DETECTING PARKINSONS DISEASE USING MACHINE LEARNING

(TEAM ID: PNT2022TMID52612)

PROJECT REPORT

Submitted by

YOKESH R S (1905062)

VISHAL KARTHIK S (1905060)

NALIN SURIYA S (1905031)

GOURAV GOPAL (1905015)

In partial fulfilment for the award of the degree of

BACHELOR OF ENGINEERING

In

COMPUTER SCIENCE AND ENGINEERING

COIMBATORE INSTITUTE OF TECHNOLOGY

(Government Aided Autonomous Institution Affiliated to Anna University)

COIMBATORE-641 014

ANNA UNIVERSITY - CHENNAI 600 025

NOVEMBER 2022

INDEX

CHAPTER-1

INTRODUCTION

Parkison’s disease (PD) is the second most common disease after Alzheimer’s and it is anticipated that the prevalence of PD is going to increase due to population ageing. The loss of dopaminergic neurons can reach up to 50% at the time of clinical diagnosis and rapidly increases completing by 4 years post-diagnosis. Any neuroprotective strategies that may emerge in the near future could be too late to effectively slow down the neurodegenerative process. Therefore, early objective diagnostic markers are critically needed. Amongst many other symptoms, PD manifests itself through speech disorders, which can be observed as early as 5 years before the diagnosis. Investigations show that Parkinsonian vocal dysfunction can be characterized by: reduced vocal tract volume and reduced tongue flexibility, significantly narrower pitch range, longer pauses and smaller variations in pitch range, voice intensity level, and articulation rate. Therefore, automated acoustic analysis is considered by many researchers as an important non-invasive tool for PD screening. To this end, acoustic analysis aims at solving either regression or classification task: PD severity evaluation based on vocal function assessment from audio samples, as in the Interspeech 2015 computational paralinguistics challenge, or early detection of PD by learning to classify audio samples into healthy control (HC) or PD cases.

1.1 Project Overview

The Parkinson's disease is progressive neuro degenerative disorder that affects a lot only people significantly affecting their quality of life. It mostly affect the motor functions of human. The main motor symptoms are called "parkinsonism" or "parkinsonian syndrome". The symptoms of Parkinson's disease will occur slowly, the symptoms include shaking, rigidity, slowness of movement and difficulty with walking, Thinking and behavior change, Depression and anxiety are also common. There is a model for detecting Parkinson's using voice. The deflections in the voice will confirm the symptoms of Parkinson's disease. In our model, a huge amount of data is collected from the normal person and also previously affected person by Parkinson's disease, these data is trained using machine learning algorithms. From the whole data 80% is used for training and 20% is used for testing. The data of any person can be entered in db to check whether the person is affected by Parkinson’s disease or not. There are 24 columns in the data set each column will indicate the symptom values of a patient except the status column. The status column has 0's and 1's.those values will decide the person is effected with Parkinson’s disease. 1's indicate person is effected, 0's indicate normal conditions.

1.2 Purpose

The purpose of the project is used to detect Parkinson disease using voice based detection by using the value extracted in voice to detect Parkinson disease by the patients from home.

CHAPTER 2

LITERATURE SURVEY

2.1 PAPERS

Paper1

Title: Early Detection of Parkinson's Disease Using Deep Learning and Machine Learning
Author: Wu Wang, Junho Lee, Fouzi Harrou, Ying Sun
Year:2020

An innovative deep learning technique is introduced to early uncover whether an individual is affected with PD or not based on premotor features. Specifically, to uncover PD at an early stage, several indicators have been considered in this study, including Rapid Eye Movement and olfactory loss, Cerebrospinal fluid data, and dopaminergic imaging markers. 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.

Paper2:

Title: High-accuracy detection of early Parkinson's Disease using multiplecharacteristics of finger movement while typing
Author: Warwick R Adams
Year: 2017

In this investigation, keystroke timing information from 103 subjects (comprising 32 with mild PD severity and the remainder non-PD controls) was captured as they typed on a computer keyboard over an extended period and showed that PD affects various characteristics of hand and finger movement and that these can be detected. A novel methodology was used to classify the subjects' disease status, by utilising a combination of many keystroke features which were analysed by an ensemble of machine learning classification models. When applied to two separate participant groups, this approach was able to successfully discriminate between early-PD subjects and controls with 96% sensitivity, 97% specificity and an AUC of 0.98. The technique does not require any specialised equipment or medical supervision, and does not rely on the experience and skill of the practitioner. Regarding more general application, it currently does not incorporate a second cardinal disease symptom, so may not differentiate PD from similar movement- related disorders.

Paper3:

Title: Reliable Parkinson's Disease Detection by Analyzing Handwritten Drawings: Construction of an Unbiased Cascaded Learning System Based onFeature Selection and Adaptive Boosting Model
Author: Liaqat Ali,Ce Zhu,Noorbakhsh Amiri Golilarz, Ashir Javeed, MingyiZhou, Yipeng Liu
Year:2019

Different computer vision and machine learning researchers have proposed micrography and computer vision based methods. But, these methods possess two main problems. The first problem is biasedness in models caused by imbalanced data i.e. machine learning models show good performance on majority class but poor performance on minority class.
Unfortunately, previous studies neither discussed this problem nor took any measures to avoid it. In order to highlight the biasedness in the constructed models and practically demonstrate it, we develop four different machine learning models. To alleviate the problem of biasedness, we propose to use random undersampling method to balance the training process. The second problem is low rate of classification accuracy which has limited clinical significance. To improve the PD detection accuracy, we propose a cascaded learning system that cascades a Chi2 model with adaptive boosting (Adaboost) model. The Chi2 model ranks and selects a subset of relevant features from the feature space while Adaboost model is used to predict PD based on the subset of features. Experimental results confirm that the proposed cascaded system shows better performance than other six similar cascaded systems that used six different state of the art machine learning models. Moreover, it was also observed that the proposed cascaded system improves the strength of conventional Adaboost model by 3.3% and reduces its complexity. Additionally, the cascaded system achieved classification accuracy of 76.44%, sensitivity of 70.94% and specificity of 81.94%.

Paper4:

Title: Parkinson's Disease Detection from Spiral and Wave Drawings usingConvolutional Neural Networks: A Multistage Classifier Approach
Author: Chakraborty, Sabyasachi & Aich, Satyabrata & Jong-Seong-Sim, &Han, Eunyoung & Park, Jinse & Kim, Hee-Cheol
Year:2020

Identification of the correct biomarkers with respect to particular health issues and detection of the same is of paramount importance for the development of clinical decision support systems. For the patients suffering from Parkinson's Disease (PD), it has been duly observed that impairment in the handwriting is directly proportional to the severity of the disease. Also, the speed and pressure applied to the pen while sketching or writing something are also much lower in patients suffering from Parkinson's disease. Therefore, correctly identifying such biomarkers accurately and precisely at the onset of the disease will lead to a better clinical diagnosis. Therefore, in this paper, a system design is proposed for analyzing Spiral drawing patterns and wave drawing patterns in patients suffering from Parkinson's disease and healthy subjects. The system developed in the study leverages two different convolutional neural networks (CNN), for analyzing the drawing patters of both spiral and wave sketches respectively. Further, the prediction probabilities are trained on a metal classifier based on ensemble voting to provide a weighted prediction from both the spiral and wave sketch. The complete model was trained on the data of 55 patients and has achieved an overall accuracy of 93.3%, average recall of 94 % , average precision of 93.5% and average f1 score of 93.94%

Paper5:

Title: Parkinson's Disease Detection Using ResNet50 with Transfer Learning
Author: Jahan, Nusrat & Nesa, Arifatun & Layek, Abu.
Year:2020

In this study, a fine motor symptom that is sketching has been studied. The experiments are done on a significant number of PD patients and Healthy Group (without PD). We proposed a system that can determine the sketching and reports whether a PD patient's sketch or not.
Deep learning algorithms can deal with the solution of different brain generalizing neural networks with the same design. Thus, we applied Convolutional Neural Network (CNN) to classify sketched images to discriminate or identify Parkinson's Disease (PD) affected patients from the regular healthy (without PD) control group. The experiment was done on different CNN models with transfer learning method and applying on Spiral and Wave sketched data. The proposed system achieved 96.67% accuracy on the ResNet50 model with spiral sketching. Contribution of the Paper: The main contribution of this paper is, we have used Transfer learning which enhanced the model performance.

2.1 Existing Problem

Instead of going to hospital and taking MRI scan the existing problem helps the patient to detect the Parkinsons at home with the some basic values extracted from voice recording which is the simple and easiest way.

