

Now when you click on predict button from top right corner you will get redirected to predict.html

Let's look how our predict.html file looks like:

Thyroid Disease Classification

goitre

Male ▾

tumor

Male ▾

hypopituitary

Male ▾

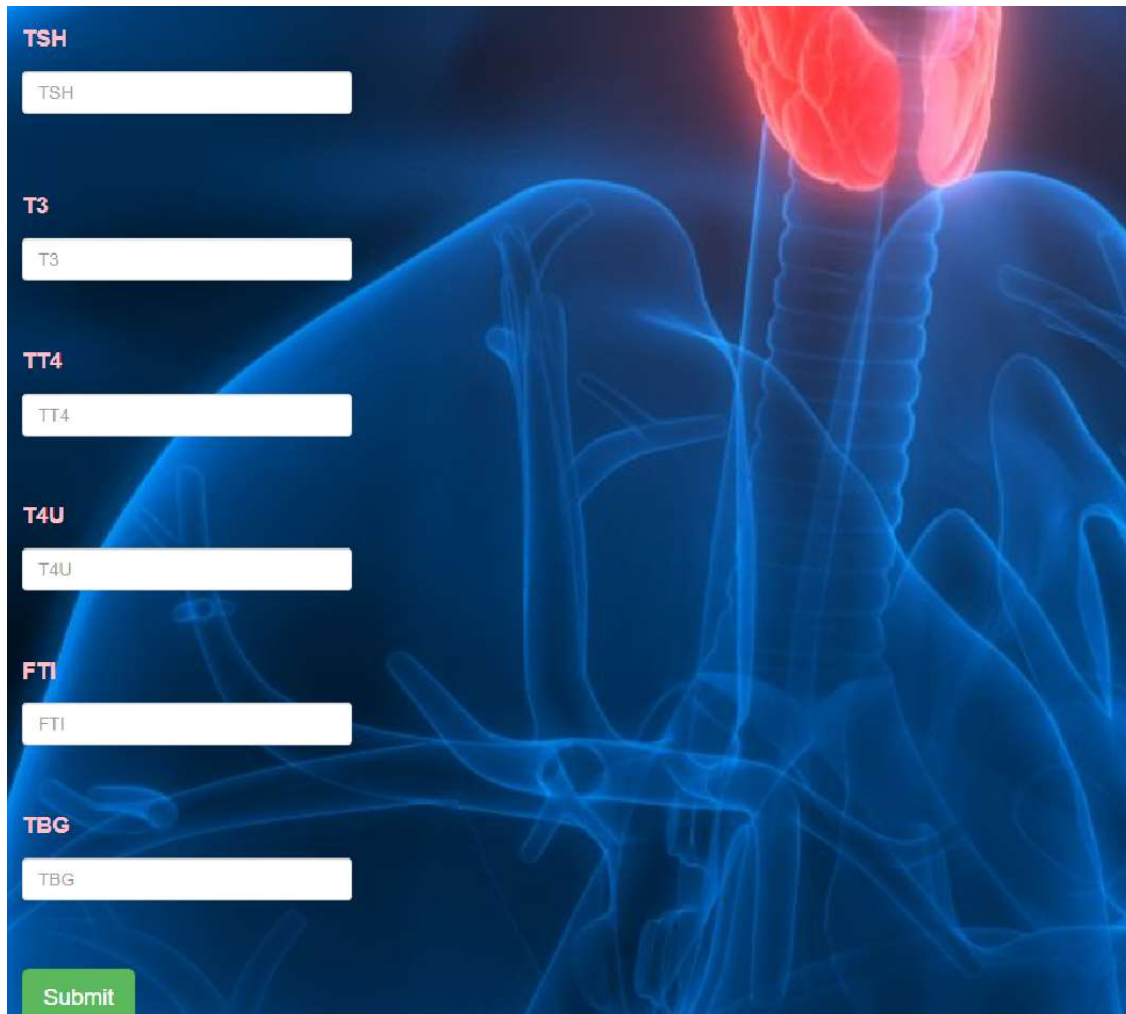
psych

Male ▾

TSH

TSH





TSH

T3

TT4

T4U

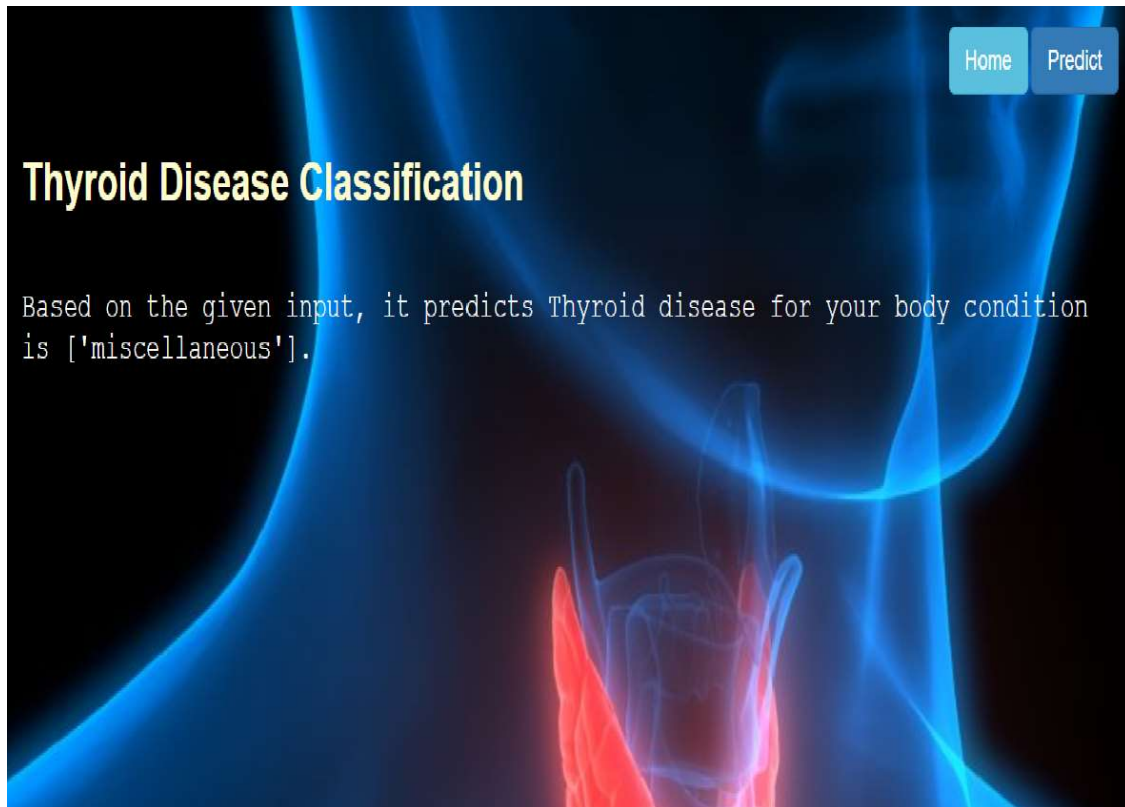
FTI

TBG

Submit

Now when you click on submit button from left bottom corner you will get redirected to submit.html

Let's look how our submit.html file looks like: it is ['miscellaneous'].



Activity 2.2: Build Python code:

Import the libraries

```
from flask import Flask, render_template, request
import numpy as np
import pickle
import pandas as pd
```

Load the saved model. Importing the flask module in the project is mandatory. An object of Flask class is our WSGI application. Flask constructor takes the name of the current module (`__name__`) as argument.

```
model = pickle.load(open(r"C:\Users\SmartBridge-PC\Downloads\Thyroid\thyroid1_model.pkl", 'rb'))
le = pickle.load(open("label_encoder.pkl", 'rb'))

app = Flask(__name__)
```

Render HTML page:

```
@app.route("/")
def about():
    return render_template('home.html')
```

Here we will be using a declared constructor to route to the HTML page which we have created earlier.

In the above example, '/' URL is bound with the home.html function. Hence, when the home page of the web server is opened in the browser, the html page will be rendered. Whenever you enter the values from the html page the values can be retrieved using POST Method.

