Detecting Parkinson’s Disease Using Machine Learning

SUBMITTED BY

Y.PRAGNYA	113319205031
R.ABENESH	113319205001
S.S.KAMALESH	113319205019
T.KAVYA	113319205020

BACHELOR OF TECHNOLOGY IN INFORMATION TECHNOLOGY

INDEX

1. **INTRODUCTION**
    1.1. Project Overview
    1.2. Purpose
2. **LITERATURE SURVEY**
    2.1. Existing problem
    2.2. References
    2.3. Problem Statement Definition
3. **IDEATION & PROPOSED SOLUTION**
    3.1. Empathy Map Canvas
    3.2. Ideation & Brainstorming
    3.3. Proposed Solution
    3.4. Problem Solution fit
4. **REQUIREMENT ANALYSIS**
    4.1. Functional requirement
    4.2. Non-Functional requirements
5. **PROJECT DESIGN**
    5.1. Data Flow Diagrams
    5.2. Solution & Technical Architecture
    5.3. User Stories
6. **PROJECT PLANNING & SCHEDULING**
    6.1. Sprint Planning & Estimation
    6.2. Sprint Delivery Schedule
    6.3. Reports from JIRA
7. **CODING & SOLUTIONING (Explain the features added in the project along with code)**
    7.1. Feature 1
    7.2. Feature 2

7.3. Database Schema (if Applicable)

8. **TESTING**

8.1. Test Cases

8.2. User Acceptance Testing

9. **RESULTS**

9.1. Performance Metrics

10. **ADVANTAGES & DISADVANTAGES**

11. **CONCLUSION**

12. **FUTURE SCOPE**

13. **APPENDIX**

Source Code

GitHub & Project Demo Link

1.INTRODUCTION

1.1 PROJECT OVERVIEW

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

The researchers found that the drawing speed was slower and the pen pressure is lower among Parkinson's patients. One of the indications of Parkinson's is tremors and rigidity in the muscles, making it difficult to draw smooth spirals and waves. It is possible to detect Parkinson's disease using the drawings alone instead of measuring the speed and pressure of the pen on paper. Our goal is to quantify the visual appearance(using HOG method) of these drawings and then train a machine learning model to classify them. In this project, We are using, Histogram of Oriented Gradients (HOG) image descriptor along with a Random Forest classifier to automatically detect Parkinson's disease in hand-drawn images of spirals and waves.

1.2 PURPOSE

Parkinson's disease (PD) is a long-term degenerative disorder of the central nervous system that mainly affects the motor system. The most obvious early symptoms are tremor, rigidity, slowness of movement, and difficulty with walking ,but cognitive and behavioral problems may also occur.The risk of developing Parkinson's disease naturally increases with age, and the average

age at which it starts is 60 years old. The main aim of this application is early prediction and proper treatments can possibly stop or slow progression of this disease to end stage.

2.LITERATURE SURVEY

2.1 EXISTING PROBLEM

There isn't a specific test to diagnose Parkinson's disease. A doctor trained in nervous system conditions (neurologist) will diagnose Parkinson's disease based on your medical history, a review of your signs and symptoms, and a neurological and physical examination.

Your doctor may suggest a specific single-photon emission computerized tomography (SPECT) scan called a dopamine transporter (DAT) scan. Although this can help support the suspicion that you have Parkinson's disease, it is your symptoms and neurological examination that ultimately determine the correct diagnosis. Most people do not require a DAT scan.

Your health care provider may order lab tests, such as blood tests, to rule out other conditions that may be causing your symptoms.

Imaging tests — such as an MRI, ultrasound of the brain and PET scans — also may be used to help rule out other disorders. Imaging tests aren't particularly helpful for diagnosing Parkinson's disease.

In addition to your examination, your health care provider may give you carbidopa-levodopa (Rytary, Sinemet, others), a Parkinson's disease medication. You must be given a sufficient dose to show the benefit, as low doses for a day or two aren't reliable. Significant improvement with this medication will often confirm your diagnosis of Parkinson's disease.

Sometimes it takes time to diagnose Parkinson's disease. Health care providers may recommend regular follow-up appointments with neurologists trained in movement disorders to evaluate your condition and symptoms over time and diagnose Parkinson's disease.

2.2 REFERENCES

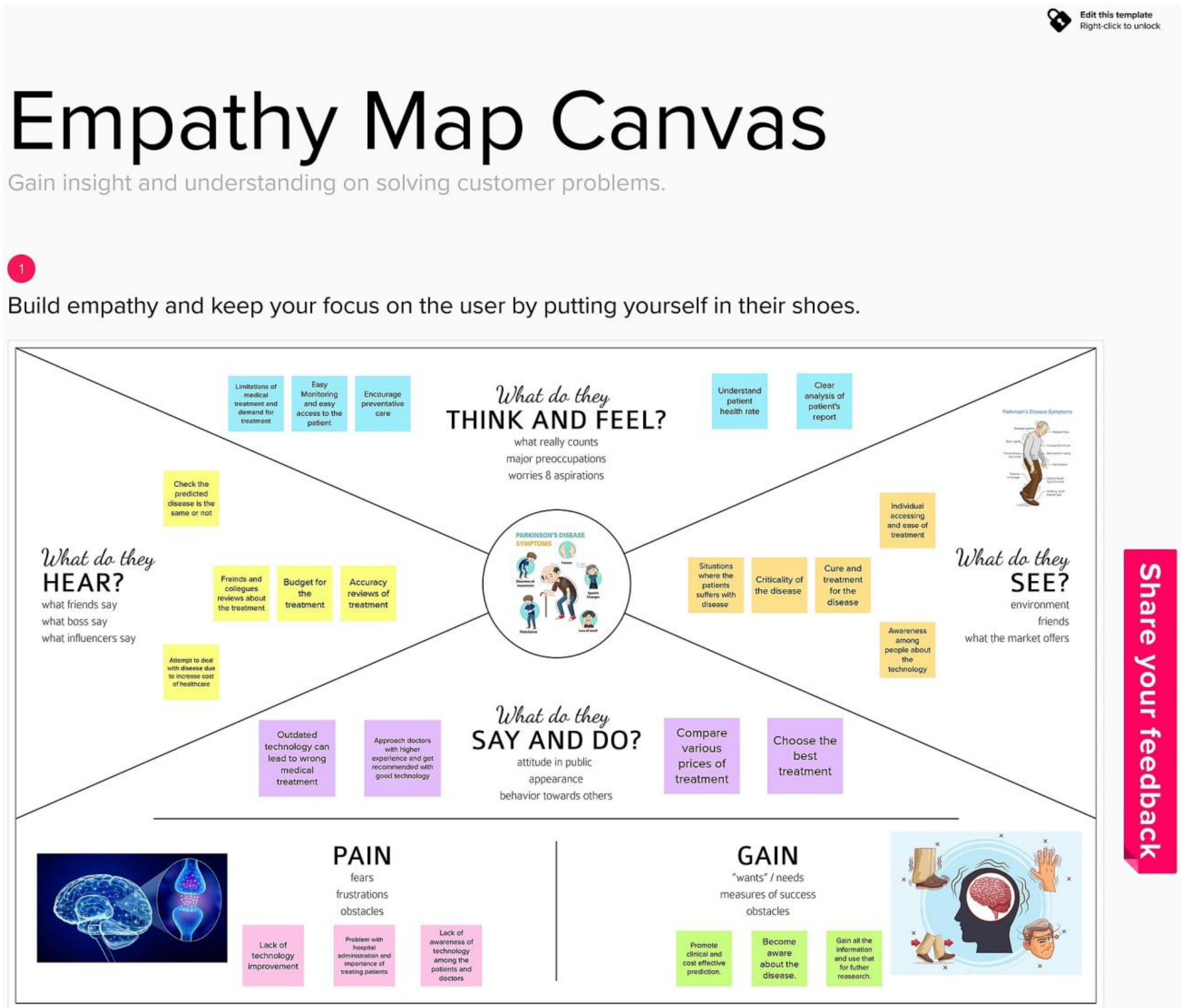
- Abiyev, R. H., and Abizade, S. (2016). Diagnosing Parkinson's diseases using fuzzy neural system. Comput. Mathe. Methods Med. 2016:1267919. doi: 10.1155/2016/1267919
- Abos, A., Baggio, H. C., Segura, B., Campabadal, A., Uribe, C., Giraldo, D. M., et al. (2019). Differentiation of multiple system atrophy from Parkinson's disease by structural connectivity derived from probabilistic tractography. Sci. Rep. 9:16488. doi: 10.1038/s41598-019-52829-8
- Abujrida, H., Agu, E., and Pahlavan, K. (2017). “Smartphone-based gait assessment to infer Parkinson's disease severity using crowdsourced data,” in 2017 IEEE Healthcare Innovations and Point of Care Technologies (HI-POCT) (Bethesda, MD), 208–211. doi: 10.1109/HIC.2017.8227621
- Adeli, E., Shi, F., An, L., Wee, C.-Y., Wu, G., Wang, T., et al. (2016). Joint featuresample selection and robust diagnosis of Parkinson's disease from MRI data. NeuroImage 141, 206–219. doi: 10.1016/j.neuroimage.2016.05.054
- Braak, H., Del Tredici, K., Rüb, U., De Vos, R. A., Steur, E. N. J., and Braak, E. (2003). Staging of brain pathology related to sporadic Parkinson's disease. Neurobiol. Aging 24, 197–211. doi: 10.1016/S0197-4580(02)00065-9
- Caramia, C., Torricelli, D., Schmid, M., Munoz-Gonzalez, A., Gonzalez-Vargas, J., Grandas, F., et al. (2018). IMU-based classification of Parkinson's disease from gait: a sensitivity analysis on sensor location and feature selection. IEEE J. Biomed. Health Inf.

