting Healthy Person & Parkinson's Person

Solution and technical architecture:

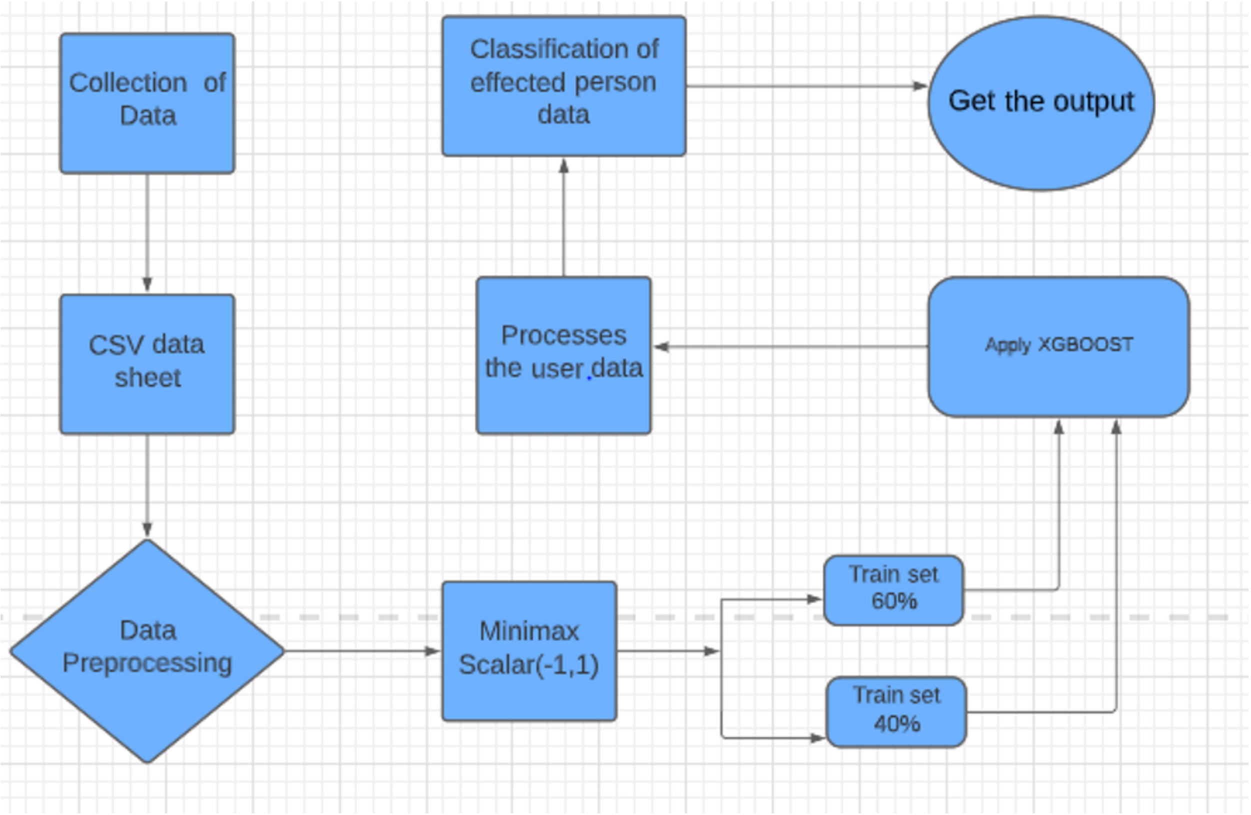


Table-1 : Components & Technologies:

S. No	Component	Description	Technology
1.	User Interface	Users interact with the application using UI	HTML,CSS,JS
2.	Application Logic-1	The logicof the processis to collect all the data set anddata set are in the CSV extension this CSV data sheet takesthe preprocessing stage.	Microsoft Excel

3.	Application Logic-2	The preprocessed data has been splitted into the train set of 60% and 40%. Applying XGBOOSTis an algorithm. That has recently been dominating applied gadget learning. XGBoost set of rules is an implementation of gradient boosted choice timber. That changed into thedesign for paceand overallperformance.	XGBOOST Algorithm
	Application Logic-3	The user datais processed by using XGBOOST algorithm and identifies the affected personby using the affected person’s data .	Jupyter Notebook , Python
4.	Database	Stored data , access thedata, granting statusvalues and checking the userstatus value	MySQL, NoSQL.
5.	Testing	The functional testingof the software isexpected to perform. 1. Create input-data basedon the function's specifications. 2. It Determines theoutput based up on thefunction's specifications.	Software Testing