2.2 References

1. W. Wang, J. Lee, F. Harrou and Y. Sun, "Early Detection of Parkinson's Disease Using Deep Learning and Machine Learning," in *IEEE Access*, vol. 8, pp. 147635-147646, 2020, doi: 10.1109/ACCESS.2020.3016062.
2. Adams WR. High-accuracy detection of early Parkinson's Disease using multiple characteristics of finger movement while typing. PLoS One. 2017 Nov 30;12(11):e0188226. doi: 10.1371/journal.pone.0188226. PMID: 29190695; PMCID:PMC5708704.
3. L. Ali, C. Zhu, N. A. Golilarz, A. Javeed, M. Zhou and Y. Liu, "Reliable Parkinson'sDisease Detection by Analyzing Handwritten Drawings: Construction of an UnbiasedCascaded Learning System Based on Feature Selection and Adaptive Boosting Model," in *IEEE Access*, vol. 7, pp. 116480-116489, 2019, doi: 10.1109/ACCESS.2019.2932037.
4. Chakraborty, Sabyasachi & Aich, Satyabrata & Jong-Seong-Sim, & Han, Eunyoung & Park, Jinse & Kim, Hee-Cheol. (2020). Parkinson's Disease Detection from Spiral and Wave Drawings using Convolutional Neural Networks: A Multistage Classifier Approach. 298-303. 10.23919/ICACT48636.2020.9061497.
5. Jahan, Nusrat & Nesa, Arifatun & Layek, Abu. (2021). Parkinson's Disease Detection Using ResNet50 with Transfer Learning. 11. 17-23.

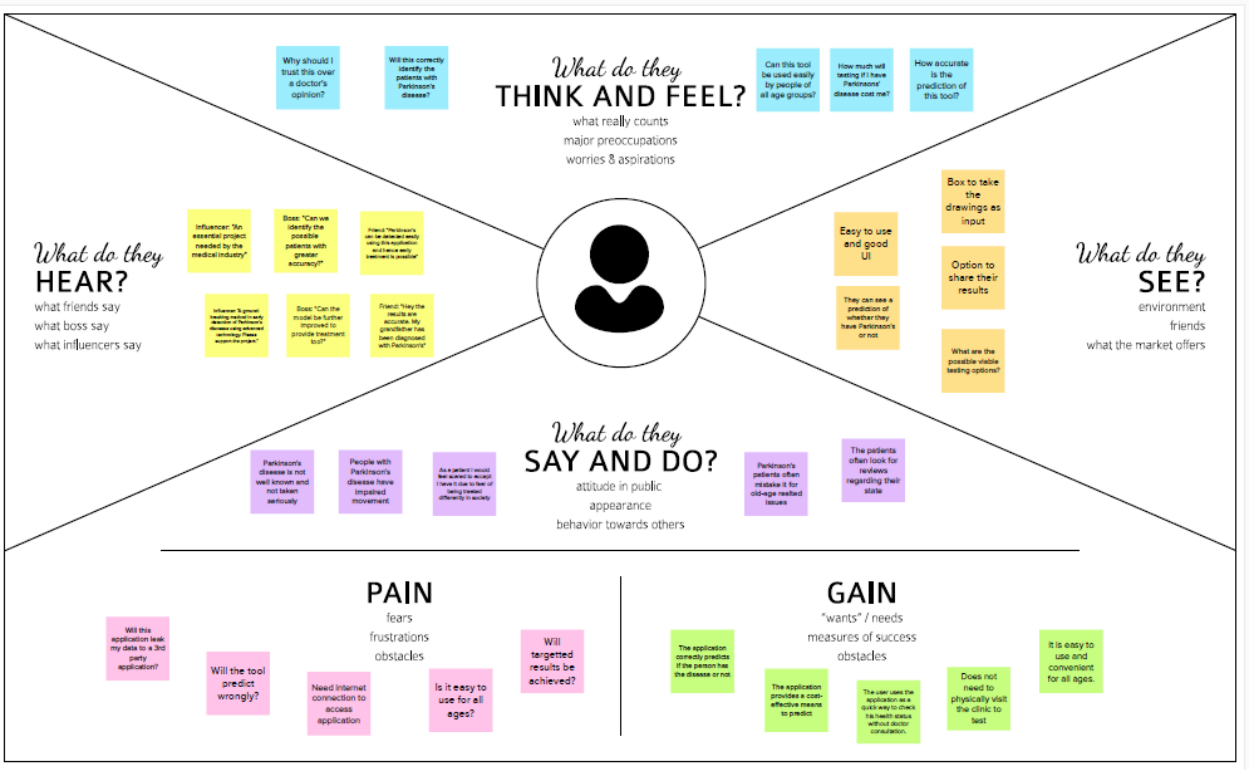
2.3 Problem Statement Defintion

Parkinson’s disease (PD) is a common, neurodegenerative disorder, recognized by the motor symptoms of bradykinesia, tremor, rigidity, and postural impairment. At clinical onset, extensive amounts of dopaminergic neurons have already been lost. The duration of this prodromal phase is uncertain, and it is thought to include predominantly non-motor symptoms. The progressive nature and the symptoms of PD are disabling and reduces the quality of life. Among patients affected in working age, early cessation of employment is common, and such socioeconomic consequences of PD may contribute to an impaired quality of life.

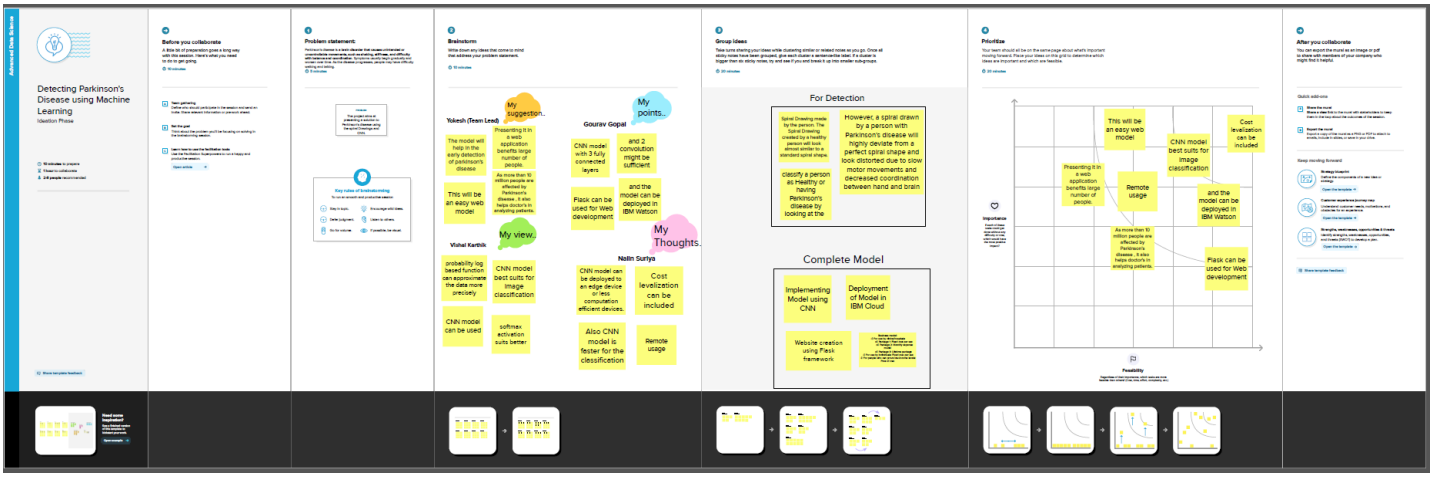
CHAPTER 3

IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation and Brainstorming



3.3Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	The project aims at presenting a solution for Parkinson's disease detection using Spiral Drawings and CNN. The main idea behind the implementation is to classify a person as Healthy or having Parkinson's disease by looking at the Spiral Drawing made by the person. The Spiral Drawing created by a healthy person will look almost similar to a standard spiral shape. However, a spiral drawn by a person with Parkinson's disease will highly deviate from a perfect spiral shape and look distorted due to slow motor movements and decreased coordination between hand and brain
2.	Idea / Solution description	Spiral drawing is a skilled and complex coordinated motor activity. Therefore, it is treated as a sensitive motor assessment and a preliminary test for early symptoms of Parkinson's disease. Hence, the project aims at presents a solution for detecting Parkinson's disease using Spiral Drawings and Convolutional Neural Networks (CNN).
3.	Novelty / Uniqueness	The project aims at optimising the model to limit the number of parameters under 250k for easy deployment on edge devices. The implementation provides a solution for Parkinson's disease detection using CNN to be deployed to an edge device or less computation efficient devices.

4.	Social Impact / Customer Satisfaction	• Test results can be generated efficiently • Early detection of disease. • Good UI experience. • Accurate prediction at good time complexity.

5.	Business Model (Revenue Model)	• For use by clinics/hospitals:
		  o Package 1: Fixed cost per use
		  o Package 2: Monthly expense model
		  o Package 3: Lifetime package
		• For use by individuals: Fixed cost peruse
		• For people who can prove low-income levels: Free of cost

6.	Scalability of the Solution	• Model works same irrespective of number of users • Proper evaluation occurs during production phase to ensure it is highly scalable

3.4 Problems Solution Fit

Project Title: Detecting Parkinson's disease using machine learning | Project Design Phase-I - Solution Fit | Team ID: PNT2022TMID52612

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

- Senior citizens
- People experiencing symptoms of Parkinson's Disease
- Medical professional wanting to perform preliminary test

6. CUSTOMER CONSTRAINTS CC

- High cost of consulting a neurologist.
- Lack of accurate test.
- No access to doctors in remote areas

5. AVAILABLE SOLUTIONS AS

- Prediction using spiral drawings of the person
- Prediction using sensors that monitor the movement of the persons.