22, 1765–1774. doi: 10.1109/JBHI.2018.2865218

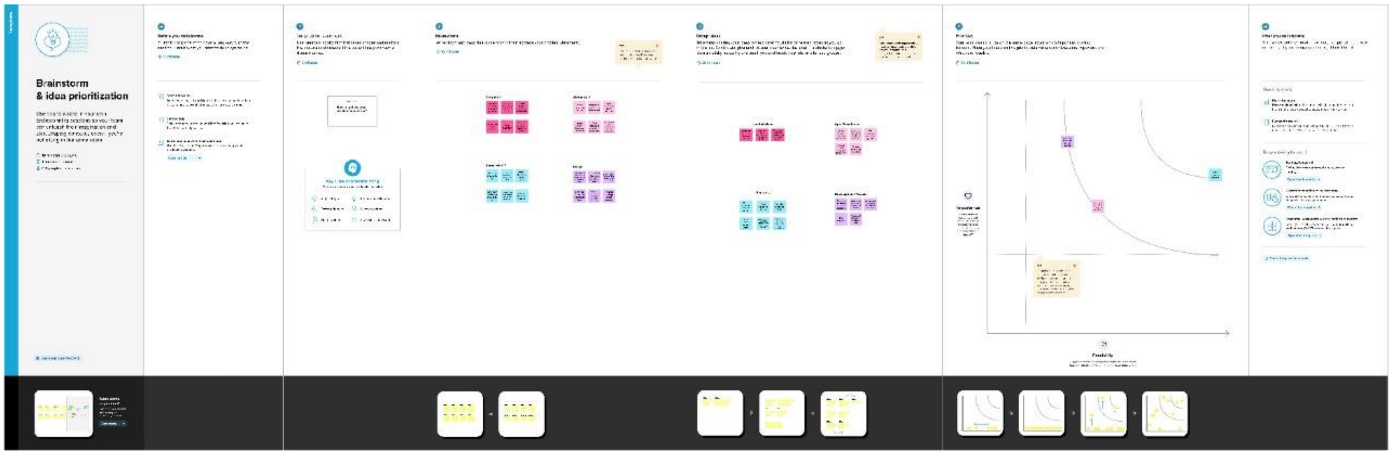
2.3 PROBLEM STATEMENT DEFINITION

3.IDEATION & PROPOSED SOLUTION

3.1 EMPATHY MAP CANVAS



3.2 IDEATION & BRAINSTROMING

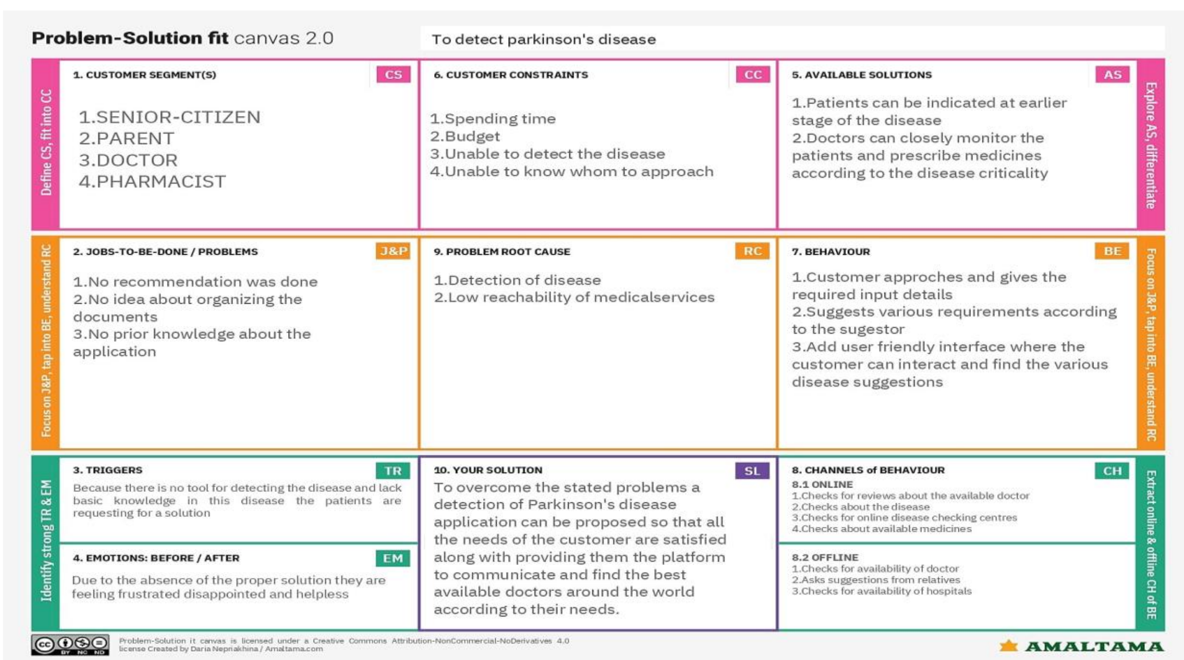


3.3 PROPOSED SOLUTION

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	User needs an application that takes images and analyses the drawing speed and detects if the patient has Parkinson's disease or not
2.	Idea / Solution description	Our goal is to quantify the visual appearance(using HOG method) of Parkinson's disease using the drawings and then train a machine learning model to classify them. In this project, We are using, Histogram of Oriented Gradients (HOG) image descriptor along with a Random Forest classifier to automatically detect Parkinson's disease in hand-drawn images of spirals and waves.
3.	Novelty / Uniqueness	Using various algorithms we have simplified the process of detecting Parkinson's disease by analyzing various images of drawings made by the patients and the pen paper pressure of it
4.	Social Impact / Customer Satisfaction	People are lacking knowledge about these kinds of disease and most of it goes unidentified at the initial stage Parkinson's cannot be cured, but early detection along with proper medication can

		significantly improve symptoms and quality of life.
5.	Business Model (Revenue Model)	As diseases like Parkinson's are rarely concerned and identified by the patients, creating an application and making people aware will cause a great change leading to the most health benefits among the users. As more people register the subscription methods can be included and the revenue for further improvisation can be achieved. The images uploaded by the users can be used for research purposes by scientists and can also generate revenue for the business.
6.	Scalability of the Solution	High-speed algorithms are used here, to analyze the humongous amount of data and retrieve the accuracy of the result.

