PROJECT REPORT

DEEP LEARNING FUNDUS IMAGE ANALYSIS FOR EARLY DETECTION OF DIABETIC RECTINOPATHY

SUBMITTED BY

Team ID: PNT2022TMID52149

A.Thanga Abiya (962719106036)
T.Thanusha (962719106037)
A.Subitha (962719106035)
K.Yoka Muthu Nivethitha (962719106040)

Team ID	PNT2022TMID52149
Project Name	Deep Learning Fundus Image Analysis For Early Detection Of Diabetic Retinopathy

INTRODUCTION:

Diabetic retinopathy is an eye condition that can cause vision loss and blindness in people who have diabetes. It affects blood vessels in the retina (the light sensitive layer of tissue in the back of your eye). If you have diabetes, it's important to get a comprehensive dilated eye exam at least once a year. Diabetic retinopathy may not have any symptoms at first — but finding it early can help you take steps to protect your vision. We also provide novel results for five different screening and clinical grading systems for diabetic retinopathy including state-of-the-art results for accurately classifying images according to clinical five-grade diabetic retinopathy. These results suggest, that a deep learning system could increase the cost-effectiveness of screening and diagnosis, while attaining higher than recommended performance, and that the system could be applied in clinical examinations requiring finer grading.

The purpose of our study is to investigate the effectiveness of UWF fundus image in DR detection.

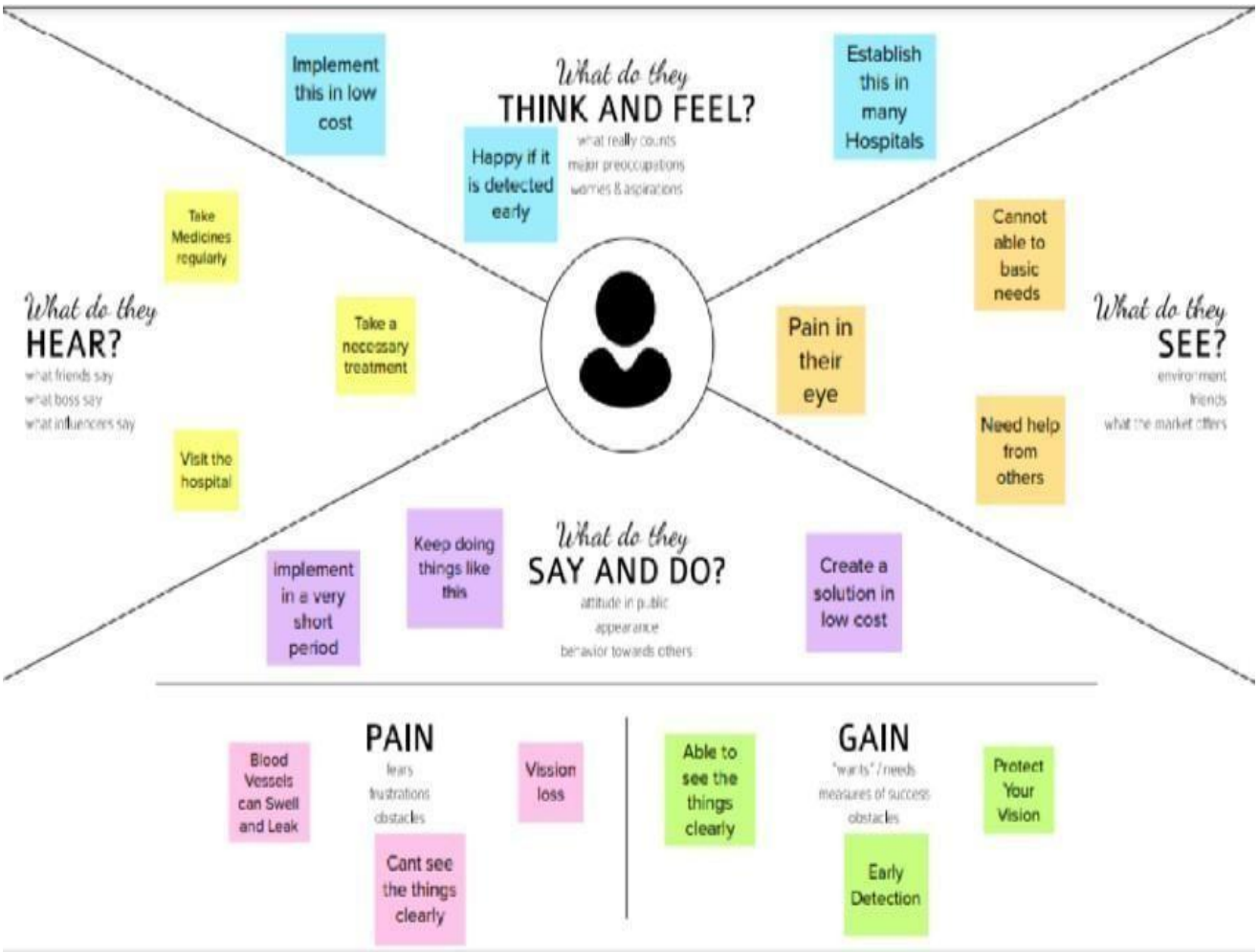
LITERATURE SURVEY:

People with diabetes can have an eye disease called diabetic retinopathy. This is when high blood sugar levels cause damage to blood vessels in the retina. These blood vessels can swell and leak. Or they can close, stopping blood from passing through. Sometimes abnormal new blood vessels grow on the retina. All of these changes can steal your vision.