Explore AS, differentiate

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

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

- Real Time Detection of Parkinson's Disease
- Provide highly accurate results
- Keep customer's information highly confidential
- Spread awareness about the condition

9. PROBLEM ROOT CAUSE RC

- No accuracy test in market
- Test are highly expensive
- No easy access to test
- No easy to use UI is present in the market

7. BEHAVIOUR BE

- Hand – drawn spiral images has to be given as input by the customer.
- With the results that is produced the customer can consult doctors

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

Identify Strong TR & EM

3. TRIGGERS TR

People will use the application when they experience any kind of symptoms of the disease.

People are assumed to be aware about the symptoms of the disease through public awareness

4. EMOTIONS: BEFORE / AFTER EM

Before : doubt , Anxious , Stressed

After : if diagnosed with the condition fear , depressed otherwise happiness , calmness , peace

10. YOUR SOLUTION SL

Our model processes the hand-drawn spiral images using a neural network that predicts whether the person has Parkinson's disease . A web application is also provided for the user to act as an interactive between themselves and our model.

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE

Upload input data to the application in order to get the results.

8.2 OFFLINE

After prediction the customers can take further actions by consulting doctors or going to hospitals for treatment if necessary.

Extract online or offline CH Or BE

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 Functional Requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Input	Application obtains the data from the user
FR-2	Data Processing	Application processes the images and voice data and prepares it for data classification
FR-3	Data classification	Application classifies the data by giving the user input to the pre-trained ML model
FR-4	Report Generation	Application generates a report for the person with the prediction that has been made by the model

4.2 Non-Functional Requirements

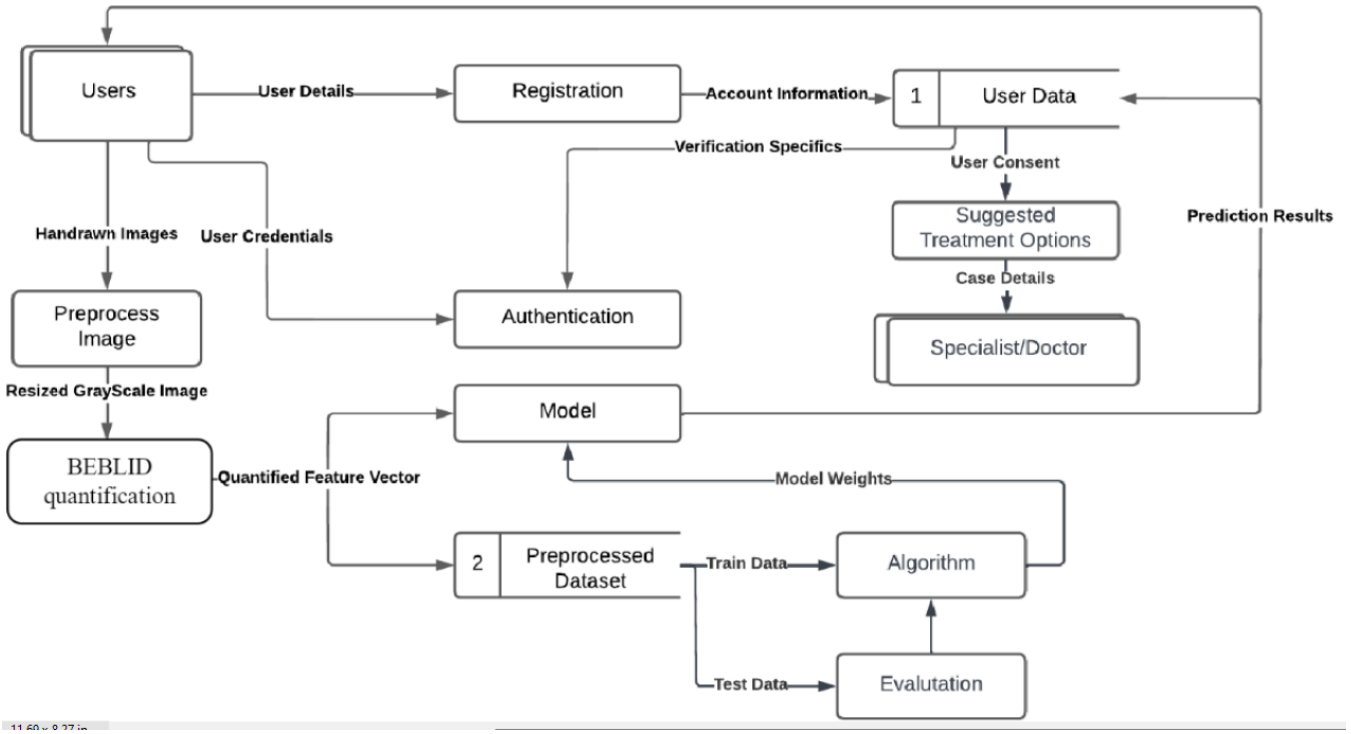
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The application is user-friendly and has a good user interface
NFR-2	**Security**	Customers details are kept confidential and protected
NFR-3	**Reliability**	The application is monitored periodically in terms of its constant prediction ability, quality and availability to the user.
NFR-4	**Performance**	Quick and highly accurate prediction
NFR-5	**Availability**	Application can be used at any time and any place with good internet connectivity
NFR-6	**Scalability**	Application's performance doesn't decrease as the number of users at a time increases.

CHAPTER 5

PROJECT DESIGN

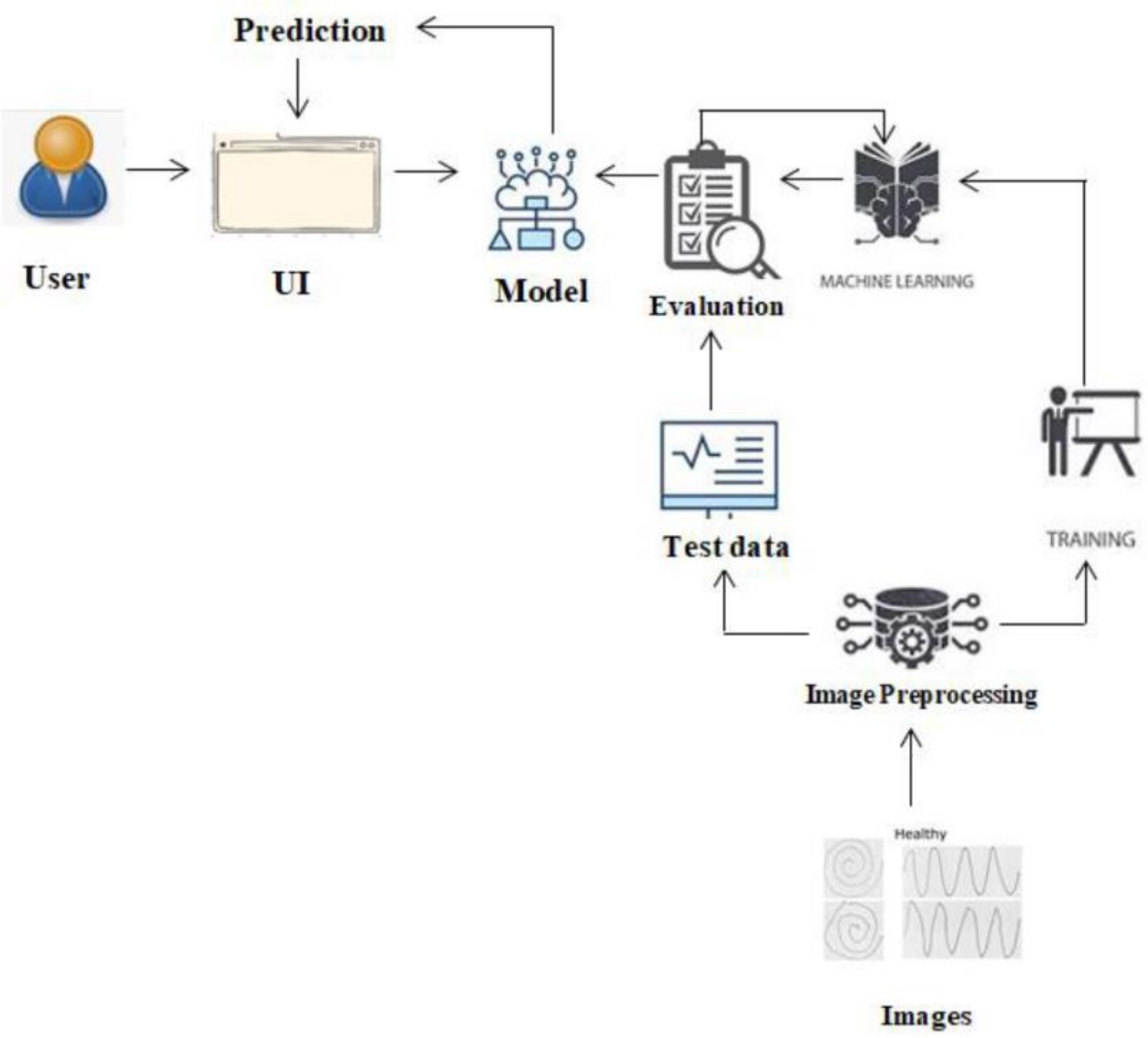
5.1 Data flow diagram

Data Flow Diagram:

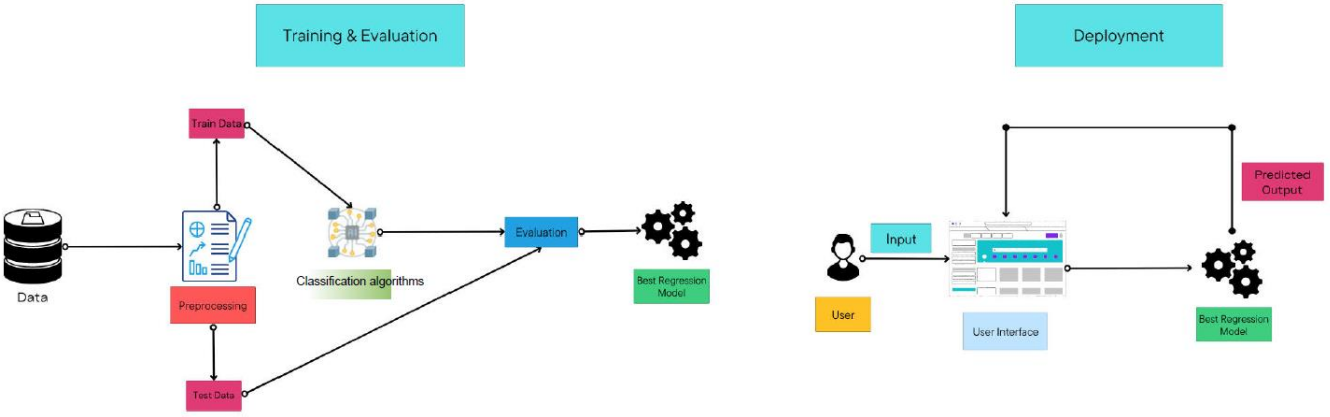


5.2 Solution &Technical Architecture

Solution Architecture



Technical Architecture



5.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release
Customer	Login	USN-1	Entering Web page	Enter the application	High	Sprint 1
	Homepage	USN-2	Entering to the "Homepage" of theUI (Webpage)	Enter the homepage	High	Sprint 1
	About	USN-3	I can click on the "About" to details about the Application	Get the details about the application	Low	Sprint 2
	Begin	USN-4	As a user I can get my voice signal values from the computer.	Choose my voice Recording from my Device and extract the values	High	Sprint 2
	Predict	USN-5	As a user I can turn on the microphone or earphone to record my voice and extract needed values from it	Turn on the microphone or earphone to record the voice and extract values from the recording	High	Sprint 3
		USN-6	Predicting by using voice signal values	Can monitor change in voice or voice shaking and predict parkinsons disease	High	Sprint 3

CHAPTER 6

PROJECT PLANNING &SCHEDULING

6.1 Sprint Planning &Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my username, email, password, contact number and confirming my password.	5	High	TM-1 TM-2
Sprint-1	Login	USN-2	As a user, I can enter the username andpassword after registration for login	5	High	TM-1 TM-2
Sprint-2	Dashboard	USN-3	As a user, I can register for the application through Gmail and see the details in Dashboard	10	Low	TM-3TM-4
Sprint-1	Details about	USN-4	As a user, I can register for the applicationthrough Gmail	5	Medium	TM-1 TM-2
Sprint-1	Login and repeated	USN-5	As a user, I can log into the application by entering email and password	5	High	TM-1 TM-2
Sprint-2	Web page details	USN-6	As a user I must extract certain values fromthe recorded voice and fill the form to detect Parkinsons Disease	10	High	TM-3 TM-4
Sprint-3	Upload the voice signal extracteddetails in the web application	USN-7	As a user I must receive a correct predictedoutput	20	High	TM-1 TM-2
Sprint-4	Provide efficient customer support	USN-8	As a user, I need to get support from developers in case of queries and failure ofservice provided	10	Medium	TM-3 TM-4
Sprint-4	Overview the entire process. Take all the responsibility and act bridge between users and developers	USN-9	We need to satisfy the customer needs in an efficient way and make sure any sort of errorsare fixed	10	High	TM-3 TM-4

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	14 Nov 2022

Velocity

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

AV = sprint duration / velocity = 20/6 = 3.33

CHAPTER 7

CODING & SOLUTIONING

7.1 Feature

Home page:

Once the app opens the home page shows the Parkinsons Detection Logo

Login page:

The login page shows after the home page and ask the user for username and password to login

Preview page:

The preview page ask for the voice signal values to upload.

Result page:

The Result page ask for voice signal values and once the click tests is choosen then the app detect the values and tells you whether the person has parkinsons disease or not

7.2 Code

Base.html

```
<!DOCTYPE
html>
            <html>
            <head>
                <meta charset="utf-8">
                <meta http-equiv="X-UA-Compatible" content="IE=edge">
                <meta name="viewport" content="width=device-width, initial-scale=1">
                <title>Parkinson Disease Detection</title>
                <link rel="stylesheet"
            href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.2/css/bootstrap.min.css">
                <link rel="stylesheet" href="{{ url_for('static', filename='css/styles1.css')}}">
            </head>
            <body>
            <nav class="navbar navbar-expand-lg navbar-light bg-light">
              <div class="collapse navbar-collapse">
                <ul class="navbar-nav ml-auto">
                  <li class="nav-item">
                    <a class="nav-link" href="{{url_for('home')}}">Home</a>
                  </li>
                    {% if not logged_in: %}
                  <li class="nav-item">
                    <a class="nav-link" href="{{url_for('login')}}">Login</a>
                  </li>
                    <li class="nav-item">
                    <a class="nav-link" href="{{url_for('register')}}">Register</a>
                  </li>
```

```
                {% endif %}
              </li>
            </ul>
          </div>
        </nav>
              {% block content %}
              {% endblock %}
        </body>
        </html>
```

Base1.html

```
<html
lang="en">
         <head>
             <meta charset="UTF-8">
             <meta name="viewport" content="width=device-width, initial-scale=1.0">
             <meta http-equiv="X-UA-Compatible" content="ie=edge">
             <title>Parkinson Disease Detection</title>
             <link href="https://cdn.bootcss.com/bootstrap/4.0.0/css/bootstrap.min.css"
        rel="stylesheet">
             <script src="https://cdn.bootcss.com/popper.js/1.12.9/umd/popper.min.js"></script>
             <script src="https://cdn.bootcss.com/jquery/3.3.1/jquery.min.js"></script>
             <script src="https://cdn.bootcss.com/bootstrap/4.0.0/js/bootstrap.min.js"></script>
             <link href="{{ url_for('static', filename='css/main.css') }}" rel="stylesheet">
        </head>
        <body>
             <nav class="navbar navbar-dark bg-dark">
                 <div class="container">
                     <a class="navbar-brand" href="#">Parkinson Disease Detection</a>
                     <!--<button class="btn btn-outline-secondary my-2 my-sm-0"
        type="submit">Help</button>-->
                 </div>
             </nav>
             <div class="container">
                 <div id="content" style="margin-top:2em">{% block content %}{% endblock %}</div>
             </div>
        </body>
        <footer>
             <script src="{{ url_for('static', filename='js/main.js') }}"
        type="text/javascript"></script>
        </footer>
        </html>
```

index.html

```
{% extends
"base1.html"
%} {% block
content %}
          <center>
```

```
<h2>Parkinson Disease Detection</h2>
</center>
<center>
<div>
    <form id="upload-file" method="post" enctype="multipart/form-data">
        <label for="imageUpload" class="upload-label">
            Upload image
        </label>
        <input type="file" name="file" id="imageUpload" accept=".png, .jpg, .jpeg">
    </form>
    <div class="image-section" style="display:none;">
        <div class="img-preview">
            <div id="imagePreview">
            </div>
        </div>
        <div>
            <button type="button" class="btn btn-primary btn-lg " id="btn-
predict">Predict!</button>
        </div>
    </div>
    <div class="loader" style="display:none;"></div>
    <h3 id="result">
        <span> </span>
    </h3>
</div>
</center>
{% endblock %}
```

Index1.html

```
{% extends
"base.html"
%}
{% block content %}
<div class="box">
       <h1>Parkinson Detection</h1><br><br>
  <a href="{{url_for('login') }} " class="btn btn-primary btn-block btn-large">Login</a>
  <a href="{{url_for('register')}}" class="btn btn-secondary btn-block btn-
large">Register</a>
</div>
{% endblock %}
```

login.html

```
{% extends
"base.html"
%}
               {% block content %}
               <div class="box">
                   <h1>Login</h1>
               {% with messages = get_flashed_messages() %}
                     {% if messages %}
                       {% for message in messages %}
                        <p>{{ message }}</p>
                       {% endfor %}
                     {% endif %}
                   {% endwith %}
                   <form action="{{ url_for('login') }}" method="post">
                       <input type="text" name="email" placeholder="Email" required="required"/>
                       <input type="password" name="password" placeholder="Password" required="required"/>
                       <button type="submit" class="btn btn-primary btn-block btn-large">Let me
               in.</button>
                   </form>
               </div>
               {% endblock %}
```

register.html

```
{% extends
"base.html"
%}
               {% block content %}
               <div class="box">
                      <h1>Register</h1>
                   <form action="{{ url_for('register') }}" method="post">
                       <input type="text" name="name" placeholder="Name" required="required" />
                              <input type="email" name="email" placeholder="Email" required="required" />
                        <input type="password" name="password" placeholder="Password" required="required" />
                        <button type="submit" class="btn btn-primary btn-block btn-large">Sign me
               up.</button>
                   </form>
               </div>
               {% endblock %}
```