3.4 PROBLEM SOLUTION FIT



4.REQUIREMENT ANALYSIS

4.1 FUNCTIONAL REQUIREMENT

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Data collection and storage	The data collected from the registration phase is stored and diagnosis is provided according to their state of disease.
FR-4	Image reccomendation	All the possible diagnosis are provided and are further steps for reducing the risk is implemented

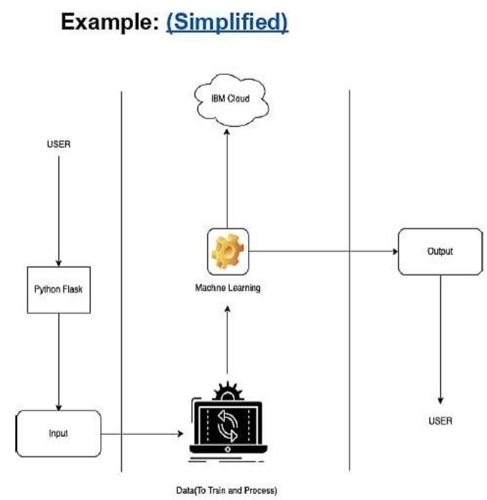
4.2 NON-FUNCTIONAL REQUIREMENT

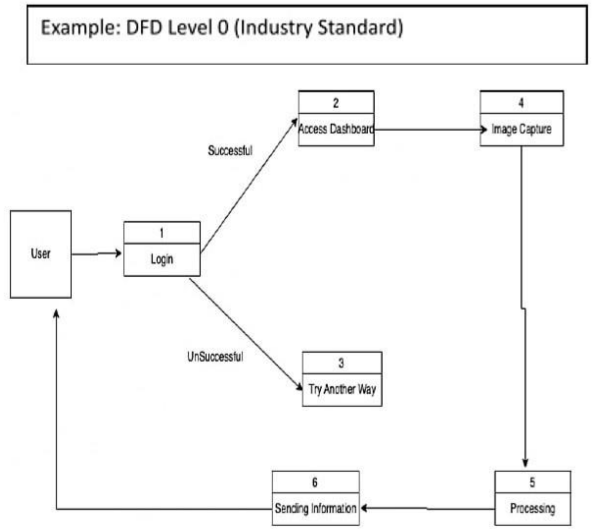
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The dashboard of the application provides all the required tests to be taken and the images to be uploaded allowing the user to give all his details and know the criticality of the disease
NFR-2	**Security**	The security of the application is designed in such a way that the user can store their sensitive information and use that for further diagnosis

NFR-3	**Reliability**	This application is highly reliable as it provides various disease curability suggestions instant doctor application and various other functions
NFR-4	**Performance**	The loading time of application is very reliable allowing it to be highly usable and user-friendly
NFR-5	**Availability**	The availability of service and the suggestion is clearly provided allowing it to be the best application for detecting the disease and its diagnosis.
NRF-6	**Scalability**	The cloud storage is highly scalable allowing it to expand the application according to the needs of the users registering allowing it to be one of the flexible and scalable unit