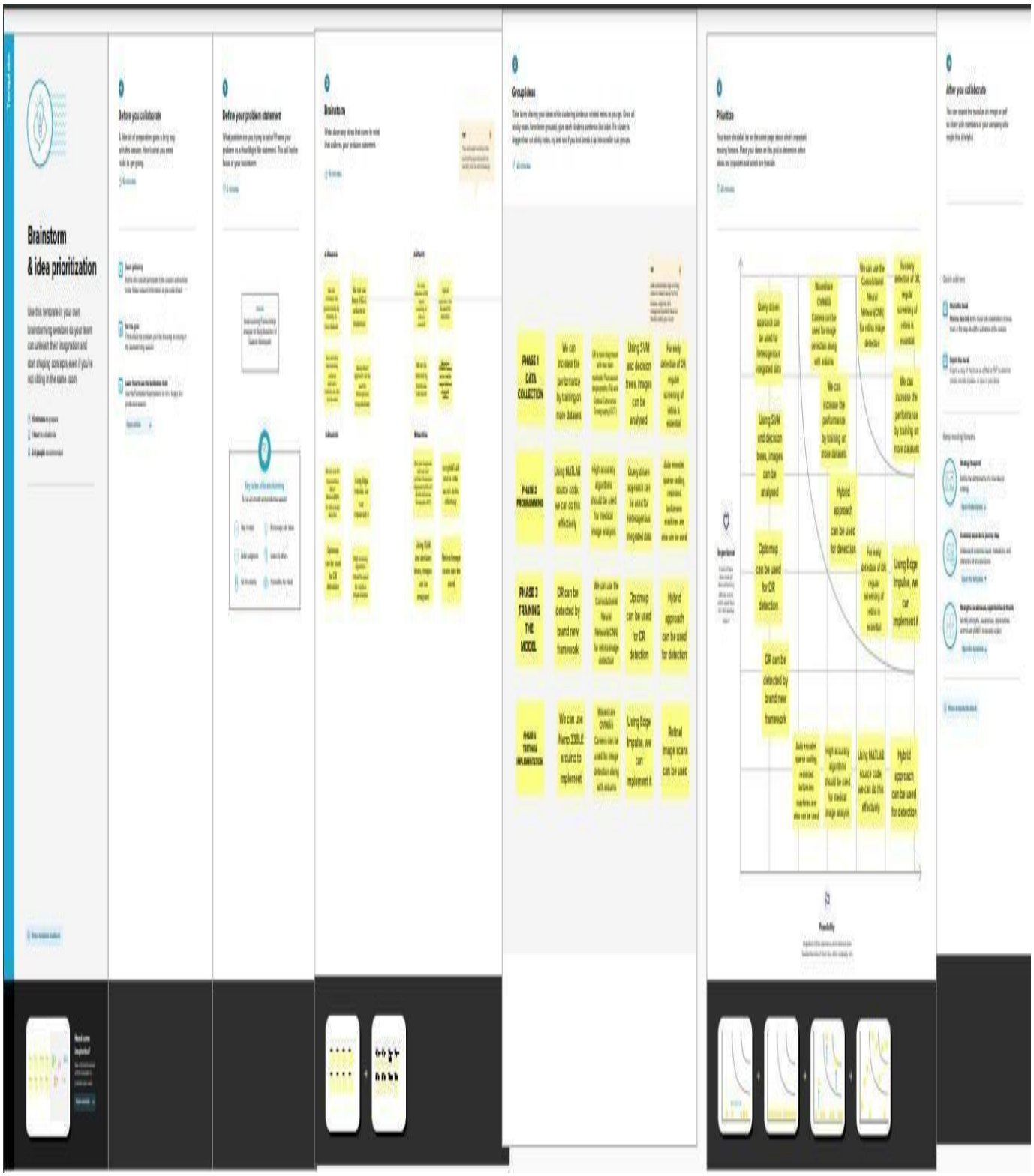
The evaluation of the severity and degree of retinopathy associated with a person having diabetes, is currently performed by medical experts based on the fundus or retinal image of the patient's eyes.