		3. Execute the testcase. 4. Compare the actualand expected outputs.	
6.	Data Storage	The data has beenstored in the database	MySQL

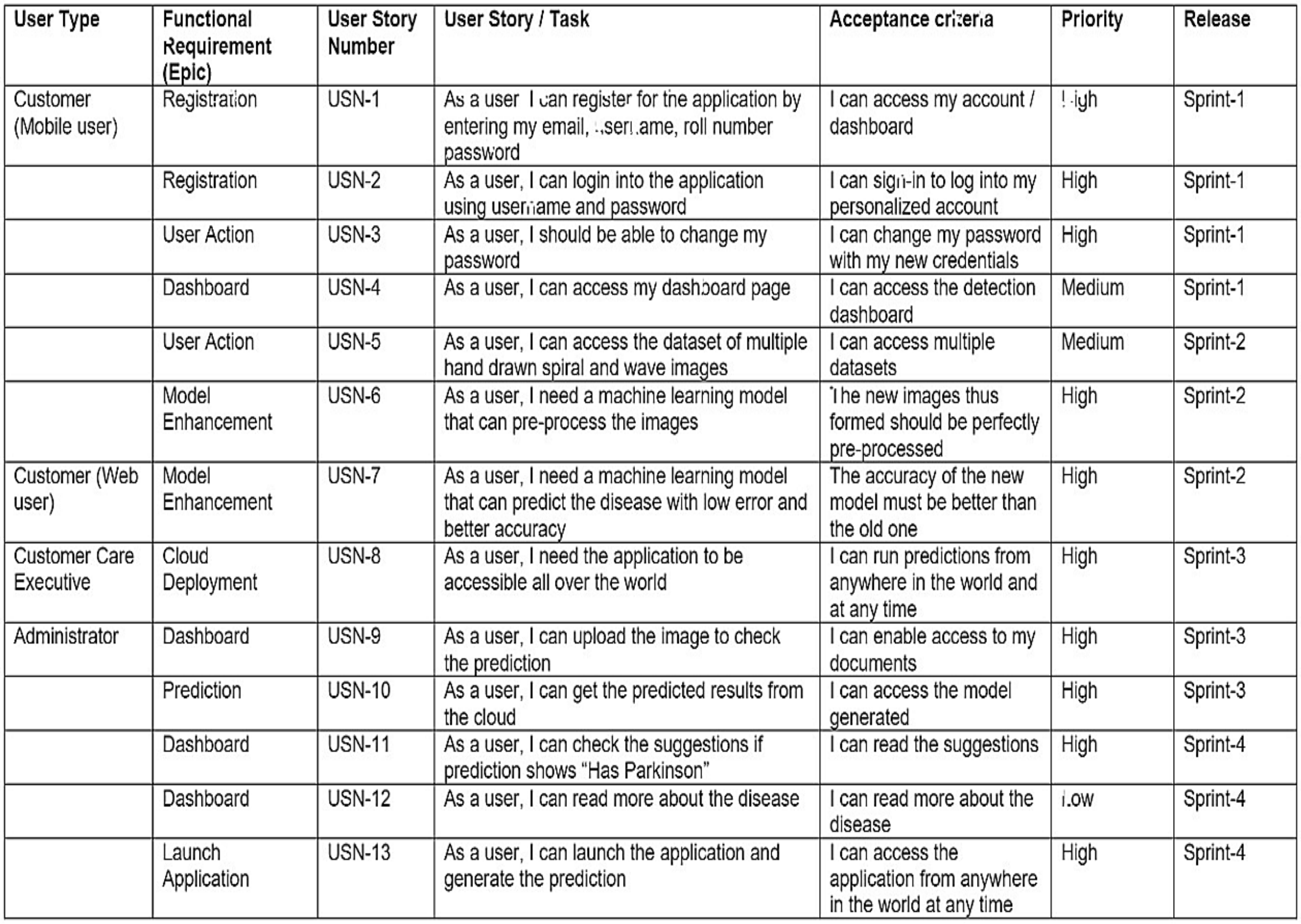
7.	Machine Learning Model-1	To build a model to accurately detectthe presence of Parkinson'sdisease in an individual.	XGBOOST algorithm.
8.	Machine Learning Model-2	Decision treeclassification was usedto gather significant, objective features for both disease classification and stageclassification .	Decision Tree Classifier.
9.	Machine Learning Model-3	To diagnose Parkinson Detection(PD) patients with 70.26% accuracy and a precision of 0.64 for test data.In[52], the authors proposed a method to diagnose PD using the selection and extrac- tionof features and preprocessing classification.	Naive Bayes