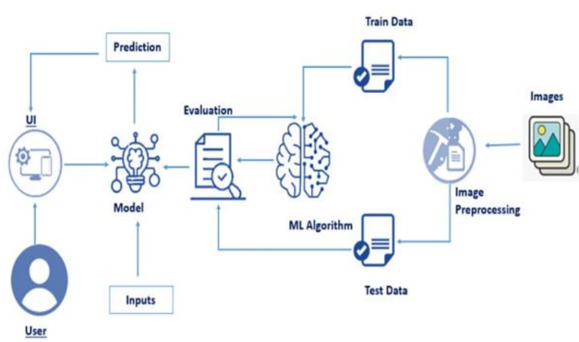
5.PROJECT DESIGN

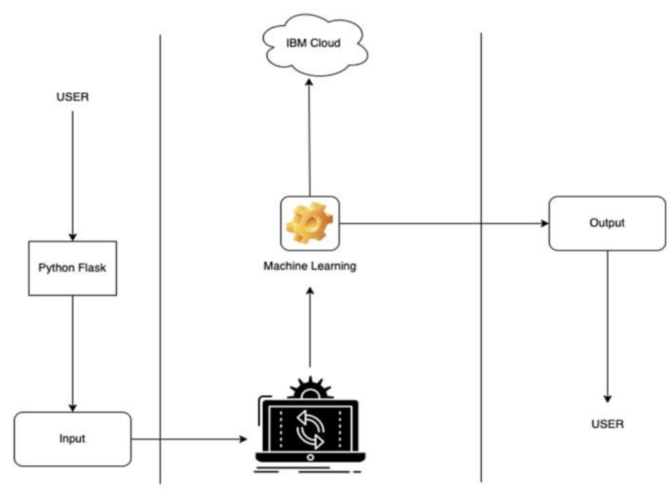
5.1 DATA FLOW DIAGRAM



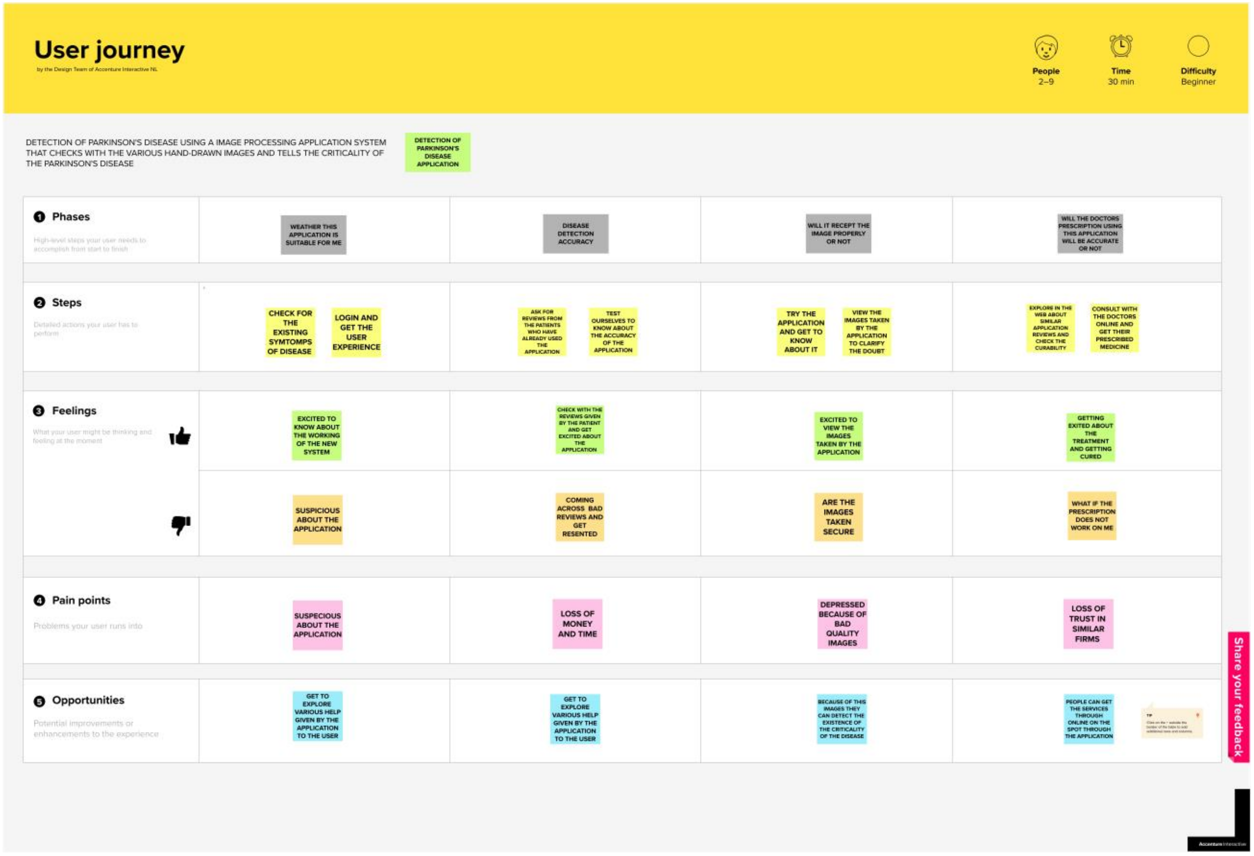


5.2 SOLUTION & TECHNICAL ARCHITECTURE





5.3 USER STORIES



6.PROJECT PLANNING AND 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 email, password, and confirming my password.	2	high	3
Sprint -1		USN-2	As a user, I will receive a confirmation email once I have registered for the application	1	high	1
Sprint -2		USN-3	As a user, I can register for the application through Facebook	2	low	1
Sprint -1		USN-4	As a user, I can register for the application through google account	2	high	1
Sprint -1	LOGIN	USN-5	As a user, I can log into the application by entering email & password	1	high	1
Sprint-1	DASHBOARD	USN-6	As a customer I can check with all the clothes available on the website and choose the ones which I want	3	high	3

Sprint-4	Customer support	USN-7	As a user I want to contact with the customer support when there is any query with	2	low	2

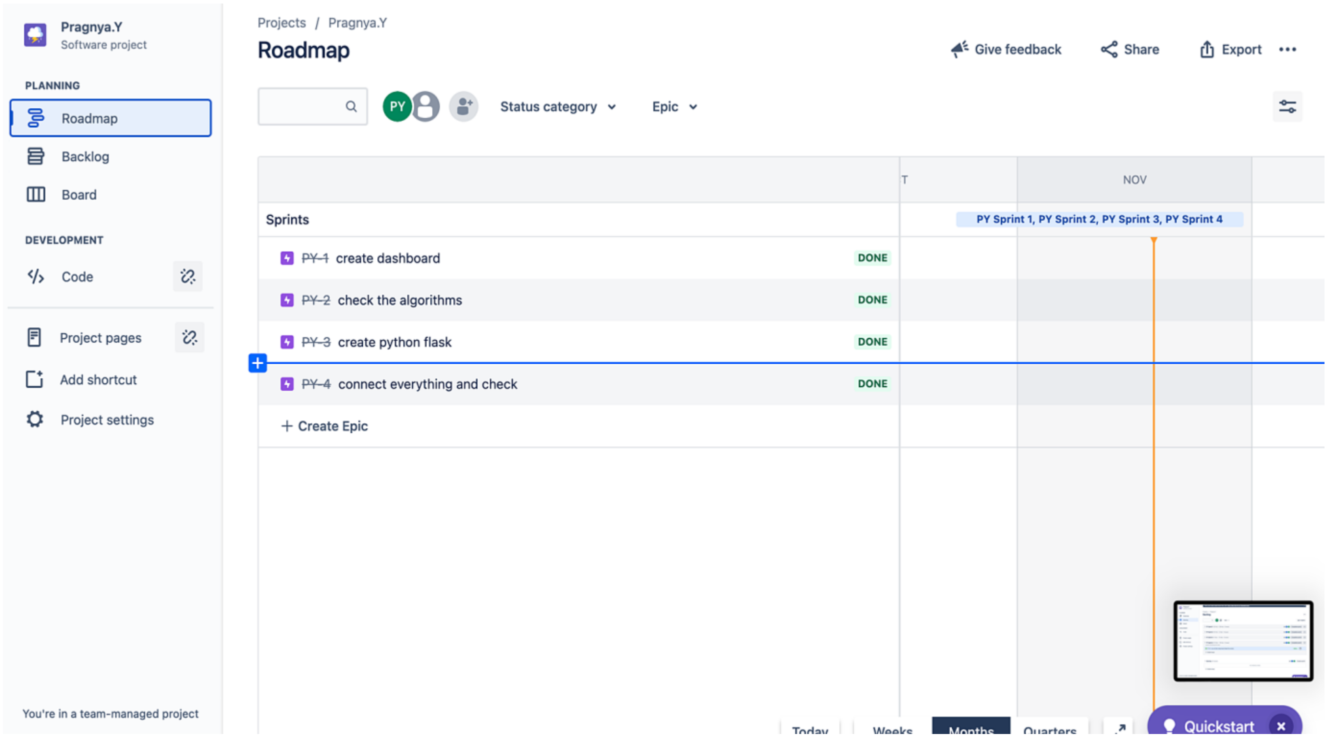
Sprint-1	User details display	USN-8	As a customer I should be able to see all my given details filled with the registration process	2	high	2
Sprint-2	algorithm	USN-9	As a customer, I should be able to get the perfect prescription for the disease criticality and get the correct doctor details	5	high	4
Sprint-2		USN-10	As a customer I should be updated with various best available	2	medium	2
Sprint-3	IBM watson for storage and organization	USN-11	As a customer I should be able to give my images and predict the out come and the prescription for my disease criticality	3	high	2
Sprint-4	Cart management	USN-12	As a customer I can add and manage all the chosen products and place my order	5	high	4