IDEATION & PROPOSED SOLUTION:

EMPATHY MAP CANVAS:



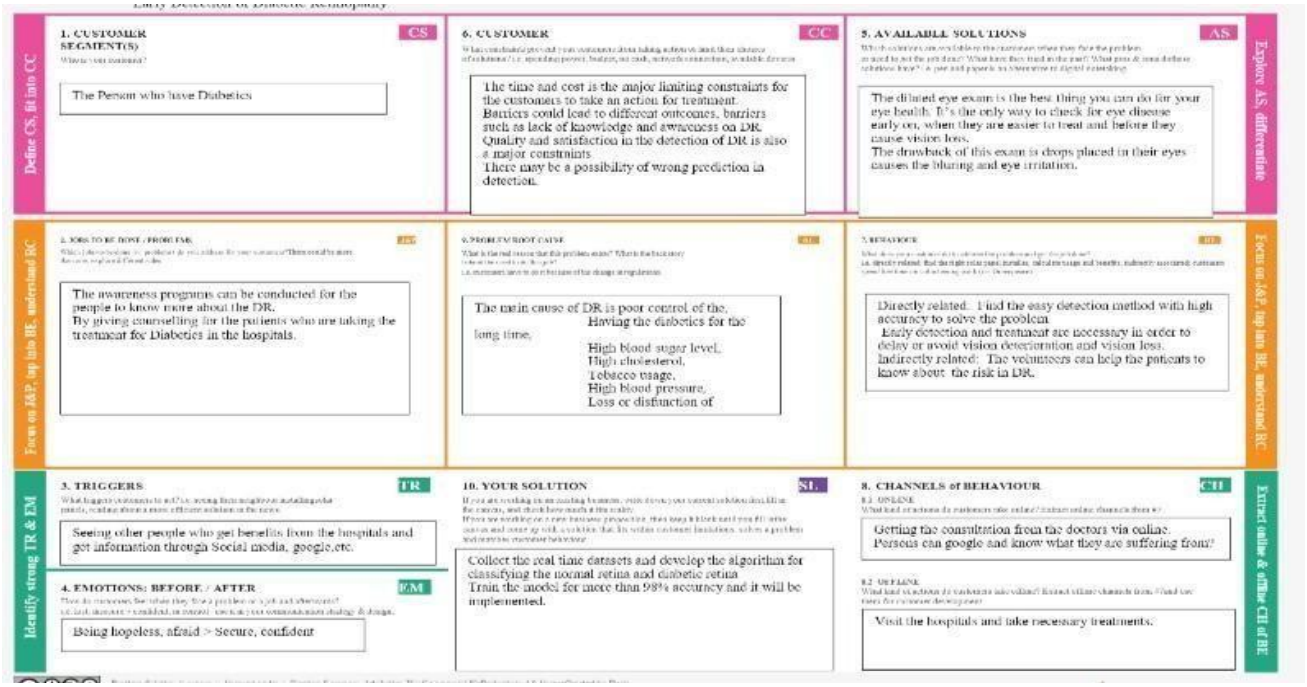
IDEATION & BRAINSTROMING:



PROPOSED SOLUTION:

S.No.	Parameter	Description
1.	Problem Statement	**Deep Learning Fundus Image Analysis for Early Detection of Diabetic Retinopathy:** The Diabetic Retinopathy is a disease which affects the vision of the patient. The project is the time consuming, cost effective detection the DR with the high accuracy without need of Clinicians.
2.	Idea / Solution description	Collect the datasets for classifying the normal retina and diabetic retina in real time. Develop the machine learning algorithm to classify normal retina and diabetic retina. After developing the algorithm, train the model with datasets collected in real time datasets. Validation for test data will be carried. Once the performance accuracy is above 98%, the algorithm will be implemented.
3.	Novelty / Uniqueness	Using the trained model with more than 98% accuracy to detect the diabetic retinopathy will result in the more accurate result.
4.	Social Impact / Customer Satisfaction	This will very helpful for the people to easily detect diabetic retinopathy within the less amount of time and take necessary treatment to prevent the Caustious effects
5.	Business Model (Revenue Model)	This will be developed as a product to detect diabetic retinopathy. This will reduce the number of Clinicians and time required to detect the DR in the hospitals.
6.	Scalability of the Solution	This will be done by collecting the correct information as a constraint and training the model with more datasets till the accuracy becomes greater than the 98%.Once the optimum accuracy is reached, then it will be implemented using embedded device

PROBLEM SOLUTION FIT:



REQUIREMENT ANALYSIS:

FUNCTIONAL REQUIREMENT:

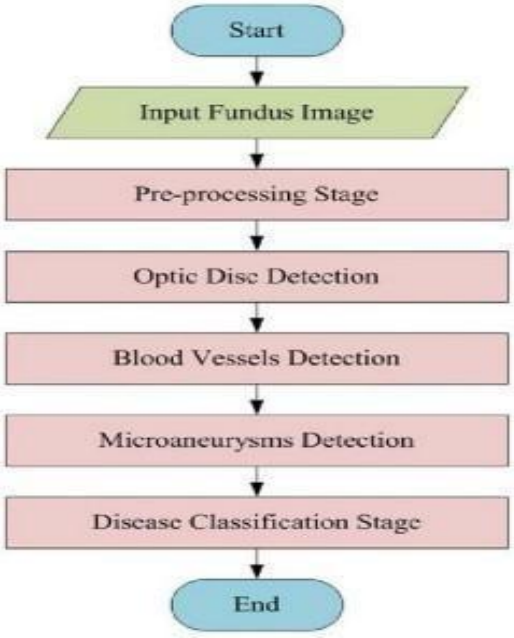
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	Datasets	For training the model, the most accurate real time datasets are required
FR-2	Camera	For getting real time images for testing the model
FR-3	Cloud Storage	For storing the required images and programming