10.	Infrastructure (Cloud)	Application Deployment on Cloud computing	Cloud Foundry.

Table-2: Application Characteristics:

S. No	Characteristics	Description	Technology
1.	Open-Source Frameworks	Flask,Tensorflow ASR	Python,Github
2.	Security Implementations	Encryption mechanisms	Built-in encryption
3.	Scalable Architecture	XGBoost,multiple requests,large data storage	Machine Learning,IBM Watson,MySQL
4.	Availability	Available all the time	IBM Cloud
5.	Performance	Flask	Python

User Stories:



User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer (Mobile user)	Registration	USN-1	As a user I can register for the application by entering my email, username, roll number password	I can access my account / dashboard	High	Sprint-1
	Registration	USN-2	As a user, I can login into the application using username and password	I can sign-in to log into my personalized account	High	Sprint-1
	User Action	USN-3	As a user, I should be able to change my password	I can change my password with my new credentials	High	Sprint-1
	Dashboard	USN-4	As a user, I can access my dashboard page	I can access the detection dashboard	Medium	Sprint-1
	User Action	USN-5	As a user, I can access the dataset of multiple hand drawn spiral and wave images	I can access multiple datasets	Medium	Sprint-2
	Model Enhancement	USN-6	As a user, I need a machine learning model that can pre-process the images	The new images thus formed should be perfectly pre-processed	High	Sprint-2
Customer (Web user)	Model Enhancement	USN-7	As a user, I need a machine learning model that can predict the disease with low error and better accuracy	The accuracy of the new model must be better than the old one	High	Sprint-2
Customer Care Executive	Cloud Deployment	USN-8	As a user, I need the application to be accessible all over the world	I can run predictions from anywhere in the world and at any time	High	Sprint-3
Administrator	Dashboard	USN-9	As a user, I can upload the image to check the prediction	I can enable access to my documents	High	Sprint-3
	Prediction	USN-10	As a user, I can get the predicted results from the cloud	I can access the model generated	High	Sprint-3
	Dashboard	USN-11	As a user, I can check the suggestions if prediction shows "Has Parkinson"	I can read the suggestions	High	Sprint-4
	Dashboard	USN-12	As a user, I can read more about the disease	I can read more about the disease	Low	Sprint-4
	Launch Application	USN-13	As a user, I can launch the application and generate the prediction	I can access the application from anywhere in the world at any time	High	Sprint-4

Project Planning and Scheduling:

Sprint Planning and Estimation:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	3	High	Karthikeyan C Mathesh M Rajesh Kumar B Lokhu Prasanth A

Sprint1		USN-2	As a user, I will receive confirmation email once I have registered for the application	2	High	Karthikeyan C Mathesh M Rajesh Kumar B Lokhu Prasanth A
Sprint2		USN-3	As a user, I can register for the application through Facebook	3	Low	Karthikeyan C Mathesh M Rajesh Kumar B Lokhu Prasanth A
Sprint2		USN-4	As a user, I can register for the application through Gmail	3	Medium	Karthikeyan C Mathesh M Rajesh Kumar B Lokhu Prasanth A
Sprint2	Login	USN-5	As a user, I can log into the application by entering email & password	3	High	Karthikeyan C Mathesh M Rajesh Kumar B Lokhu Prasanth A
Sprint3	Dashboard	USN-6	As a user, I can upload my images and get my details.	3	High	Karthikeyan C Mathesh M Rajesh Kumar B Lokhu Prasanth A
Sprint1	Logout	USN-7	As a user I can logout successfully.	2	Medium	Karthikeyan C Mathesh M Rajesh Kumar B Lokhu Prasanth A

Sprint4	Feedback	USN-8	A customer care executive, I can able to interact with all the customer and get their feedback which is used to enhance the scope of the project.	2	Medium	Karthikeyan C Mathesh M Rajesh Kumar B Lokhu Prasanth A
Sprint	**Functional Requirement (Epic)**	**User Story Number**	**User Story / Task**	**Story Points**	**Priority**	**Team Members**
Sprint3	Image processing localization	USN-9	The uploaded image is preprocessed and fed into trained model.	3	High	Karthikeyan C Mathesh M Rajesh Kumar B Lokhu Prasanth A
Sprint4	Classification and prediction	USN-9	The model classifies and predicts the type of disease.	3	High	Karthikeyan C Mathesh M Rajesh Kumar B Lokhu Prasanth A
Sprint4	Report generation	USN-10	Based on the prediction of Parkinson’s disease, the health care is generated to provide the feedback.	2	Medium	Karthikeyan C Mathesh M Rajesh Kumar B Lokhu Prasanth A