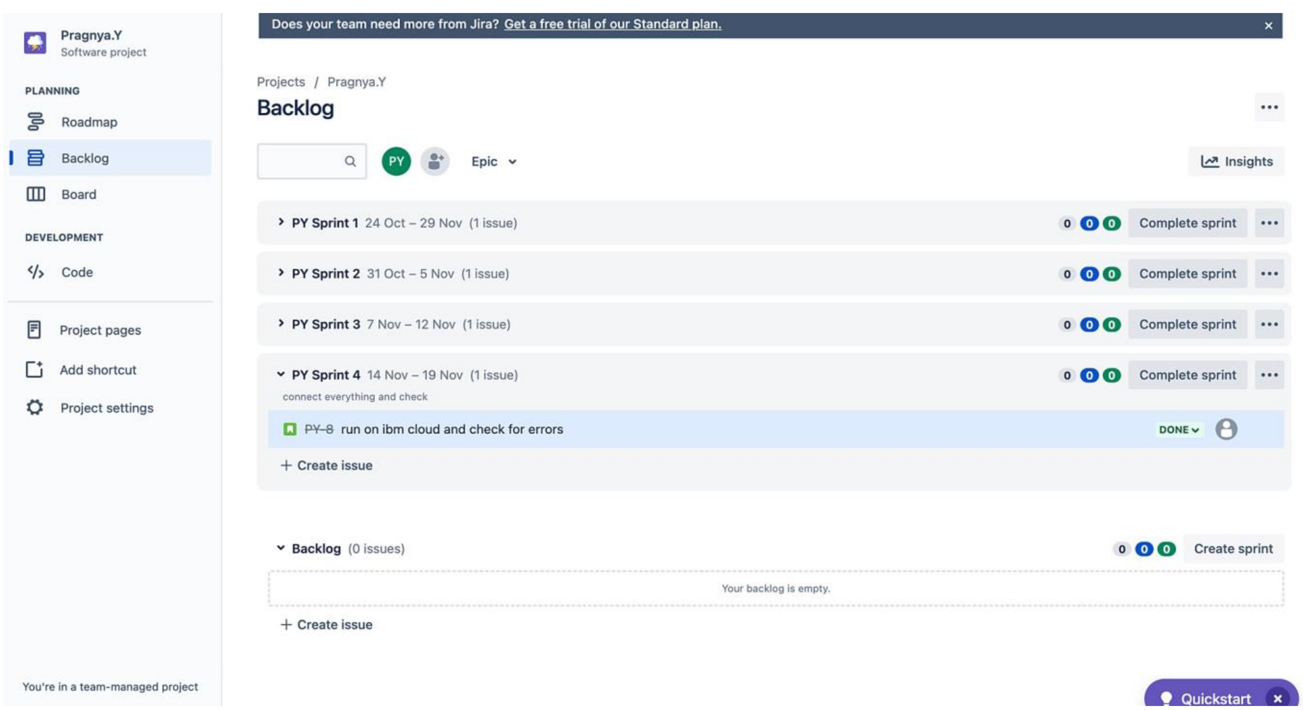
6.2 SPRINT DELIVERY SCHEDULE

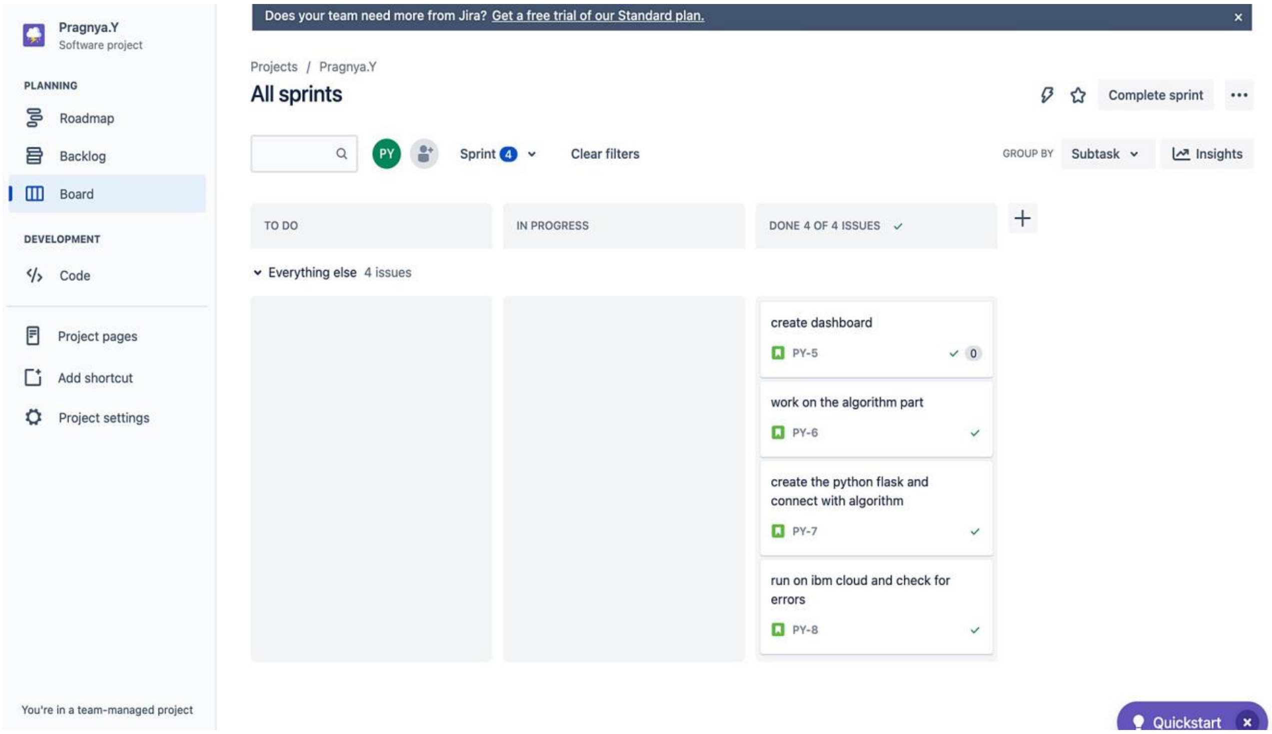
Sprint	**Total Story Points**	**Duration**	**Sprint Start Date**	**Sprint End Date (Planned)**
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022

Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022

6.3 REPORTS FROM JIRA



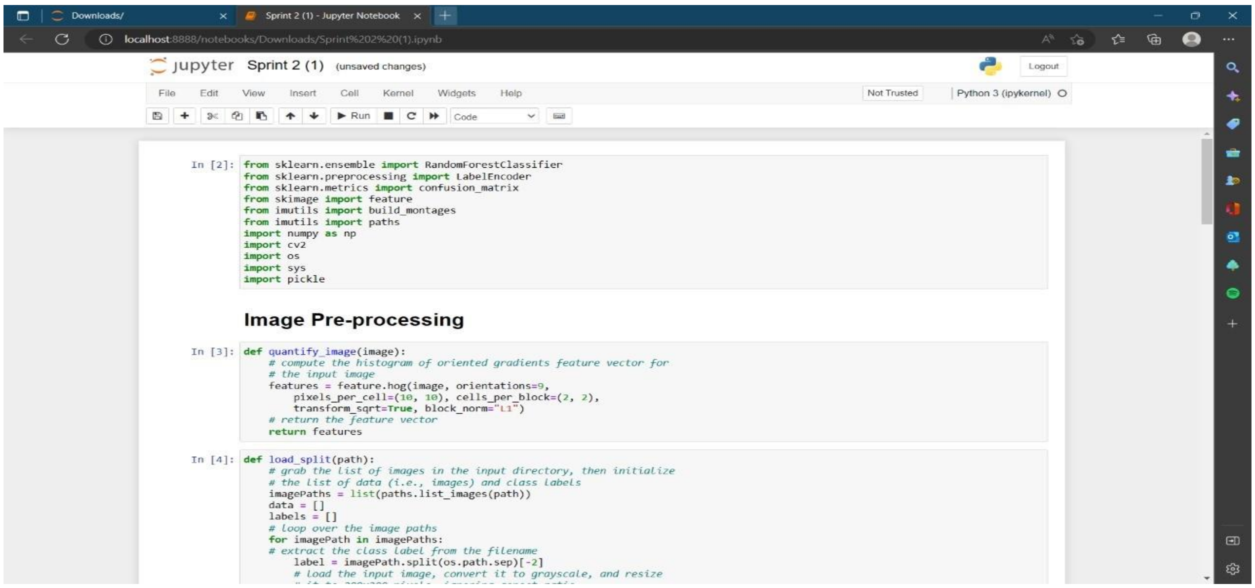


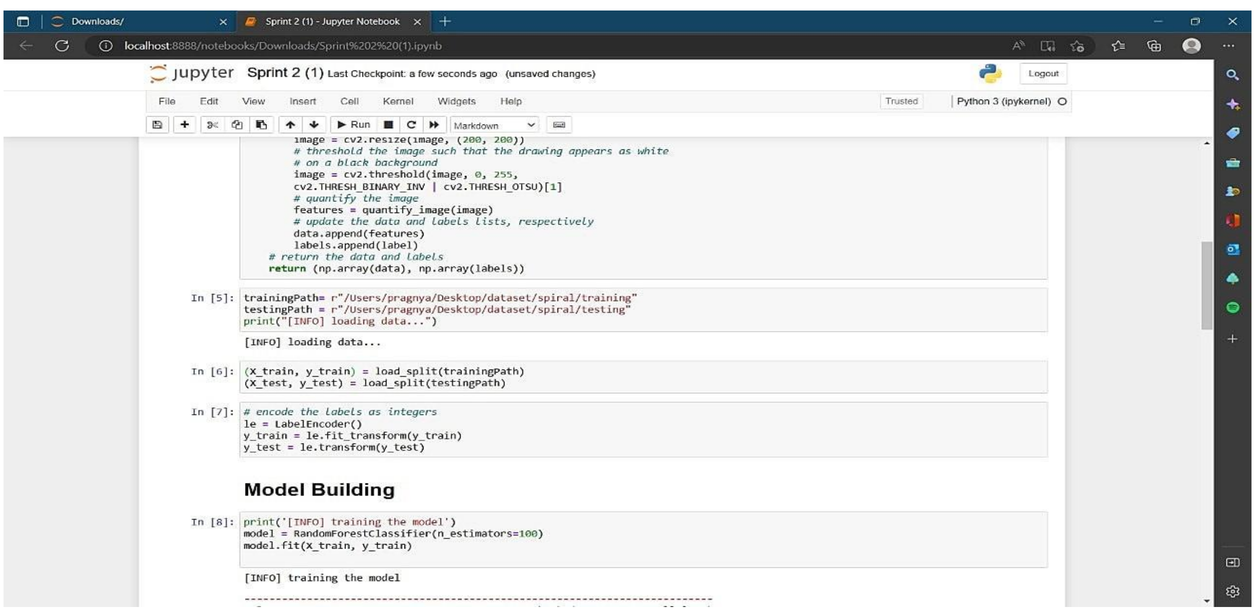

Pragnya.Y
Software project
PLANNING
Roadmap
Backlog
Board
DEVELOPMENT
Code
Project pages
Add shortcut
Project settings
You're in a team-managed project
Does your team need more from Jira? Get a free trial of our Standard plan.
Projects / Pragnya.Y
All sprints
Complete sprint
PY
Sprint 4
Clear filters
GROUP BY
Subtask
Insights
TO DO
IN PROGRESS
DONE 4 OF 4 ISSUES
Everything else 4 issues
create dashboard
PY-5
0
work on the algorithm part
PY-6
create the python flask and connect with algorithm
PY-7
run on ibm cloud and check for errors
PY-8
Quickstart

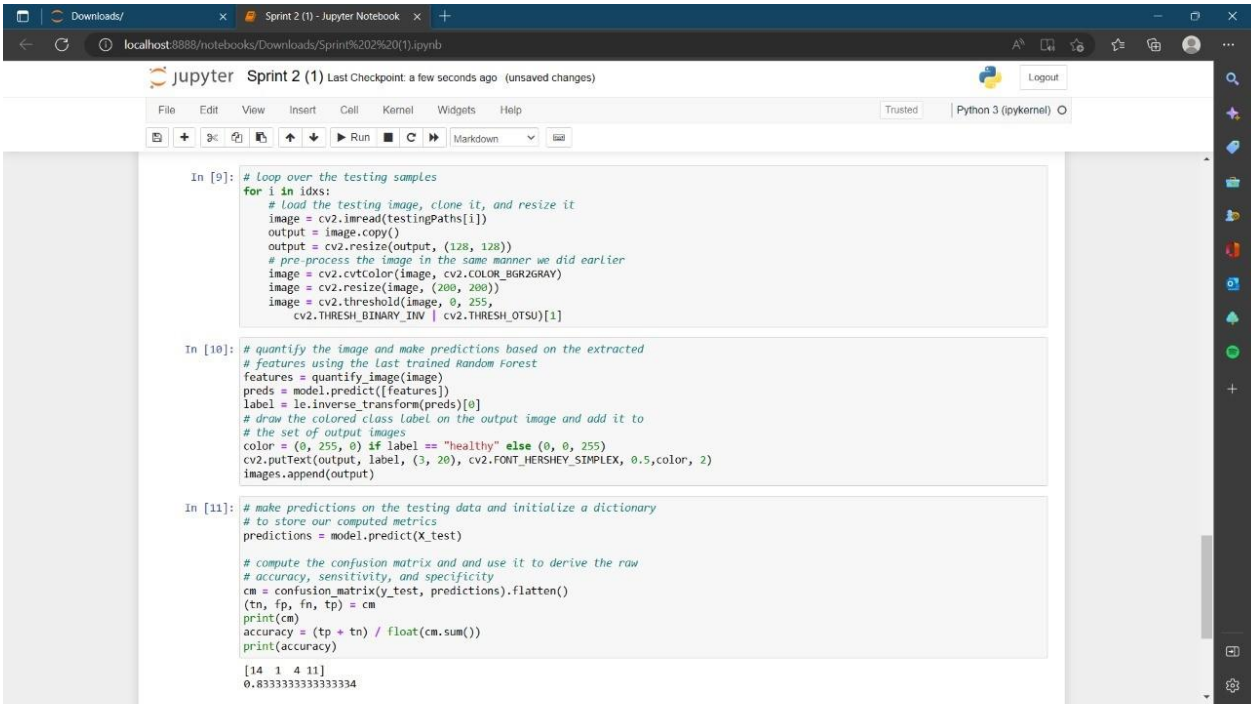
7.CODING AND SOLUTIONING

7.1 FEATURE 1

We have loaded the dataset and worked on the random forest algorithm to predict the parkinson's disease.