NON FUNCTIONAL REQUIREMENT:

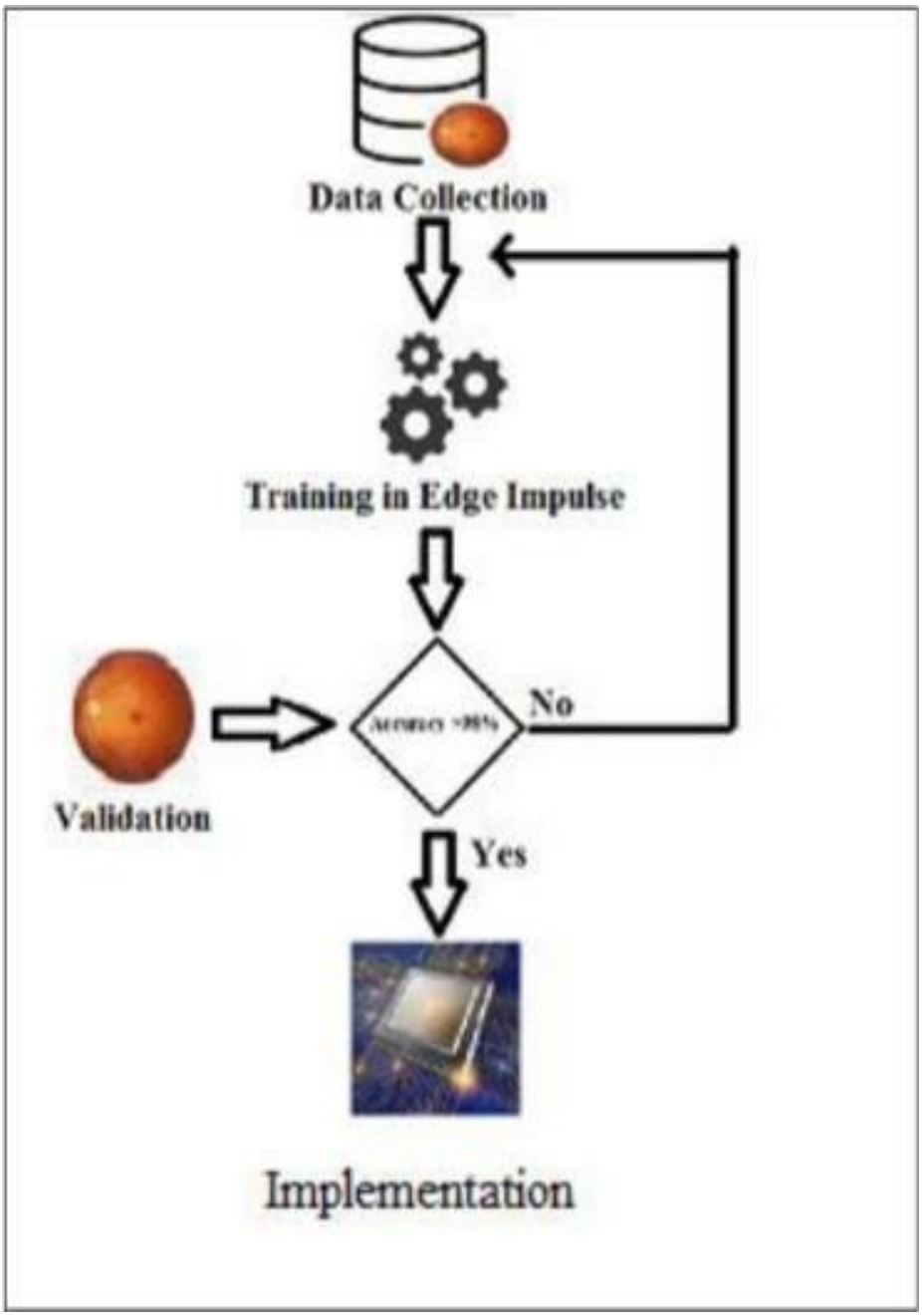
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The user can easily upload their images for processing
NFR-2	**Security**	This will protect the user data like their eye images and their results
NFR-3	**Reliability**	It will process the images more quickly, so that we can process the more number of images within the limited time
NFR-4	**Performance**	This will give more than 98% accuracy
NFR-5	**Availability**	This will be avalaible in low cost , so that we can implement in many places
NFR-6	**Scalability**	It will be enhances for other diseases also

PROJECT DESIGN:

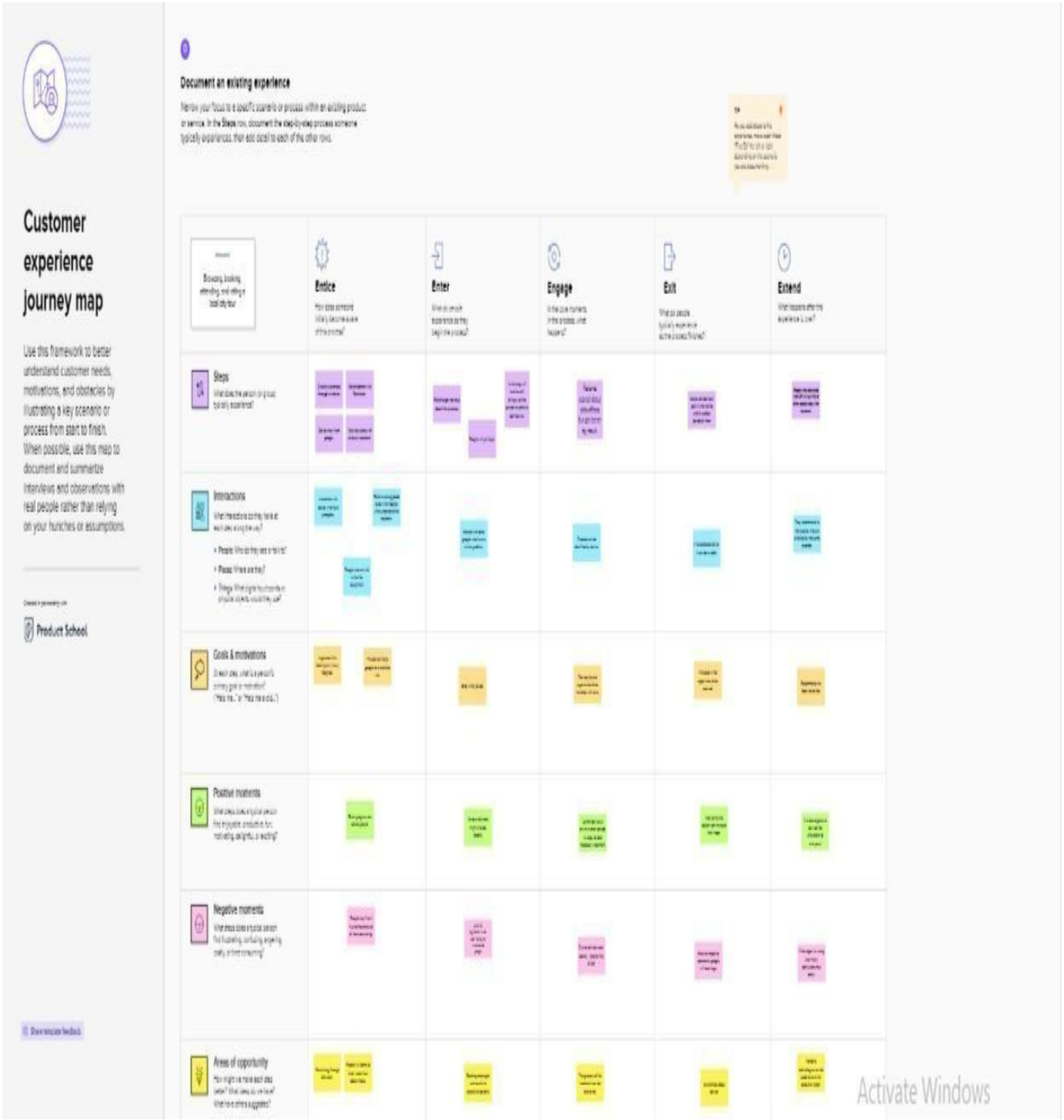
DATA FLOW DIAGRAM:



SOLUTION ARCHITECTURE:



USER STORIES:



PROJECT PLANNING AND SCHEDULING :

SPRINT PLANNING AND ESTIMATION:

S.NO	ACTIVITY TITLE	DESCRIPTION	DURATION
1	Understanding the project and its requirement	Assign the teammembers and createrepository in the GitHub, Assign the task to each team member and teach how to use the GitHub and IBM careereducation .	1 week
2	Start the project	Advice students toattend classes of IBM portal create and develop an rough diagram based on project description and gather informationon AI and IBM project and team leader assign task to each member ofthe project .	1 week

3	Attend class	Team members and team lead mustattend the classes and learnfrom classes provided byIBM and NALAYATHIRAN and must gainaccess of MIT license forthe project.	4 weeks

4	Budget and scope of project	Reduce cost efficiencyand analyse the use ofAI in the project	Progress

SPRINT DELIVERY SCHEDULE:

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority
Sprint-1	Data Collection	Task-1	In Deep Learning Model, It can be spilt into Testing and Training set.	4	Medium
Sprint-1	Data Pre-processing	Task-2	Import the required data for pre-processing. Application of the image data generator to the train and test set.	7	Low
Sprint-1	Build Homepage	USN-1	Homepage give the brief description to the user.	4	Medium
Sprint-2	Create Registration page	USN-2	In this page, User will able to register for the application.	2	Low
Sprint-2	Train,Save,Test	Task-3	To train the model with the configured neural network and save the model. Test the build model against the testing dataset.	3	High
Sprint-3	Create Service Instance	Task-4	Configure the location of resource such as web server and cloud storage for an application.	7	High
Sprint-3	Creating Database	Task-5	IBM Cloud, offered the required credentials to access the services and the database accessed by the users.	6	High
Sprint-3	Creating Tables in Database	Task-6	Structure the required tables with necessary attributes in cloud DB.	4	Medium