Main.css

```
img-
preview
{
            width: 256px;
            height: 256px;
            position: relative;
            border: 5px solid #F8F8F8;
            box-shadow: 0px 2px 4px 0px rgba(0, 0, 0, 0.1);
            margin-top: 1em;
            margin-bottom: 1em;
        }
        .img-preview>div {
            width: 100%;
            height: 100%;
            background-size: 256px 256px;
            background-repeat: no-repeat;
            background-position: center;
        }
        input[type="file"] {
            display: none;
        }
        .upload-label{
            display: inline-block;
            padding: 12px 30px;
            background: #541690;
            color: #fff;
            font-size: 1em;
            transition: all .4s;
            cursor: pointer;
        }
        .upload-label:hover{
            background: #34495E;
            color: #541690;
        }
        .loader {
            border: 8px solid #34495E; /* Light grey */
            border-top: 8px solid #34495E; /* Blue */
            border-radius: 50%;
            width: 50px;
            height: 50px;
            animation: spin 1s linear infinite;
        }
        @keyframes spin {
            0% { transform: rotate(0deg); }
            100% { transform: rotate(360deg); }
        }
```

Styles.css

```
*,
*:before,
*:after {
                box-sizing: border-box;
            }
             html {
                font-size: 18px;
                line-height: 1.5;
                font-weight: 300;
                color: #333;
                font-family: "Nunito Sans", sans-serif;
            }
             body {
                margin: 0;
                padding: 0;
                height: 100vh;
                background-color: #ecf0f9;
                background-attachment: fixed;
            }
            .large {
                 font-size: 3rem;
            }
             .content {
                display: flex;
                margin: 0 auto;
                justify-content: center;
                align-items: center;
                flex-wrap: wrap;
                max-width: 1500px;
            }
             p.overview {
                font-size: 12px;
                height: 200px;
                width: 100%;
                overflow: hidden;
                text-overflow: ellipsis;
            }
             .heading {
                width: 100%;
                margin-left: 1rem;
                font-weight: 900;
                font-size: 1.618rem;
                text-transform: uppercase;
                letter-spacing: 0.1ch;
                line-height: 1;
                padding-bottom: 0.5em;
                margin-bottom: 1rem;
```

```
        position: relative;
}
 .heading:after {
        display: block;
        content: '';
        position: absolute;
        width: 60px;
        height: 4px;
        background: linear-gradient(135deg, #1a9be6, #1a57e6);
        bottom: 0;
}
 .description {
        width: 100%;
        margin-top: 0;
        margin-left: 1rem;
        margin-bottom: 3rem;
}
 .card {
        color: inherit;
        cursor: pointer;
        width: calc(33% - 3rem);
        min-width: calc(33% - 3rem);
        height: 400px;
        min-height: 400px;
        perspective: 1000px;
        margin: 1rem auto;
        position: relative;
}
 @media screen and (max-width: 800px) {
        .card {
                width: calc(50% - 3rem);
       }
}
 @media screen and (max-width: 500px) {
        .card {
                width: 100%;
       }
}
 .front, .back {
        display: flex;
        border-radius: 6px;
        background-position: center;
        background-size: cover;
        text-align: center;
        justify-content: center;
        align-items: center;
        position: absolute;
        height: 100%;
        width: 100%;
        -webkit-backface-visibility: hidden;
```

```
        backface-visibility: hidden;
        transform-style: preserve-3d;
        transition: ease-in-out 600ms;
}
 .front {
        background-size: cover;
        padding: 2rem;
        font-size: 1.618rem;
        font-weight: 600;
        color: #fff;
        overflow: hidden;
        font-family: Poppins, sans-serif;
}
 .front:before {
        position: absolute;
        display: block;
        content: '';
        top: 0;
        left: 0;
        right: 0;
        bottom: 0;
        background: linear-gradient(135deg, #1a9be6, #1a57e6);
        opacity: 0.25;
        z-index: -1;
}
 .card:hover .front {
        transform: rotateY(180deg);
}
 .card:nth-child(even):hover .front {
        transform: rotateY(-180deg);
}
 .back {
        background: #fff;
        transform: rotateY(-180deg);
        padding: 0 2em;
}
 .card:hover .back {
        transform: rotateY(0deg);
}
 .card:nth-child(even) .back {
        transform: rotateY(180deg);
}
 .card:nth-child(even):hover .back {
        transform: rotateY(0deg);
}
 .button {
        transform: translateZ(40px);
        cursor: pointer;
        -webkit-backface-visibility: hidden;
        backface-visibility: hidden;
```

```
        font-weight: bold;
        color: #fff;
        padding: 0.5em 1em;
        border-radius: 100px;
        font: inherit;
        background: linear-gradient(135deg, #1a9be6, #1a57e6);
        border: none;
        position: relative;
        transform-style: preserve-3d;
        transition: 300ms ease;
}
 .button:before {
        transition: 300ms ease;
        position: absolute;
        display: block;
        content: '';
        transform: translateZ(-40px);
        -webkit-backface-visibility: hidden;
        backface-visibility: hidden;
        height: calc(100% - 20px);
        width: calc(100% - 20px);
        border-radius: 100px;
        left: 10px;
        top: 16px;
        box-shadow: 0 0 10px 10px rgba(26, 87, 230, 0.25);
        background-color: rgba(26, 87, 230, 0.25);
}
.button.delete-button {
        background-color: rgba(230, 87, 230, 0.25);
        background: linear-gradient(135deg, #e61a46, #e61a1a);
}
.button.delete-button:before {
        background-color: rgba(230, 87, 230, 0.25);
        box-shadow: 0 0 10px 10px rgba(230, 87, 230, 0.25);
}
 .button:hover {
        transform: translateZ(55px);
}
 .button:hover:before {
        transform: translateZ(-55px);
}
 .button:active {
        transform: translateZ(20px);
}
 .button:active:before {
        transform: translateZ(-20px);
        top: 12px;
        top: 12px;
}
.container.add {
```

```
    margin-top: 40px;
    margin-bottom: 20px;
}
.rating {
    color: #E4BB23;
}
.review {
    font-style: italic;
}
 .movie_gens {
        font-size: 11.5px;
}
 .title {
        font-weight: bold;
}
 .release_date {
        font-weight: normal;
}
```

Main.js

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

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

Main.py

```
import
pickle
        import sklearn
        from flask import Flask, render_template,request,redirect,url_for,flash
        from flask_bootstrap import Bootstrap
        from flask_sqlalchemy import SQLAlchemy
        from sqlalchemy.orm import relationship
        from flask_wtf import FlaskForm
        from werkzeug.utils import secure_filename
        from wtforms import StringField, SubmitField,FloatField,IntegerField
        from wtforms.validators import DataRequired
        from werkzeug.security import generate_password_hash,check_password_hash
        import os
        import cv2
        from skimage import feature
        from flask_login import login_user,logout_user,LoginManager,UserMixin,current_user,login_required
        app = Flask(__name__)
        app.config['SECRET_KEY'] = '8BYkEfBA6O6donzWlSihBXox7C0sKR6b'
        app.config['SQLALCHEMY_DATABASE_URI']='sqlite:///database.db'
        app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
        db = SQLAlchemy(app)
        Bootstrap(app)
        login_manager = LoginManager()
        login_manager.init_app(app)
        class users(UserMixin,db.Model):
```

```
    id = db.Column(db.Integer,primary_key=True)
    email= db.Column(db.String(200),nullable=False)
    password = db.Column(db.String(300),nullable=False)
    name = db.Column(db.String(100),nullable=False)
@login_manager.user_loader
def user_load(id):
    return users.query.get(int(id))
@app.route("/")
def home():
    return render_template("index1.html")
@app.route("/register",methods=['GET','POST'])
def register():
    if request.method == 'POST':
       if users.query.filter_by(email=request.form['email']).first():
            flash('User already registered')
            return redirect(url_for('login'))
       else:
        password =
generate_password_hash(request.form['password'],method="pbkdf2:sha256",salt_length=8)
        user = users(
            email = request.form['email'],
            password = password,
            name = request.form['name']
        )
        db.session.add(user)
        db.session.commit()
        return redirect(url_for('home'))
    return render_template('register.html')
@app.route("/login",methods=['GET','POST'])
def login():
    if request.method == 'POST':
        email= request.form['email']
        password = request.form['password']
        k=users.query.filter_by(email=email).first()
        if not k:
            flash('User not registered')
            return redirect(url_for('login'))
        elif check_password_hash(k.password,password):
            login_user(k)
            return redirect(url_for('model'))
        else:
            flash('Wrong password')
            return redirect(url_for('login'))
    return render_template('login.html')
@app.route("/logout")
def logout():
    logout_user()
    return redirect(url_for('home'))
@app.route("/parkinson")
def model():
```

```
    return render_template('index.html')
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
@app.route('/predict', methods=['GET', 'POST'])
def upload():
    if request.method == 'POST':
        f = request.files['file']  # requesting the file
        basepath = os.path.dirname(os.path.realpath('__file__'))  # storing the file directory
        filepath = os.path.join(basepath, "uploads", f.filename)  # storing the file in uploads
folder
        f.save(filepath)
        image = cv2.imread(filepath)
        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)
        model = pickle.loads(open("C:/IBM PROJECT\parkinson_spiral.sav", "rb").read())
        preds= model.predict([features])
        ls = ["healthy", "parkinson"]
        result = ls[preds[0]]
        if(result=="healthy"):
            result=": You are healthy!!"
        elif(result=="parkinson"):
            result+=" disease detected, please visit nearby doctor "
        return result
    return None
admin=[1]
if __name__ == '__main__':
    app.run(debug=True)
```