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)
Sprint1	20	6 Days	20 Oct 2022	26 Oct 2022	20	26 Oct 2022
Sprint2	20	6 Days	27 Oct 2022	02 Nov 2022	20	31 O ct 2022
Sprint3	20	6 Days	02 Nov 2022	08 Nov 2022	20	06 Nov 2022
Sprint4	20	6 Days	08 Nov 2022	14 Nov 2022	20	08 Nov 2022

Coding and Solutioning:

Register and Login:

In the home page, the links for login and register are available. If the user is new then he/she can register for a new account using the register button. If the user has already registered, he/she can login using his username and password.

home.html

```
<!DOCTYPE
html>
<html lang="en">
<head>
   <meta charset="UTF-8">
   <meta http-equiv="X-UA-Compatible" content="IE=edge">
   <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">
   <link rel="stylesheet" type="text/css" href="{{ url_for('static',
filename='styles.css')
}}" />
```

```
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Home</title>
    <style>
        .bg-img{
        background-image: url("{{ url_for('static',filename='pd.jpg') }}");
        }
    </style>
</head>

<body>
    <div class="col-md-8">
        {% with messages = get_flashed_messages(with_categories=true) %}
            {% if messages %}
                {% for category, message in messages %}
                    <div class="alert alert-{{category}}">
                        {{ message }}
                    </div>
                {% endfor %}
            {% endif %}
        {% endwith %}
        {% block content %} {% endblock %}
    </div>
    <div class="bg-img">
        <div class="topnav">
            <div class="topnav-right">
    <a href="{{ url_for('login') }}">Login Page</a>
    <a href="{{ url_for('register') }}">Register Page</a>
    <a href="{{ url_for('update') }}">Update Password</a>
</div>
</div>
</div>
    <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
integrity="sha384q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8ab
tTE1Pi6jizo" crossorigin="anonymous"></script>
```

```
 <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.14.7/dist/umd/popper.min.js"
integrity="sha384-
UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0
W1
" crossorigin="anonymous"></script>
<script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/js/bootstrap.min.js"
integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"
crossorigin="anonymous"></script>
</body>
</html>
```

login.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Login</title>
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.css"
integrity="sha384-
ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T"
crossorigin="anonymous">
    <link rel="stylesheet" type="text/css" href="{{ url_for('static',
filename='styles.css') }}" />
    <link rel="stylesheet" type="text/css" href="{{ url_for('static',
filename='style.css') }}" />
    <link rel="stylesheet" href="../static/styles.css">
    <style>
```

```
body{
    background-image: url("{{ url_for('static',filename='login.jpg') }}");
min-height: 520px;
  }
  </style>
</head>
<body>
  <div class="text">
  <div class="col-md-8">
    {% with messages = get_flashed_messages(with_categories=true) %}
      {% if messages %}
        {% for category, message in messages %}
          <div class="alert alert-{{category}}">
            {{ message }}
          </div>
        {% endfor %}
      {% endif %}
    {% endwith %}
    {% block content %} {% endblock %}
  </div>

  <h1 style="font-family: Copperplate, fantasy;">Enter Login Details</h1><br>

  <form method="POST" action="">
    {{ form.hidden_tag() }}
    <fieldset class="form-group">
      <div>
        {{ form.username.label(class="form-control-label") }}

        {% if form.username.errors %}
          {{ form.username(class="form-control form-control-lg is-invalid") }}
          <div class="invalid-feedback">
            {% for error in form.username.errors %}
              <span>{{ error }}</span>
```

```
                    {% endfor %}
                </div>
            {% else %}
                {{ form.username(class="form-control form-control-lg") }}
{% endif %}
        </div>



        <div>
            {{ form.password.label(class="form-control-label") }}

            {% if form.password.errors %}
                {{ form.password(class="form-control form-control-lg is-invalid") }}
                <div class="invalid-feedback">
                    {% for error in form.password.errors %}
                        <span>{{ error }}</span>
                    {% endfor %}
                </div>
            {% else %}
                {{ form.password(class="form-control form-control-lg") }}
{% endif %}
        </div>

      </fieldset>
      <div class="form-group">
        {{ form.submit(class="btn btn-outline-info") }}
      </div>

      <small class="text-muted ml-2">
        <a href='{{url_for('register')}}'>Do not have an account? Sign Up?</a>
</small>
    </form>
    </div>
    <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
```