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority
Sprint-4	Building login page	USN-3	User will be able to login by using the credentials.	3	Low
Sprint-4	Create Image uploading page	Task-7	Upload the test image	2	Low
Sprint-4	Building Prediction page	USN-4	User able to receive the diagnosis on their diabetic retinopathy.	2	Medium
Sprint-4	Building logout page	USN-5	User will be able to logout their account in this Page.	2	Medium
Sprint-4	Build Python code	Task-8	The Necessary modules should be initialize and the libraries should be imported.	1	Medium
Sprint-4		Task-9	Use the database using initiating client and rendering HTML page.	2	Medium
Sprint-4		Task-10	Configuring the registration, login pages and evaluating the credentials.	2	Medium
Sprint-4		Task-11	The model prediction will be showcased on UI.	1	High
Sprint-4	Run the Application	Task-12	Run to check the application.	2	High
Sprint-4		Task-13	Upload image in the homepage to predict the diabetic retinopathy.	5	High
Sprint-4	Train model on IBM	Task-14	Train the model on IBM and integrate it with the flask application.	3	High

CODING:

```
import numpy as np
import os
from tensorflow.keras.models import load_model
from  tensorflow.keras.preprocessing import image
from tensorflow.keras.applications.inception_v3 import
preprocess_input
from flask import Flask, request,flash, render_template, redirect,url_for
from cloudant.client import Cloudant
from twilio.rest import Client
```

```
model = load_model("inception-diabetic.h5")
app = Flask(__name__)
app.secret_key="abc"
app.config['UPLOAD_FOLDER'] = "User_Images"
# Authenticate using an IAM API key
client=Cloudant.iam('0f5ab837-7e5c-486c-a220-5256e075616c-
bluemix','0UGpqPnFUGkN6XC93fLwLujtajQ7wWmOVf7HGB2z2gq
X',connect=True)
my_database=client.create_database('my_database')
if my_database.exists():
   print("Database '{0}' successfully created.".format('my_db'))
# default home page or route

user = ""

@app.route('/')
def index():
   return render_template('index.html', pred="Login", vis ="visible")

@ app.route('/index')
def home():
   return render_template("index.html", pred="Login", vis ="visible")



# registration page
@ app.route('/register',methods=["GET","POST"])
def register():
```

```
    if request.method == "POST":
        name = request.form.get("name")
        mail = request.form.get("emailid")
        mobile = request.form.get("num")
        pswd = request.form.get("pass")
        data = {
            'name': name,
            'mail': mail,
            'mobile': mobile,
            'psw': pswd
             }
        print(data)
        query = {'mail': {'$eq': data['mail']}}
        docs = my_database.get_query_result(query)
        print(docs)
        print(len(docs.all()))
        if (len(docs.all()) == 0):
            url = my_database.create_document(data)
            return render_template("register.html", pred=" Registration
Successful , please login using your details ")
        else:
            return render_template('register.html', pred=" You are already a
member , please login using your details ")
    else:
        return render_template('register.html')
```

```
@ app.route('/login', methods=['GET','POST'])
def login():
    if request.method == "GET":
        user = request.args.get('mail')
        passw = request.args.get('pass')
        print(user, passw)
        query = {'mail': {'$eq': user}}
        docs = my_database.get_query_result(query)
        print(docs)
        print(len(docs.all()))
        if (len(docs.all()) == 0):
            return render_template('login.html', pred="")
        else:
            if ((user == docs[0][0]['mail'] and passw == docs[0][0]['psw'])):
                flash("Logged in as " + str(user))
                return render_template('index.html', pred="Logged in as
"+str(user), vis ="hidden", vis2="visible")
            else:
                return render_template('login.html', pred="The password is
wrong.")
    else:
        return render_template('login.html')


@ app.route('/logout')
def logout():
     return render_template('logout.html')
```