CHAPTER 8

TESTING

8.1 Test Cases

Test Scenario	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation[Y/N	BUG ID	Executed By
user is able to see the login	1.Enter URL and click go 2. User to login	http://127.0.0.1:5000	Login should display	Working as	Pass				
Verify the UI elements in Login	1.Enter URL and click go 2.Verify login with below UI elements: b.password text box c.Login button d.New customer? Create account link e.Last password? Recovery password link	http://127.0.0.1:5000	Application should show below UI elements: a.username text box b.password text box c.Login button with green colour d.New customer? Create account link e.Last password? Recovery	Working as expected	Fail	Steps are not clear to follow		BUG-1234	
Verify user is able to log into application with Valid credentials	1.Enter URL(http://127.0.0.1:5000) and click go. 2.Enter Valid username/email in Email text box 3.Enter valid password in password text box 4.Click on login button	Username: user1 password: 1234	User should navigate to user account homepage	Working as expected	Pass				
Verify user is able to log into application with InValid credentials	1.Enter URL(http://127.0.0.1:5000) and click go 2.Enter InValid username text box 3.Enter valid password in password text box 4.Click on login button	Username: jjjj password: 7588	Application should show 'Incorrect email or password' validation message.	Working as expected	Pass				
Verify user is able to log into application with InValid credentials	1.Enter URL(http://127.0.0.1:5000) and click go 2.Enter Valid username/email in Email text box 3.Enter Invalid password in password text box 4.Click on login button	Username: aaaa password: 6876	Application should show 'Incorrect email or password' validation message.	Working as expected	Pass				
Verify user is able to log into application with InValid credentials	1.Enter URL(http://127.0.0.1:5000) and click go 2.Enter InValid username in text box 3.Enter Invalid password in password text box 4.Click on login button	Username: bbbb password:19788	Application should show 'Incorrect email or password' validation message.	Working as expected	Pass				

8.2 User Acceptance Testing

- Purpose of Document:

The purpose of this document is to briefly explain the test coverage and open issues of the [Product Name] project at the time of the release to User Acceptance Testing (UAT).

- Defect Analysis:

This report shows the number of resolved or closed bugs at each severity level, and howthey were resolved.

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	12	3	4	3	22
Duplicate	2	0	3	0	5
External	3	2	0	2	7
Fixed	10	1	3	21	35
Not Reproduced	0	0	1	0	1
Skipped	0	1	1	1	3
Won't Fix	0	4	1	2	7
Totals	27	11	13	29	80

- Test Case Analysis

  This report shows the number of test cases that have passed, failed, and untested

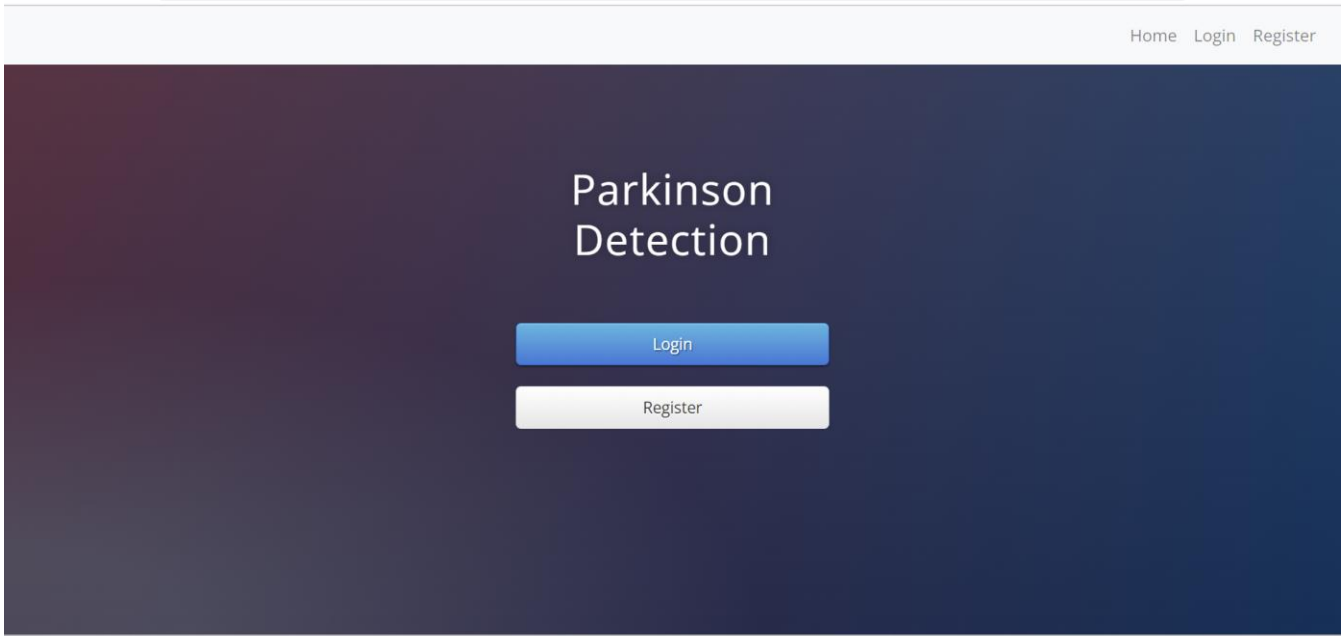
Section	Total Cases	Not Tested	Fail	Pass
Print Engine	8	0	0	8
Client Application	53	0	0	53
Security	2	0	0	2
Outsource Shipping	4	0	0	4
Exception Reporting	8	0	0	8
Final Report Output	5	0	0	5
Version Control	2	0	0	2