```
integrity="sha384q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8ab
tTE1Pi6jizo" crossorigin="anonymous"></script>
    <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.14.7/dist/umd/popper.min.js"
integrity="sha384-
UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0
W1
" crossorigin="anonymous"></script>
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/js/bootstrap.min.js"
integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"
crossorigin="anonymous"></script>
</body>

</html>
```

register.html

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

<head>
    <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/css/bootstrap.min.css"
integrity="sha384ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x
9JvoRxT2MZw1T" crossorigin="anonymous">
    <link rel="stylesheet" type="text/css" href="{{ url_for('static',
filename='styles.css') }}" />
    <link rel="stylesheet" type="text/css" href="{{ url_for('static',
filename='style.css')
}}" />

    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Register</title>
    <style>
body{
            background-image: url("{{ url_for('static',filename='login.jpg') }}");
min-height: 520px;
        }
    </style>
</head>

<body>
    <div class="text" style="margin-top: 50px;padding: 30px 30px;">
    <h1 style="font-family: Copperplate, fantasy;">Register With Us</h1> <br>

    <!-- <form method="POST" action="">
        {{ form.hidden_tag() }}
        {{ form.email }} <br> <br>
        {{ form.username }} <br> <br>
        {{ form.rollnumber }} <br> <br>
        {{ form.password }} <br> <br>
        {{ form.submit }} <br> <br>
    </form> -->

    <form method="POST" action="">
        {{ form.hidden_tag() }}
        <fieldset class="form-group">
            <!-- <legend class="border-bottom mb-4">Registration Page</legend> -->

            <div>
                {{ form.email.label(class="form-control-label") }}

                {% if form.email.errors %}
                    {{ form.email(class="form-control form-control-lg is-invalid") }}
                    <div class="invalid-feedback">
            {% for error in form.email.errors %}
              <span>{{ error }}</span>
            {% endfor %}
          </div>
        {% else %}
          {{ form.email(class="form-control form-control-lg") }}
        {% endif %}
      </div>


      <div>
        {{ form.username.label(class="form-control-label") }}

        {% if form.username.errors %}
          {{ form.username(class="form-control form-control-lg is-invalid") }}
          <div class="invalid-feedback">
            {% for error in form.username.errors %}
              <span>{{ error }}</span>
            {% endfor %}
          </div>
        {% else %}
          {{ form.username(class="form-control form-control-lg") }}
{% endif %}
      </div>

      <div>
        {{ form.rollnumber.label(class="form-control-label") }}

        {% if form.rollnumber.errors %}
          {{ form.rollnumber(class="form-control form-control-lg is-invalid") }}
          <div class="invalid-feedback">
            {% for error in form.rollnumber.errors %}
              <span>{{ error }}</span>
            {% endfor %}
          </div>
        {% else %}
          {{ form.rollnumber(class="form-control form-control-lg") }}
{% endif %}
      </div>


      <div>
        {{ form.password.label(class="form-control-label") }}

        {% if form.password.errors %}
          {{ form.password(class="form-control form-control-lg is-invalid") }}
          <div class="invalid-feedback">
            {% for error in form.password.errors %}
              <span>{{ error }}</span>
            {% endfor %}
          </div>
        {% else %}
          {{ form.password(class="form-control form-control-lg") }}
        {% endif %}
      </div>

    </fieldset>
    <div class="form-group">
      {{ form.submit(class="btn btn-outline-info") }}
    </div>

    <small class="text-muted ml-2">
      <a href="{{ url_for('login') }}">Already have an account? Log In</a>
    </small>

  </form>
</div>
  <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js"
```

```
integrity="sha384q8i/X+965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8ab
tTE1Pi6jizo" crossorigin="anonymous"></script>
    <script
src="https://cdn.jsdelivr.net/npm/popper.js@1.14.7/dist/umd/popper.min.js"
integrity="sha384-
UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0
W1
" crossorigin="anonymous"></script>
    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@4.3.1/dist/js/bootstrap.min.js"
integrity="sha384-
JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM"
crossorigin="anonymous"></script>
</body>
</html>
```

Prediction:

After logging in user can test if he/she has pakinson's disease by uploading handdrawn spiral or wave image. The model predicts whether the user has Parkinson's disease. If the user has Parkinson's disease the application offers the user medical suggestions and healthy diets.

predict.html

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <style>

        body{
            background-image: url("{{ url_for('static', filename='predict.jpg') }}");
background-repeat: no-repeat;
            background-size: cover;
height: 100%;