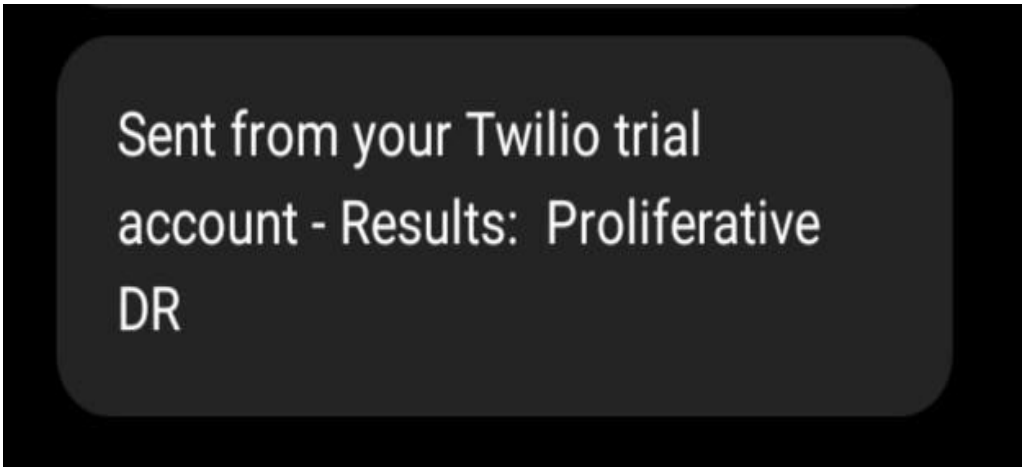
```
@app.route("/predict",methods=["GET",  "POST"])
def predict():
    if request.method == "POST":
        f = request.files['file']
        # getting the current path 1.e where app.py is present
        basepath = os.path.dirname(_file_)
        #print ( " current path " , basepath )
        # from anywhere in the system we can give image but we want that
        filepath = os.path.join(str(basepath), 'User Images', str(f.filename))
        #print ( " upload folder is " , filepath )
        f.save(filepath)
        img = image.load_img(filepath, target_size=(224, 224))
        x = image.img_to_array(img)  # ing to array
        x = np.expand_dims(x, axis=0) # used for adding one more
dimension
        #print ( x )
        img_data = preprocess_input(x)
        prediction = np.argmax(model.predict(img_data), axis=1)
        index = [' No Diabetic Retinopathy ', ' Mild NPDR ',
                ' Moderate NPDR ', ' Severe NPDR ', ' Proliferative DR ']
        result = str(index[prediction[0]])
        print(result)
        account_sid = 'AC040db3f6fc52556d2a15a7c8814238e2'
        auth_token = '1dedefd14bb721ecb05dd4c68dc1ec82'

        client = Client(account_sid, auth_token)
```

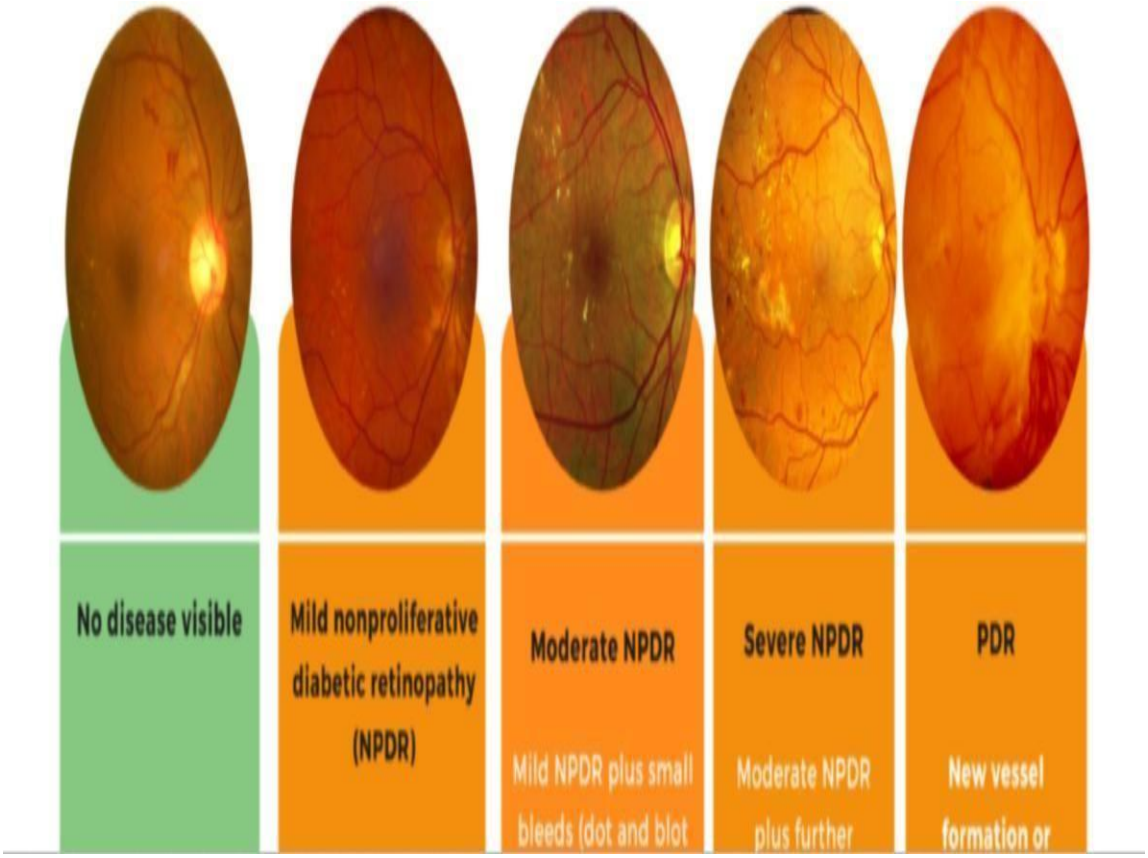
```
        ''' Change the value of 'from' with the number
        received from Twilio and the value of 'to'
        with the number in which you want to send message.'''
        message = client.messages.create(
                                from_='+17262274397',
                                body ='Results: '+ result,
                                to ='+918925176648'
                            )
        return render_template('prediction.html', prediction=result, fname =
filepath)
    else:
        return   render_template("prediction.html")

if__name__== "__main__":
    app.debug = True
    app.run()
```