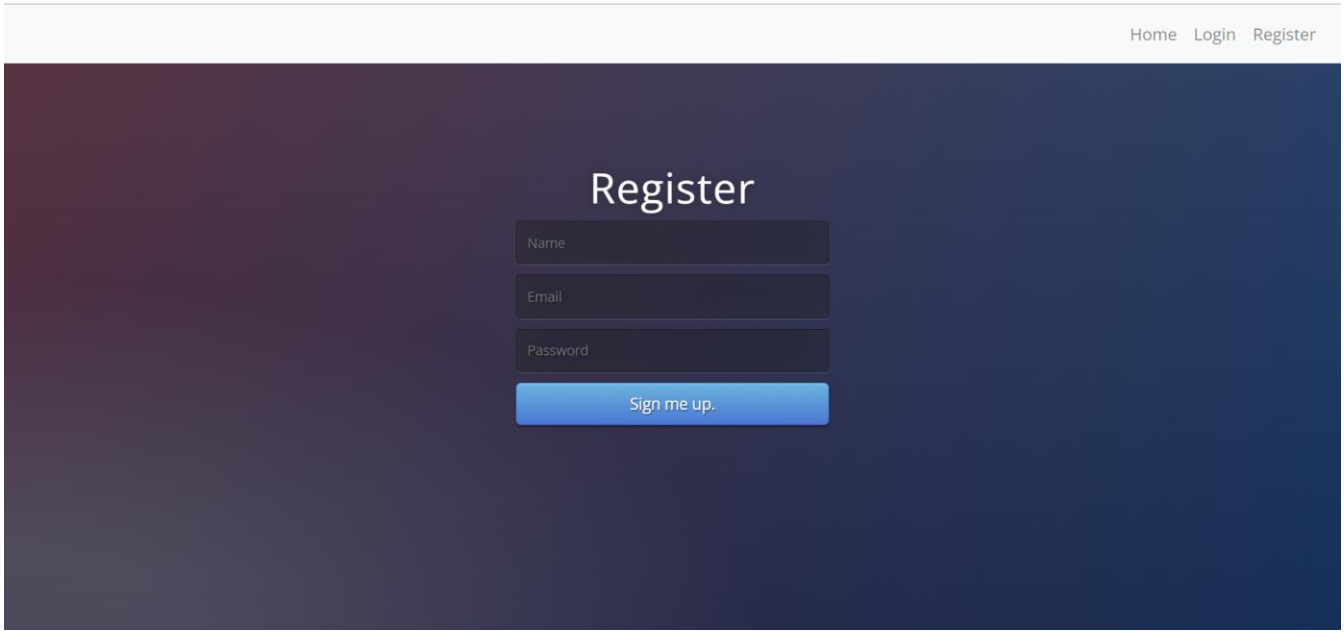
CHAPTER 9

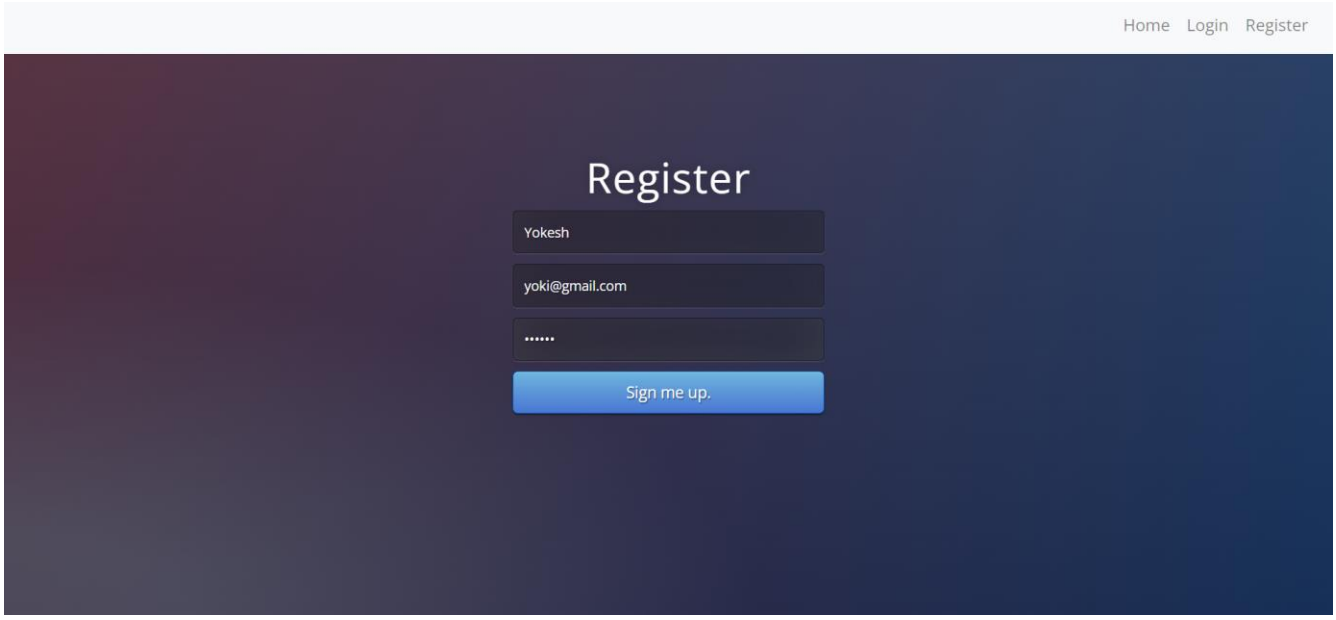
RESULTS

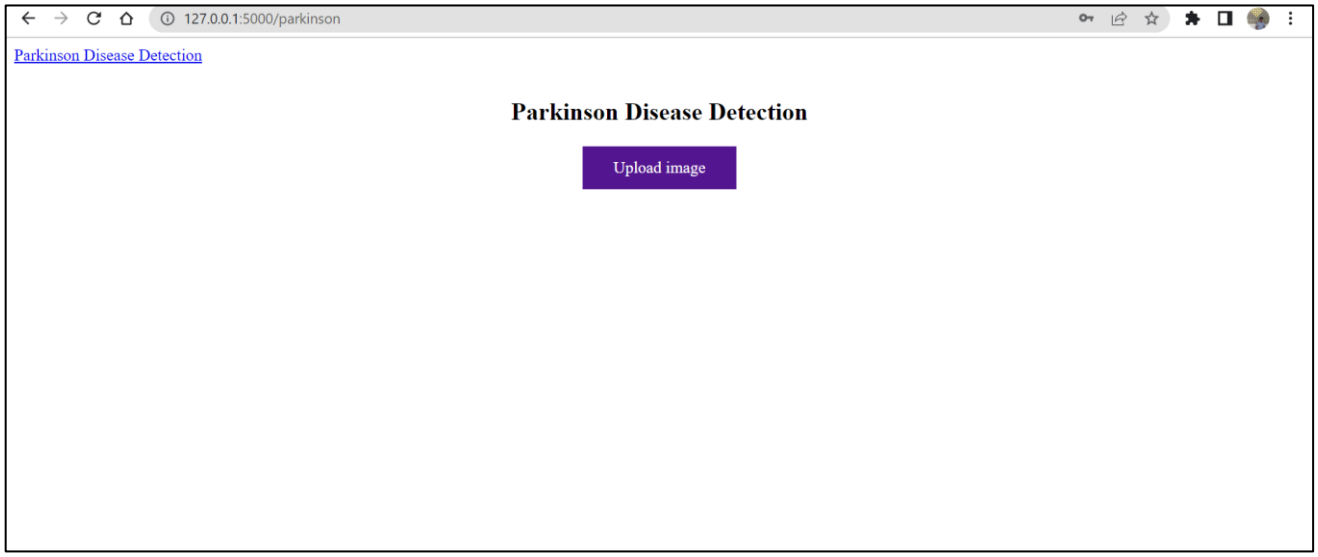
In this project, we found that Parkinsons disease can be detected using the value's obtained from voice recording.

Final findings (Output) of the project along the screenshots as follows**.**






Home
Login
Register
Register
Yokesh
yoki@gmail.com
.....
Sign me up.

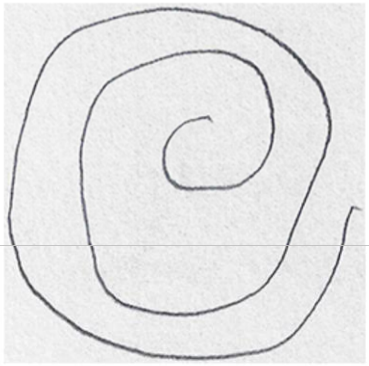

127.0.0.1:5000/parkinson
Parkinson Disease Detection
Parkinson Disease Detection
Upload image

127.0.0.1:5000/parkinson

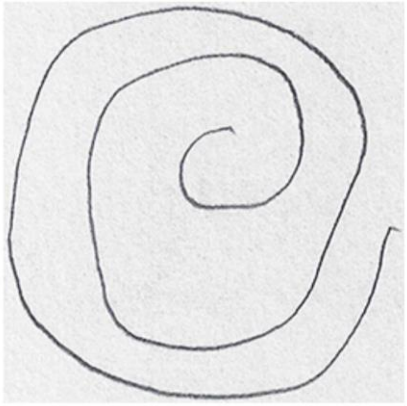
Parkinson Disease Detection

Parkinson Disease Detection

Upload image



Predict!

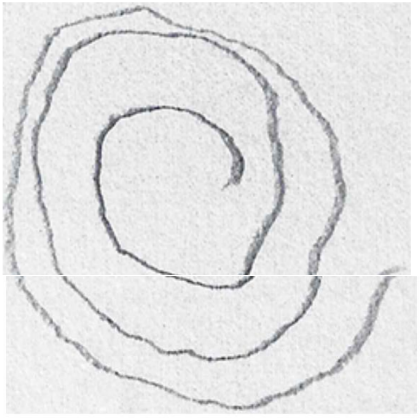


Result: : You are healthy!!

