Application Building

Building an application to integrate the model

After the model is built, we will be integrating it to a web application so that normal users can also use it to know if any website is phishing or safe in a no-code manner.

In the application, the user provides any website URL to check and the corresponding parameter values are generated by analysing the URL using which legitimate websites are detected.

Let's get started!

Flask App (Step - 1)

Build the python flask app

In the flask application, the URL is taken from the HTML page and it is scraped to get the different factors or the behaviour of the URL. These factors are then given to the model to know if the URL is phishing or safe and is sent back to the HTML page to notify the user.

Input the following commands to Import required libraries

```
1  import numpy as np
2  from flask import Flask, request, jsonify, render_template
3  import pickle
4  #importing the inputScript file used to analyze the URL
5  import inputScript
```

Load the model and initialize Flask App

```
8  #load model
9  app = Flask(__name__)
10 model = pickle.load(open('Phishing_Website.pkl', 'rb'))
11
```

Flask App (Step - 2)

Configure app.py to fetch the URL from the UI, process the URL, get the input parameters from the URL and return the prediction.

Input the following commands:

```
#Redirects to the page to give the user iput URL.
@app.route('/predict')
def predict():
    return render_template('final.html')

#Fetches the URL given by the URL and passes to inputScript
@app.route('/y_predict',methods=['POST'])
def y_predict():
    '''
    For rendering results on HTML GUI
    '''
    url = request.form['URL']
    checkprediction = inputScript.main(url)
    prediction = model.predict(checkprediction)
    print(prediction)
    output=prediction[0]
    if(output==1):
        pred="Your are safe!!  This is a Legitimate Website."

    else:
        pred="You are on the wrong site. Be cautious!"
    return render_template('final.html', prediction_text='{}'.format(pred),url=url)

#Takes the input parameters fetched from the URL by inputScript and returns the predictions
@app.route('/predict_api',methods=['POST'])
def predict_api():
    '''
    For direct API calls trought request
    '''
    data = request.get_json(force=True)
    prediction = model.y_predict([np.array(list(data.values()))])

    output = prediction[0]
    return jsonify(output)

```

Run the app

Enter commands as shown below

```

if __name__ == '__main__':
    app.run(host='0.0.0.0', debug=True)

```

Build An HTML Page

We Build an HTML page to take the URL as a text and upon clicking on the button for submission it has to redirect to the URL for “y predicts” which returns if the URL given is phishing or safe. The output is to be then displayed on the page. The HTML pages are put under the templates folder and any style sheets if present is kept in the static folder.

Execute And Test Your Model

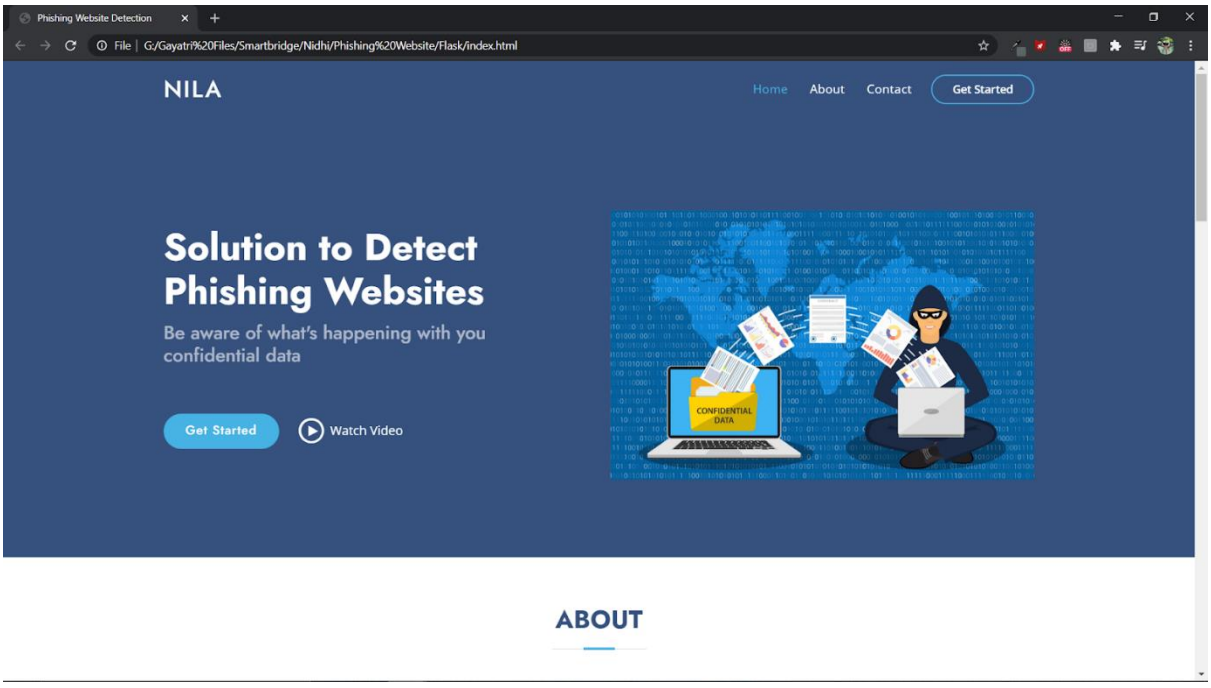
Now we execute the model using Anaconda Prompt