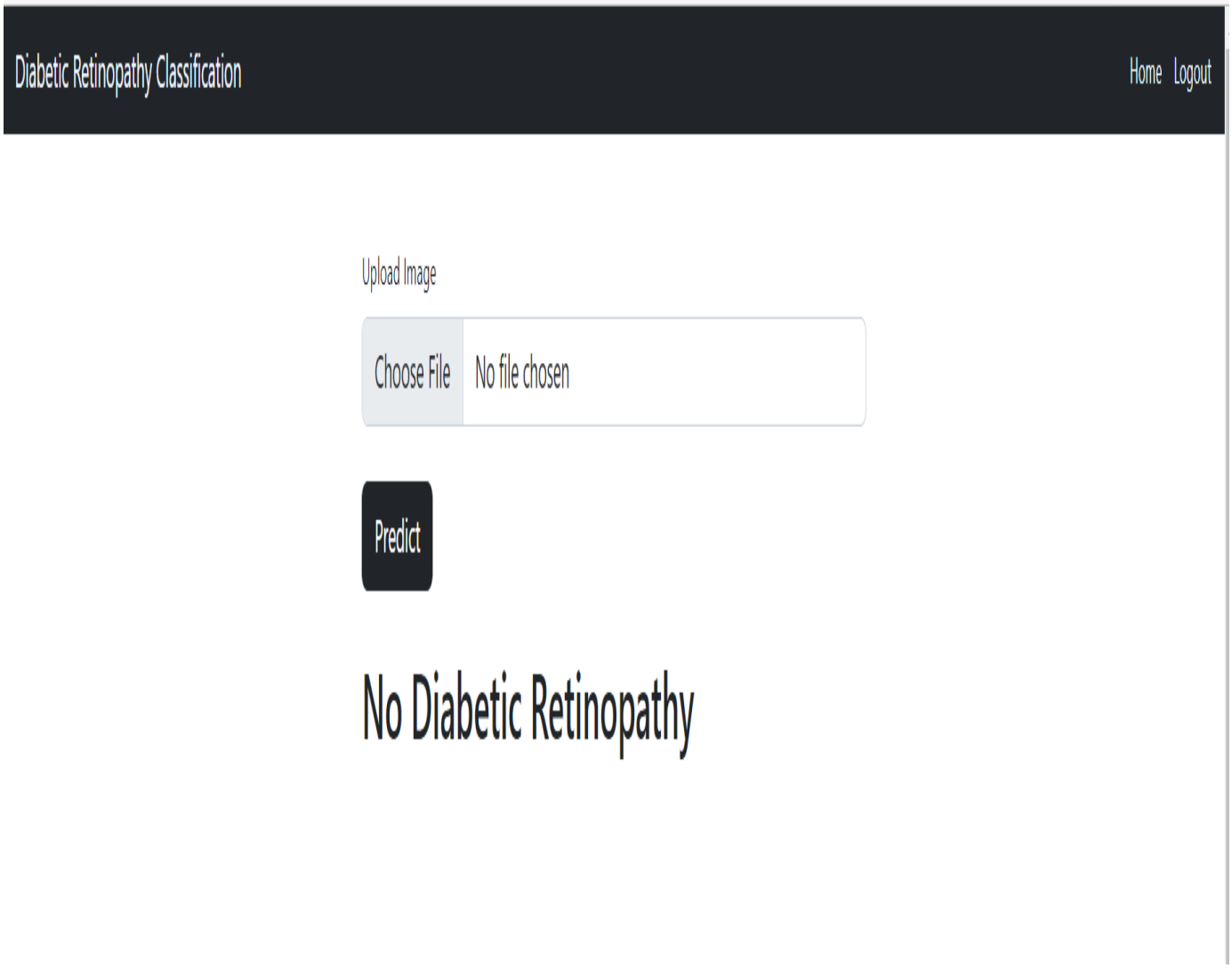
SOLUTION :



TESTING:


No disease visible
Mild nonproliferative diabetic retinopathy (NPDR)
Moderate NPDR
Mild NPDR plus small bleeds (dot and blot
Severe NPDR
Moderate NPDR plus further
PDR
New vessel formation or

RESULTS:


Diabetic Retinopathy Classification
Home Logout
Upload Image
Choose File
No file chosen
Predict
No Diabetic Retinopathy

ADVANTAGES:

Earlier detection reduce the risk of Vision loss.

The amount of time for detecting the DR is less.

Cost of detecting is less.

DISADVANTAGES:

If the images is not uploaded correctly then detection may be difficult.