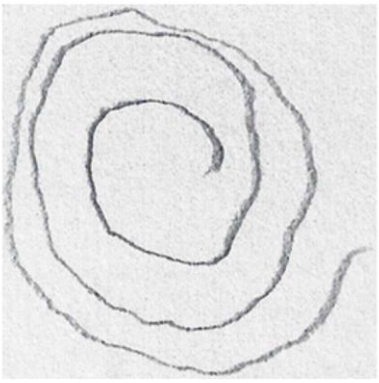
Parkinson Disease Detection

Parkinson Disease Detection

Upload image

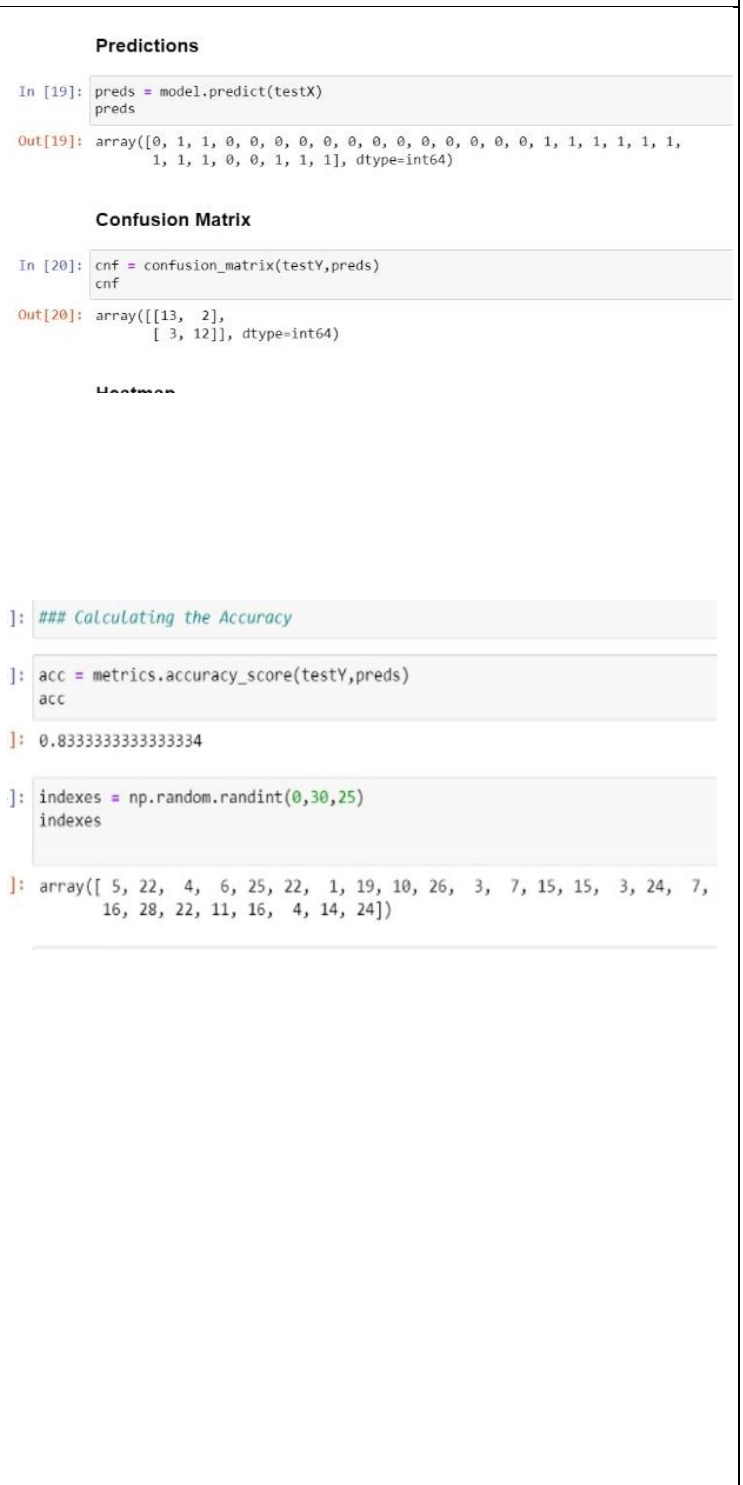


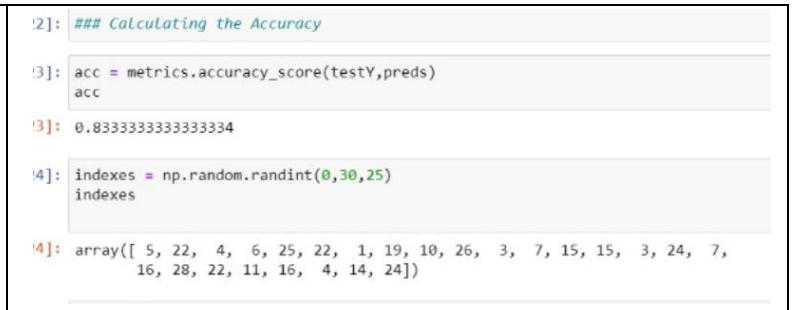
Predict!



Result: parkinson disease detected, please visit nearby doctor

9.1 Performance Metrics

S.No.	Parameter	Values	Screenshot
1.	Metrics	**Regression Model:** MAE -0.084746 , MSE - 0.084746 , RMSE - 0.291111 , R2 score - 0.656177 **Classification Model:** Confusion Matrix, Accuray Score- 0.8391525423728814 & Classification Report	

2.	Tune the Model	Hyperparameter Tuning Validation Method – [0.95744681 0.91489362 0.93617021 0.91489362 0.85106383]	

CHAPTER 10

ADVANTAGES &DISADVANTAGES

Advantages:

- Major advantage of this tool is that it helps to detect the Parkinsons disease from home.
- It is also easy to use and is quicker to detect Parkinsons disease.
- It can also be performed in any place and everywhere.

Disadvantages:

- The person's who doesn't able to speak cannot detect Parkinsons using this tool

CHAPTER 11

CONCLUSION

Parkinson's disease has been plaguing humans for thousands of years and was described in detail in ancient medical writings. Early sufferers from it effects were treated with varying results by a variety of plant-based treatments, some of which are still in use today. With the discovery of dopamine in the twentieth century and the subsequent development of dopamine replacement therapy, plus surgical techniques such as deep brain stimulation (DBS), many of the debilitating symptoms are now successfully treated—at least for a time.

The hope is to find the cause of PD, along with treatments that stop the disease from progressing. Of particular interest, PD research is uncovering what may turn out to be a common pathophysiologic mechanism underlying dementia and PD. For now, healthcare providers must continue to educate themselves about currently available treatments and hope for better alternatives in the near future.

CHAPTER 12

FUTURE SCOPE

- The tool can be made more accurate by adding even more algorithms.
- The tool can be not only detected by voice also by image and also Gait detection.
- Can add and get more personal information and past medical information.
- Can add more security and authentication.

CHAPTER 13

APPENDIX

Source code

main.py

```
import
pickle
        import sklearn
        from flask import Flask, render_template,request,redirect,url_for,flash
        from flask_bootstrap import Bootstrap
        from flask_sqlalchemy import SQLAlchemy
        from sqlalchemy.orm import relationship
        from flask_wtf import FlaskForm
        from werkzeug.utils import secure_filename
        from wtforms import StringField, SubmitField,FloatField,IntegerField
        from wtforms.validators import DataRequired
        from werkzeug.security import generate_password_hash,check_password_hash
        import os
        import cv2
        from skimage import feature
        from flask_login import login_user,logout_user,LoginManager,UserMixin,current_user,login_required
        app = Flask(__name__)
        app.config['SECRET_KEY'] = '8BYkEfBA6O6donzWlSihBXox7C0sKR6b'
        app.config['SQLALCHEMY_DATABASE_URI']='sqlite:///database.db'
        app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = False
        db = SQLAlchemy(app)
        Bootstrap(app)
        login_manager = LoginManager()
        login_manager.init_app(app)
        class users(UserMixin,db.Model):
            id = db.Column(db.Integer,primary_key=True)
            email= db.Column(db.String(200),nullable=False)
            password = db.Column(db.String(300),nullable=False)
            name = db.Column(db.String(100),nullable=False)
        @login_manager.user_loader
        def user_load(id):
            return users.query.get(int(id))
        @app.route("/")
        def home():
            return render_template("index1.html")
        @app.route("/register",methods=['GET','POST'])
        def register():
            if request.method == 'POST':
               if users.query.filter_by(email=request.form['email']).first():
                    flash('User already registered')
```

```
            return redirect(url_for('login'))
        else:
         password =
generate_password_hash(request.form['password'],method="pbkdf2:sha256",salt_length=8)
         user = users(
             email = request.form['email'],
             password = password,
             name = request.form['name']
         )
         db.session.add(user)
         db.session.commit()
         return redirect(url_for('home'))
    return render_template('register.html')
@app.route("/login",methods=['GET','POST'])
def login():
    if request.method == 'POST':
         email= request.form['email']
         password = request.form['password']
         k=users.query.filter_by(email=email).first()
         if not k:
             flash('User not registered')
             return redirect(url_for('login'))
         elif check_password_hash(k.password,password):
             login_user(k)
             return redirect(url_for('model'))
         else:
             flash('Wrong password')
             return redirect(url_for('login'))
    return render_template('login.html')
@app.route("/logout")
def logout():
    logout_user()
    return redirect(url_for('home'))
@app.route("/parkinson")
def model():
    return render_template('index.html')
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
@app.route('/predict', methods=['GET', 'POST'])
def upload():
    if request.method == 'POST':
         f = request.files['file']  # requesting the file
         basepath = os.path.dirname(os.path.realpath('__file__'))  # storing the file directory
         filepath = os.path.join(basepath, "uploads", f.filename)  # storing the file in uploads
```

```
folder
        f.save(filepath)
        image = cv2.imread(filepath)
        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)
        model = pickle.loads(open("C:/IBM PROJECT\parkinson_spiral.sav", "rb").read())
        preds= model.predict([features])
        ls = ["healthy", "parkinson"]
        result = ls[preds[0]]
        if(result=="healthy"):
            result=": You are healthy!!"
        elif(result=="parkinson"):
            result+=" disease detected, please visit nearby doctor "
        return result
    return None
admin=[1]
if __name__ == '__main__':
    app.run(debug=True)
```

GitHub Link: [IBM-Project-48434-1660807342]