        }
        .text{           text-align:
center;           align-items:
center;           justify-content:
center;            position:
absolute;
            top: 25%;;
bottom: 25%;
            left: 0;
            right: 0;
margin: auto;           font-
weight: bold;
color:aliceblue;
            background-color: rgb(0,0,0); /* Fallback color */
background-color: rgba(0,0,0, 0.7);
            font-weight: bold;
border: 3px solid #f1f1f1;
            width: 40%;
        }
a{
            text-decoration: none;
            color:aliceblue;
        }
        a:hover {
            text-decoration: underline;
        }
    </style>
    <title>Prediction</title>
</head>
```

```
<body>
    <div class="text">
        <h1>The predicted result is:</h1>
        <h1>{{predict}}</h1>
        <a href="/welcome">Click here to go Back to the Dashboard</a>
</div>
    <script src="{{ url_for('static', filename='confetti.js') }}"></script>
    <script>          const start = ()
=> {
setTimeout(function() {
confetti.start()
            }, 1000); // 1000 is time that after 1 second start the confetti ( 1000 = 1 sec)
        };
        //  for stopping the confetti

        const stop = () => {
setTimeout(function() {
confetti.stop()
            }, 5000); // 5000 is time that after 5 second stop the confetti ( 5000 = 5 sec)
        };
// after this here we are calling both the function so it
works         start();         stop();
    </script>
</body>
</html>
```

app.py (prediction part)

```
@app.route('/predict', methods=['GET',
'POST'])def predictSpecies():     if
request.method == 'POST':
        f=request.files['file'] #requesting the file
        basepath=os.path.dirname(__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...")

        dataset = request.form['dataset']
if dataset=='spiral':
            m="C:/Users/Digant Gandhi/OneDrive/Desktop/Sprint4/parkinson.pkl"
else:
            m="C:/Users/Digant
Gandhi/OneDrive/Desktop/Sprint4/parkinson_wave.pkl"         model =
joblib.load(m)        image = cv2.imread(filepath)         output = image.copy()
        output = cv2.resize(output, (128, 128))
        # pre-process the image in the same manner we did earlier
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)
res=model.predict([features])
if(res[0]):
            value="Parkinson"             return
redirect(url_for('suggestion'))         else:
            value="Healthy"
    return render_template('predict.html',predict=value)
```

Database Schema:

```
id INTEGER PRIMARY KEY AUTOINCREMENT,
email TEXT NOT NULL, username
TEXT NOT NULL,roll_number
INTEGER NOT NULL, pass_word
TEXT NOT NULL
```

Testing:

Testcases:

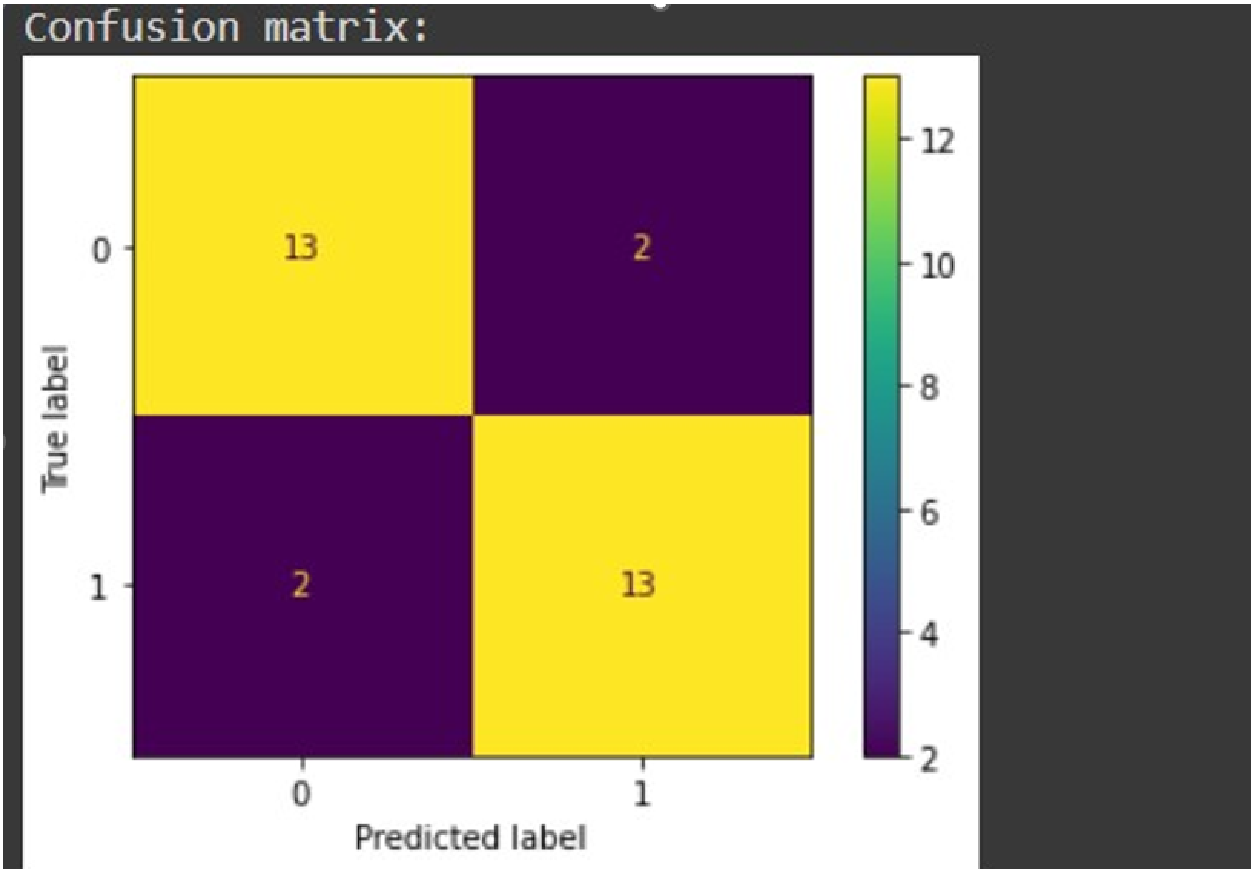
In order to test the functioning of our model, we collected a sample of Parkinson's disease and healthy handdrawn images. We tested our model against them to check if it detected the images accurately.

User Acceptance Testing:

The application performs as expected by detecting whether the patient has Parkinson's disease. All the other functionalities such as Login, Register, Update Password etc ae working as expected.

Results:

Performance Metrics:



Confusion matrix:

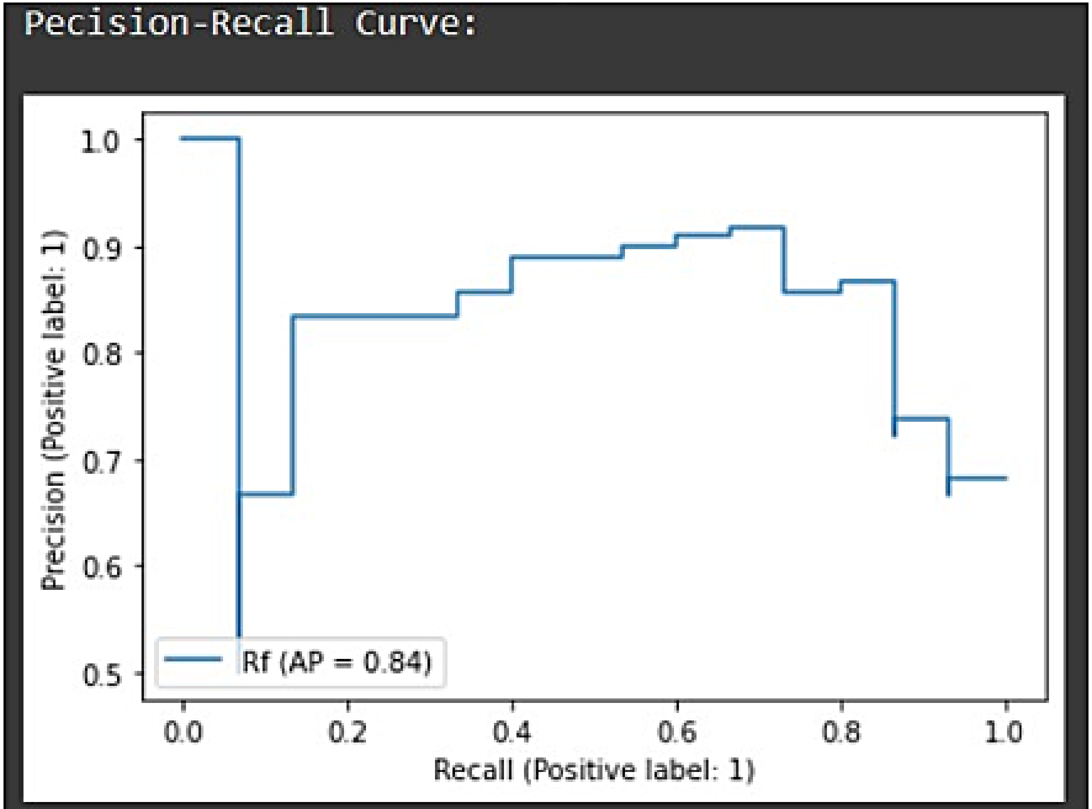
Accuracy: 0.8666666666666667

Precision: 0.8666666666666667

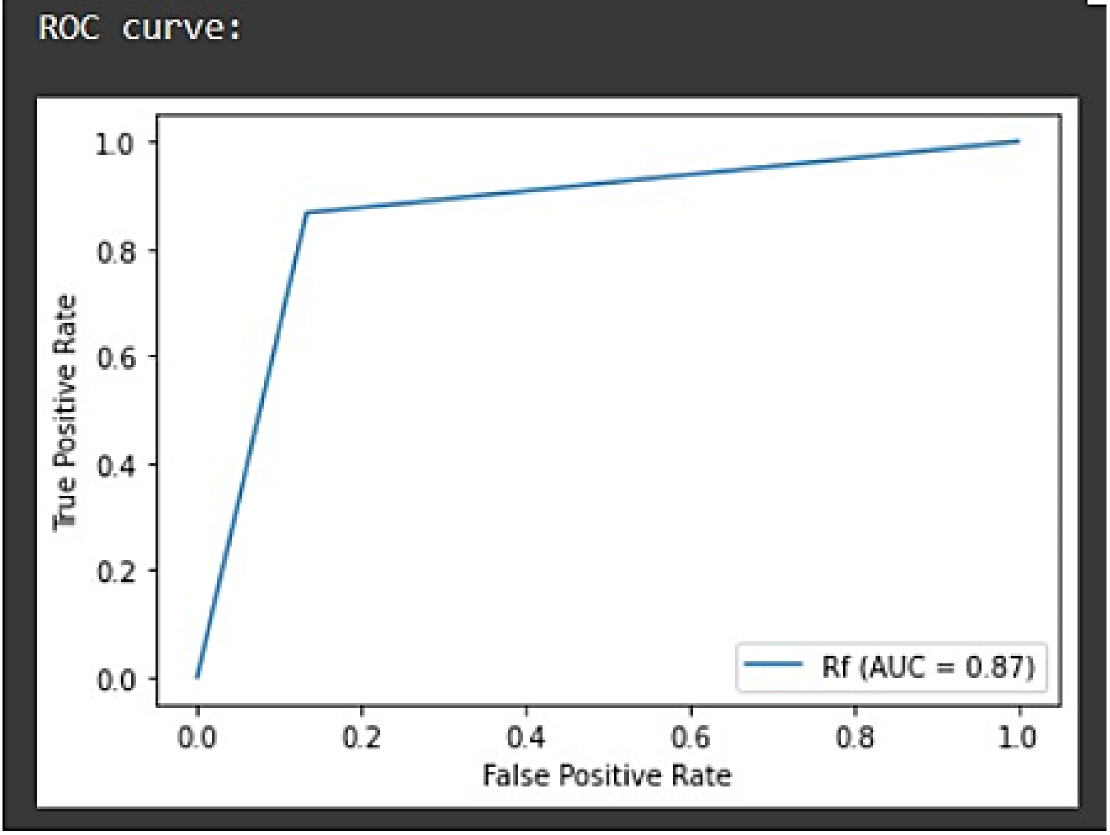
Recall: 0.8666666666666667

Specificity0.8666666666666667

F1 score: 0.8666666666666667



Pecision-Recall Curve:



ROC curve:

Advantages and Disadvantages:

Advantages:

1. Easily accessible
2. Application stays active 24X7
3. Predictions are highly accurate
4. User friendly and provides necessary information about the disease such as symptoms and causes.
5. Provides medical suggestions along with results for those affected with the disease

Disadvantages:

6. May not work properly on huge load (i.e. large number of requests per second)

Conclusion:

We have developed a web application that will help the patients to check whether they have Parkinson's disease. Thus, our application prevents expenditure on testing the disease and helps people with poor economic backgrounds. It also provides medical suggestions to those who are affected by the disease.

Future Work:

In future, the work can be extended to not only predict the disease but also to find out the severity of the disease. According to the severity of the disease necessary medical suggestions and medications can be provided.

Appendix:

Project Demo Link:

https://drive.google.com/file/d/151CBaKnxuiy_O64a1_Pyg0EOnrjtgsom/view?usp=share_link

Source Code:

https://github.com/IBM-EPBL/IBM-Project-8946-1658939044