Execute the python code by giving the command python app.py in anaconda prompt as shown below

```
(base) G:\Gayatri Files\Smartbridge\Nidhi\Phishing Website\Flask>python app.py
 * 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: on
 * Restarting with windowsapi reloader
 * Debugger is active!
 * Debugger PIN: 715-830-168
 * Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
```

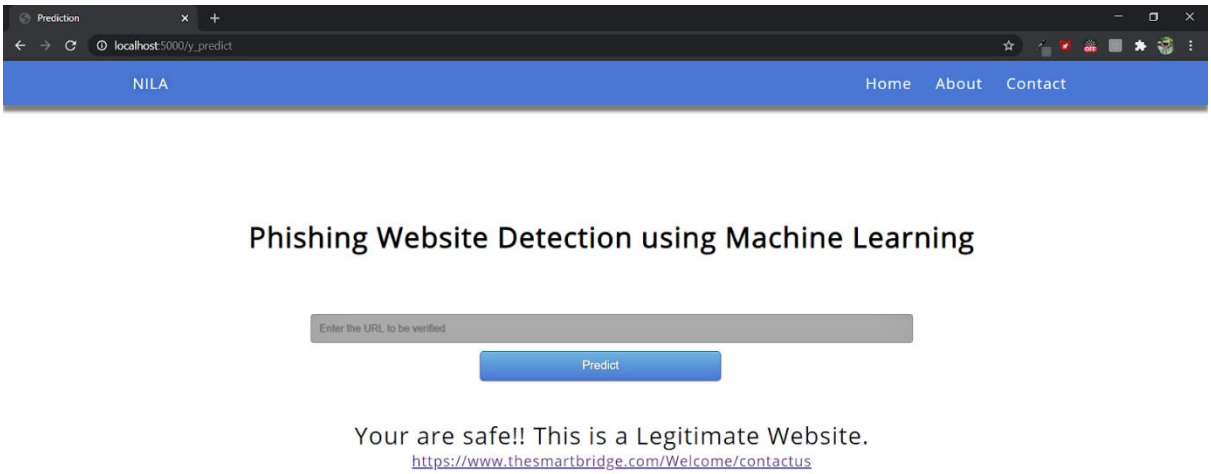
Now, while the app is running, open index.html present in project folder. Scroll the webpage down to find the buttons to test the application.

This is the home page of the web application(index.html)

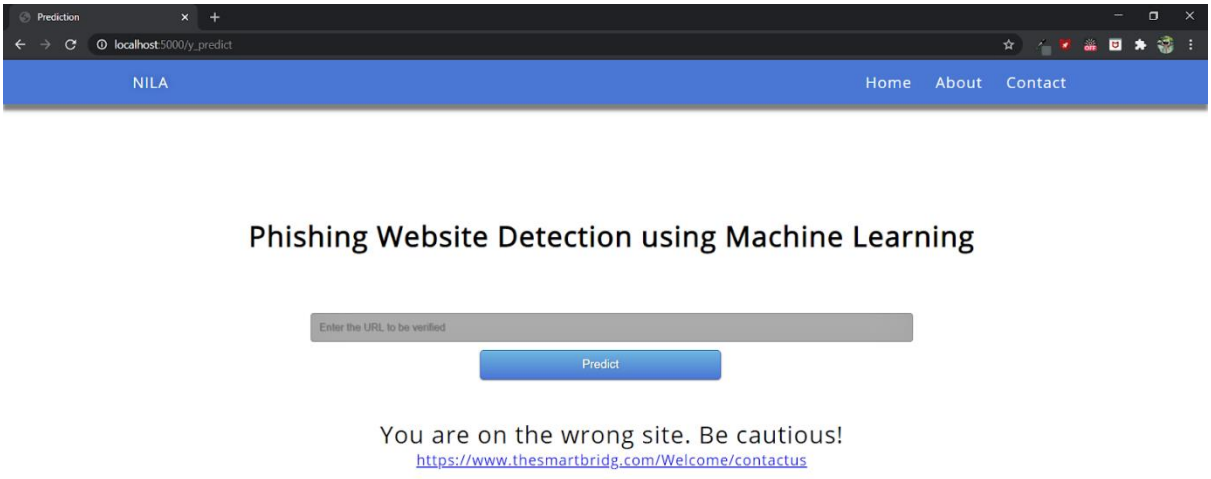


The Final Step

When the URL is given, the model analyses and gives the output whether it is a phishing or legitimate website.



Here, we will try to specify the same link given above by altering the spelling of the domain name. It validates with the domain name and if not found, It warns about the risk of phishing.



Hurray!!! You have made it.

Congratulations on successfully completing the "Web Phishing Detection using Machine Learning"