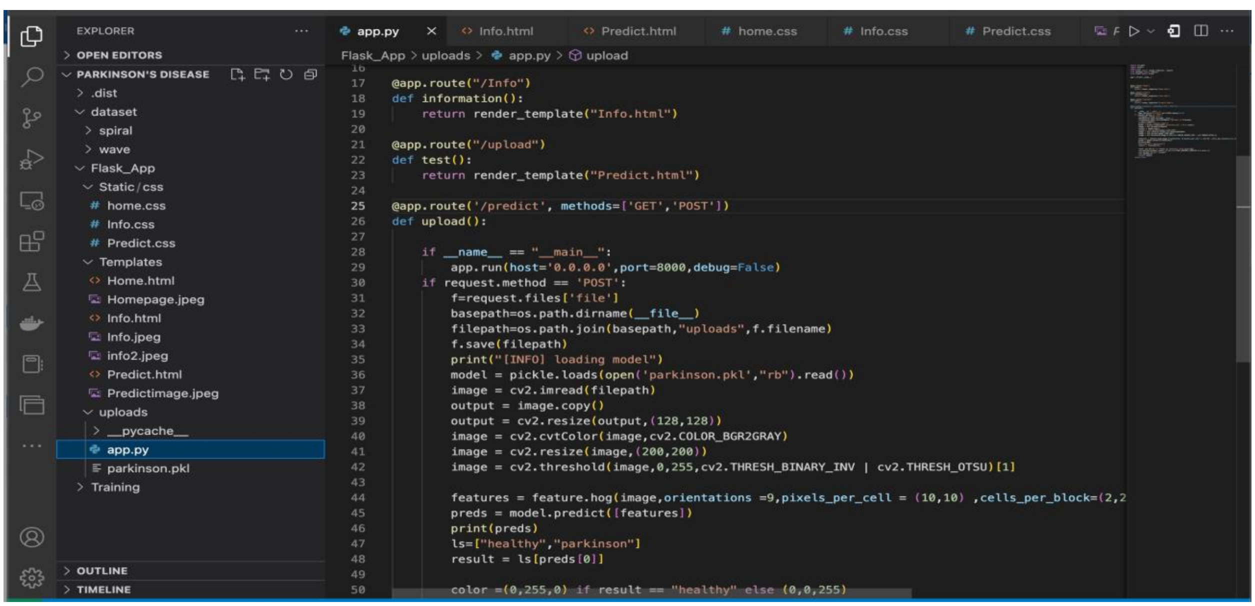


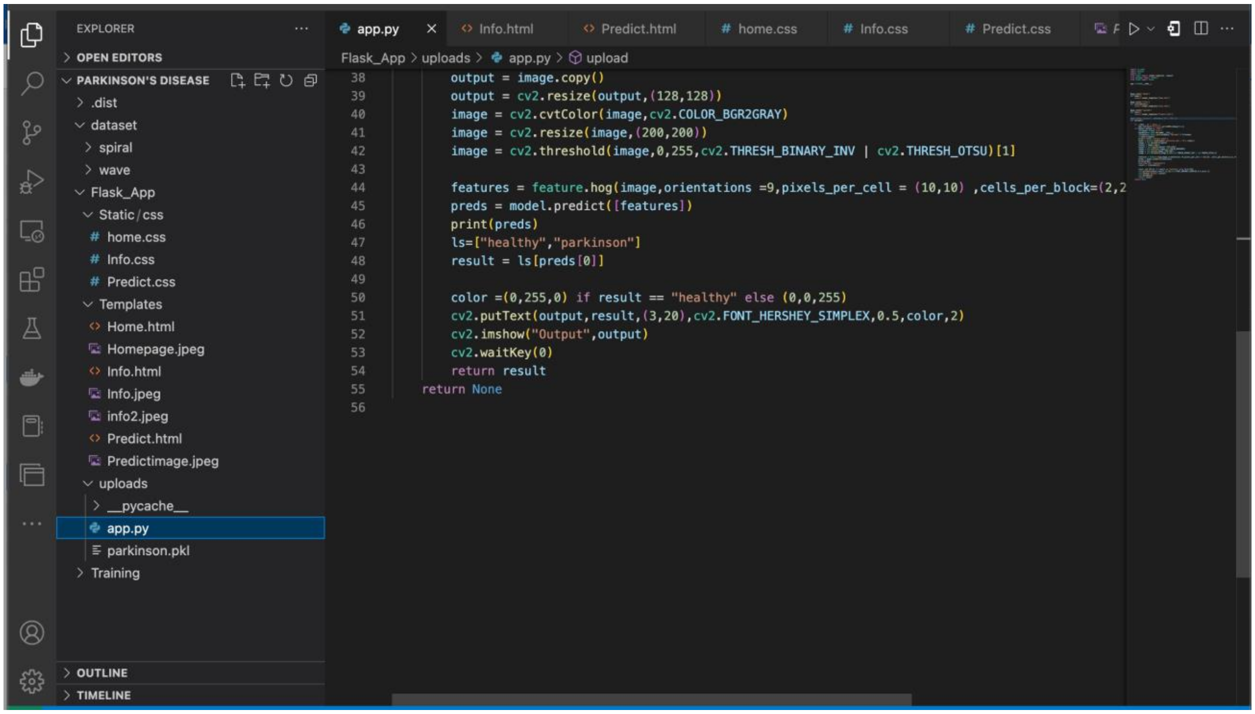




7.2 FEATURE 2

We have created a file called app.py where the flask code is implemented and it is connected with the algorithms and the dataset.




EXPLORER
OPEN EDITORS
PARKINSON'S DISEASE
.dist
dataset
spiral
wave
Flask_App
Static / css
home.css
Info.css
Predict.css
Templates
Home.html
Homepage.jpeg
Info.html
Info.jpeg
info2.jpeg
Predict.html
Predictimage.jpeg
uploads
__pycache__
app.py
parkinson.pkl
Training
OUTLINE
TIMELINE
app.py
Info.html
Predict.html
home.css
Info.css
Predict.css
Flask_App > uploads > app.py > upload
output = image.copy()
output = cv2.resize(output,(128,128))
image = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)
image = cv2.resize(image,(200,200))
image = cv2.threshold(image,0,255,cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]
features = feature.hog(image,orientations =9,pixels_per_cell = (10,10) ,cells_per_block=(2,2
preds = model.predict([features])
print(preds)
ls=["healthy","parkinson"]
result = ls[preds[0]]
color =(0,255,0) if result == "healthy" else (0,0,255)
cv2.putText(output,result,(3,20),cv2.FONT_HERSHEY_SIMPLEX,0.5,color,2)
cv2.imshow("Output",output)
cv2.waitKey(0)
return result
return None

8.TESTING

8.1 TEST CASES

			Date	18-Nov-22									
			Team ID	PNT2022TMID24541									
			Project Name	Detecting Parkinsons Disease using Machine Learning									
			Maximum Marks	4 marks									
Test case ID	Feature Type	Component	Test Scenario	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation (Y/N)	BUG ID	Executed By
Main page	Functional	Home Page	Verify whether the user is able to access their images from th	1.Enter URL and click go 2.Check whether it is working	http://127.0.0.1:5500/Flask_App/Templates/Predict.html	to upload image	Working as expected	Pass				Pragnya . y
Main page	UI	Home Page	Verify the UI elements in the pages	1.Enter URL and click go 2.Click on all the buttons in the navbar and check	http://127.0.0.1:5500/Flask_App/Templates/Predict.html	to check for redirecting of pages	Working as expected	Pass				Abenesh R
Main page	Functional	Home page	Verify the user is able to read the contents properly	1.Enter URL and click go 2.Check all the pages for typo error	http://127.0.0.1:5500/Flask_App/Templates/Predict.html	User should be able to read with difficulty	Working as expected	Pass				Kavya.T
Main page	Functional	Home page	Verify whether the user is able to get the accurate output	1.Enter URL and click go 2.Check the buttons are correctly working and the code is working	http://127.0.0.1:5500/Flask_App/Templates/Predict.html	User should be able to get the correct output	Working as expected	Pass				Kamalesh SS

8.2 USER ACCEPTANCE TESTING

Acceptance Testing
UAT Execution & Report Submission

Date	18 November 2022
Team ID	PNT2022TMID24541
Project Name	Detecting Parkinsons Disease using Machine Learning
Maximum Marks	4 Marks

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the Detecting Parkinsons Disease using Machine Learning project at the time of the release to User Acceptance Testing (UAT).

2. Defect Analysis

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

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	8	6	1	4	20
Duplicate	2	0	2	0	4
External	2	2	0	2	6
Fixed	11	2	4	20	37
Not Reproduced	0	0	1	0	1
Skipped	0	0	1	1	2
Won't Fix	0	5	2	1	8
Totals	24	14	13	26	77

3. 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	7	0	0	7
Client Application	51	0	0	51
Security	2	0	0	2
Outsource Shipping	3	0	0	3
Exception Reporting	9	0	0	9
Final Report Output	4	0	0	4
Version Control	2	0	0	2

9.RESULT

9.1 PERFORMANCE MATRICS

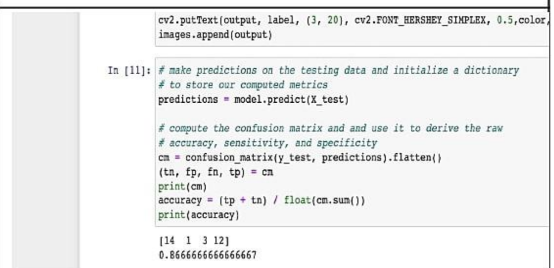
Project Development Phase
Model Performance Test

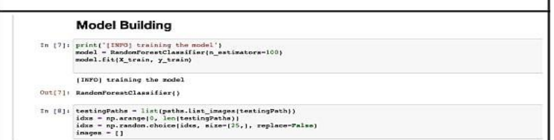
Date	18 November 2022
Team ID	PNT2022TMID24541
Project Name	Detecting Parkinsons Disease using Machine Learning
Maximum Marks	10 Marks

Model Performance Testing:

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot

1.	Metrics	**Regression Model:**
		MAE - , MSE - , RMSE - ,
		R2 score -
		
		Classification Model:
		Confusion Matrix - ,
		Accuray Score- &
		Classification Report -
			
			cv2.putText(output, label, (3, 20), cv2.FONT_HERSHEY_SIMPLEX, 0.5,color,
			images.append(output)
			In [11]: # make predictions on the testing data and initialize a dictionary
			# to store our computed metrics
			predictions = model.predict(X_test)
			# compute the confusion matrix and and use it to derive the raw
			# accuracy, sensitivity, and specificity
			cm = confusion_matrix(y_test, predictions).flatten()
			(tn, fp, fn, tp) = cm
			print(cm)
			accuracy = (tp + tn) / float(cm.sum())
			print(accuracy)
			[14 1 3 12]
			0.8666666666666667

2.	Tune the Model	Hyperparameter Tuning - Validation Method -	 Model Building In [7]: print('[INFO] training the model') model = RandomForestClassifier(n_estimators=100) model.fit(X_train, y_train) [INFO] training the model Out[7]: RandomForestClassifier() In [8]: testingPaths = list(paths.list_images(testingPath)) idxs = np.arange(0, len(testingPaths)) idxs = np.random.choice(idxs, size=(25,), replace=False) images = []

10. ADVANTAGES AND DISADVANTAGES

ADVANTAGES:

- Accurately detecting Parkinson's disease (PD) at an early stage is certainly indispensable for slowing down its progress and providing patients the possibility of accessing to disease-modifying therapy.
- This model makes it possible by predicting the disease using the images that has been uploaded by the patient.
- It is an user-friendly method for predicting the disease at the earlier stage .

DISADVANTAGES:

- The patient or the user should draw the image and upload it to check for the disease.
- The user can't connect with the doctors through this application.

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. Parkinson's disease affects the CNS of the brain and has yet no treatment unless it's detected early. Late detection leads to no treatment and loss of life. Thus its early detection is significant. For early detection of the disease, we utilized machine learning algorithms such as Random Forest Classifier. We checked our Parkinson disease data and find out that Random Forest Classifier is the best Algorithm to predict the onset of the disease which will enable early treatment and save a life.

12.FUTURE SCOPE

The main aim of this application is early prediction and proper

treatments can possibly stop or slow progression of this disease to end stage.Can generate revenue through direct customer. Can collaborate with health care sector and generate revenue from their customers.We will also enable the facility to connect the parkinson's affected patients to connect with the doctors virutally and get diagonsed . We will also enable the feature of drawing virtually and predicting rather than uploading the images manually.

13.APPENDIX

Source code:

```
from sklearn.ensemble import

RandomForestClassifier from sklearn.preprocessing

import LabelEncoder from sklearn.metrics import

confusion_matrix from skimage import feature from

imutils import build_montages from imutils import

paths import numpy as np import cv2 import os

import sys import pickle def quantify_image(image):

   # compute the histogram of oriented gradients feature vector for

   # the input image

   features = feature.hog(image, orientations=9,

      pixels_per_cell=(10, 10), cells_per_block=(2, 2),

      transform_sqrt=True, block_norm="L1")

  # return the feature

vector    return features

def load_split(path):

   # grab the list of images in the input directory, then initialize

   # the list of data (i.e., images) and class labels

   imagePaths = list(paths.list_images(path))

   data = []
```

```
    labels = []
    # loop over the image paths
    for imagePath in imagePaths:
    # extract the class label from the filename
        label = imagePath.split(os.path.sep)[-2]
        # load the input image, convert it to grayscale, and resize
        # it to 200x200 pixels, ignoring aspect ratio
        image = cv2.imread(imagePath)
        image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
        image = cv2.resize(image, (200, 200))
        # threshold the image such that the drawing appears as white
        # on a black background
        image = cv2.threshold(image, 0, 255,
        cv2.THRESH_BINARY_INV | cv2.THRESH_OTSU)[1]
        # quantify the image
      features = quantify_image(image)
        # update the data and labels lists, respectively
        data.append(features)
        labels.append(label)
    # return the data and labels     return (np.array(data),
np.array(labels)) trainingPath=
r"/Users/pragnya/Desktop/dataset/spiral/training"
```

```
testingPath =

r"/Users/pragnya/Desktop/dataset/spiral/testing"

print("[INFO] loading data...")

(X_train, y_train) = load_split(trainingPath)

(X_test, y_test) = load_split(testingPath) # encode the

labels as integers le = LabelEncoder() y_train =

le.fit_transform(y_train) y_test = le.transform(y_test)

print('[INFO] training the model') model =

RandomForestClassifier(n_estimators=100)

model.fit(X_train, y_train) testingPaths =

list(paths.list_images(testingPath)) idxs = np.arange(0,

len(testingPaths)) idxs = np.random.choice(idxs,

size=(25,), replace=False) images = []


# loop over the testing samples

for i in idxs:

    # load the testing image, clone it, and resize it

    image = cv2.imread(testingPaths[i])

    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]

# quantify the image and make predictions based on the extracted

# features using the last trained Random

Forest features = quantify_image(image)

preds = model.predict([features]) label =

le.inverse_transform(preds)[0]

# draw the colored class label on the output image and add it to

# the set of output images color = (0, 255, 0) if label == "healthy" else (0, 0, 255)

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

images.append(output)

# make predictions on the testing data and initialize a dictionary

# to store our computed metrics

predictions = model.predict(X_test)


# compute the confusion matrix and and use it to derive the raw

# accuracy, sensitivity, and specificity cm =

confusion_matrix(y_test, predictions).flatten()

(tn, fp, fn, tp) = cm print(cm)

accuracy = (tp + tn) / float(cm.sum())

print(accuracy)
```

```
# create a montage using 128x128 "tiles" with 5 rows and 5 columns

montage = build_montages(images, (128, 128), (5, 5))[0]


# show the output montage

cv2.imshow("Output", montage) cv2.waitKey()

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

APP.py :

```
import os.path

import pickle

import cv2 from

flask import

render_template,

request from

skimage import

feature from flask

import Flask


app = Flask(_name_)

@app.route("/Home")

def home():

   return render_template("Home.html")
```

```
@app.route("/Info")

def information():

   return render_template("Info.html")


@app.route("/upload")

def test():

   return render_template("Predict.html")


@app.route('/predict', methods=['GET','POST']) def

upload():


   if _name_ == "_main_":

      app.run(host='0.0.0.0',port=8000,debug=False)     if request.method == 'POST':

      f=request.files['file']

      basepath=os.path.dirname(_file_)

      filepath=os.path.join(basepath,"uploads",f.filename)

      f.save(filepath)

      print("[INFO] loading model")

      model = pickle.loads(open('parkinson.pkl',"rb").read())

      image = cv2.imread(filepath)

      output = image.copy()
```

```
        output = cv2.resize(output,(128,128))

        image = cv2.cvtColor(image,cv2.COLOR_BGR2GRAY)

        image = cv2.resize(image,(200,200))

                                                                    image  =

cv2.threshold(image,0,255,cv2.THRESH_BINARY_INV          | cv2.THRESH_OTSU)[1]


        features = feature.hog(image,orientations =9,pixels_per_cell = (10,10)

,cells_per_block=(2,2),transform_sqrt=True,block_norm="L1")

        preds = model.predict([features])

        print(preds)

        ls=["healthy","parkinson"]

        result = ls[preds[0]]

        color =(0,255,0) if result == "healthy" else (0,0,255)

        cv2.putText(output,result,(3,20),cv2.FONT_HERSHEY_SIMPLEX,0.5,color,2)

        cv2.imshow("Output",output)

        cv2.waitKey(0)

        return result

    return None
```

GITHUB: https://github.com/IBM-EPBL/IBM-Project-53397-1661399495.git

PROJECTDEMO:

https://drive.google.com/file/d/1hxZCXTuccahb10K0yhee83YxcG3bJM8Z/view?u

sp=share_link