Retrieves the value from UI:

```
@app.route("/pred", methods=['POST', 'GET'])
def predict():
    x = [[float(x) for x in request.form.values()]]

    print(x)
    col = ['goitre', 'tumor', 'hypopituitary', 'psych', 'TSH', 'T3', 'TT4', 'T4U', 'FTI', 'TBG']
    x = pd.DataFrame(x, columns=col)

    #print(x.shape)

    print(x)
    pred = model.predict(x)
    pred = le.inverse_transform(pred)
    print(pred[0])
    return render_template('submit.html', prediction_text=str(pred))
```

Here we are routing our app to predict() function. This function retrieves all the values from the HTML page using Post request. That is stored in an array. This array is passed to the model.predict() function. This function returns the

prediction. And this prediction value will be rendered to the text that we have mentioned in the submit.html page earlier.

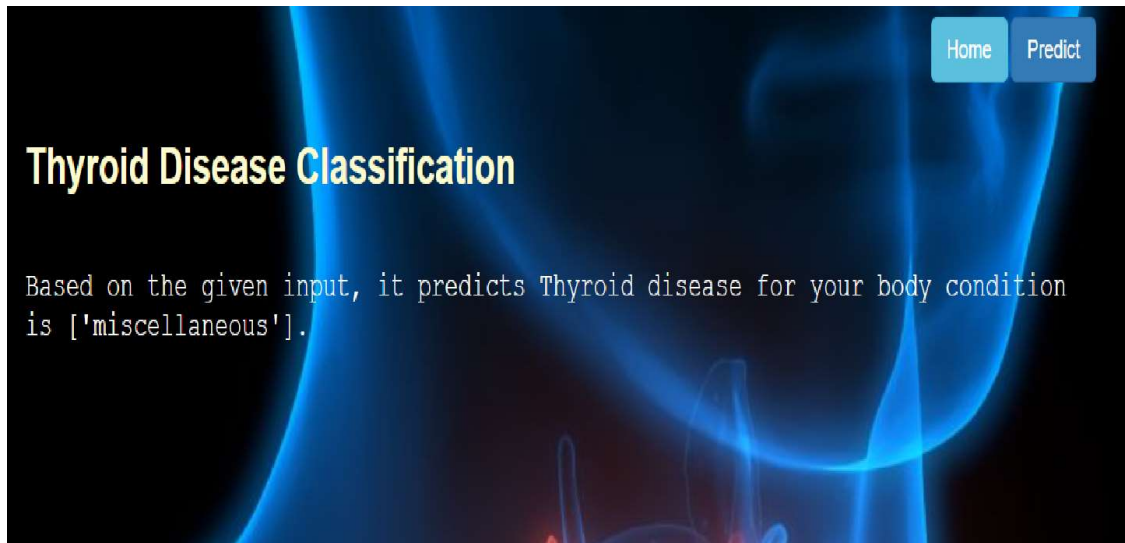
Main Function:

```
if __name__ == "__main__":  
    app.run(debug=False)
```

Activity 2.3: Run the application

- Open anaconda prompt from the start menu
- Navigate to the folder where your python script is.
- Now type “python app.py” command
- Navigate to the localhost where you can view your web page.
- Click on the predict button from the top right corner, enter the inputs, click on the submit button, and see the result/prediction on the web.

```
In [32]: runfile('C:/Users/SmartBridge-PC/  
Downloads/Thyroid/app.py', wdir='C:/Users/  
SmartBridge-PC/Downloads/Thyroid')  
* Serving Flask app "app" (lazy loading)  
* Environment: production  
  WARNING: This is a development server. Do  
not use it in a production deployment.  
  Use a production WSGI server instead.  
* Debug mode: off  
* Running on http://127.0.0.1:5000/ (Press  
CTRL+C to quit)
```

Milestone 7: Project Demonstration & Documentation

Below mentioned deliverables to be submitted along with other deliverables

Activity 1:- Record explanation Video for the project end to end solution

Activity 2:- Project Documentation-Step by step project development procedure

Create document as per the template provided