

Web Phishing Detection

Testing Model

Date	16 November 2022
Team ID	PNT2022TMID22501
Project Name	Project – Web Phishing Detection
Maximum Marks	-

Executing the model:

The screenshot displays the Visual Studio Code interface with a Flask application named 'phishing_detection.py' open. The Explorer panel on the left shows the project structure, including a 'Flask_app' directory with files like 'app.py', 'inputScript.py', and 'phishing_detect.py'. The main editor shows the code for 'phishing_detection.py', which includes imports for Flask, pickle, and request, and defines routes for '/', '/about', '/predict', and '/y_predict'. The Terminal panel at the bottom shows the output of running the application, indicating it is serving on http://127.0.0.1:5000/ and reloading due to a change in the environment.

```
File Edit Selection View Go Run Terminal Help phishing_detection.py - Web Phishing Detection - Visual Studio Code

EXPLORER
WEB PHISHING DETECTION
  Flask_app
    __pycache__
    static
    phishing1.jfif
    phishing1.jfif
    phishing2.jfif
    phishing3.jfif
    Templates
    app.py
    inputScript.py
    phishing_detect.py
    phishing_detection.py

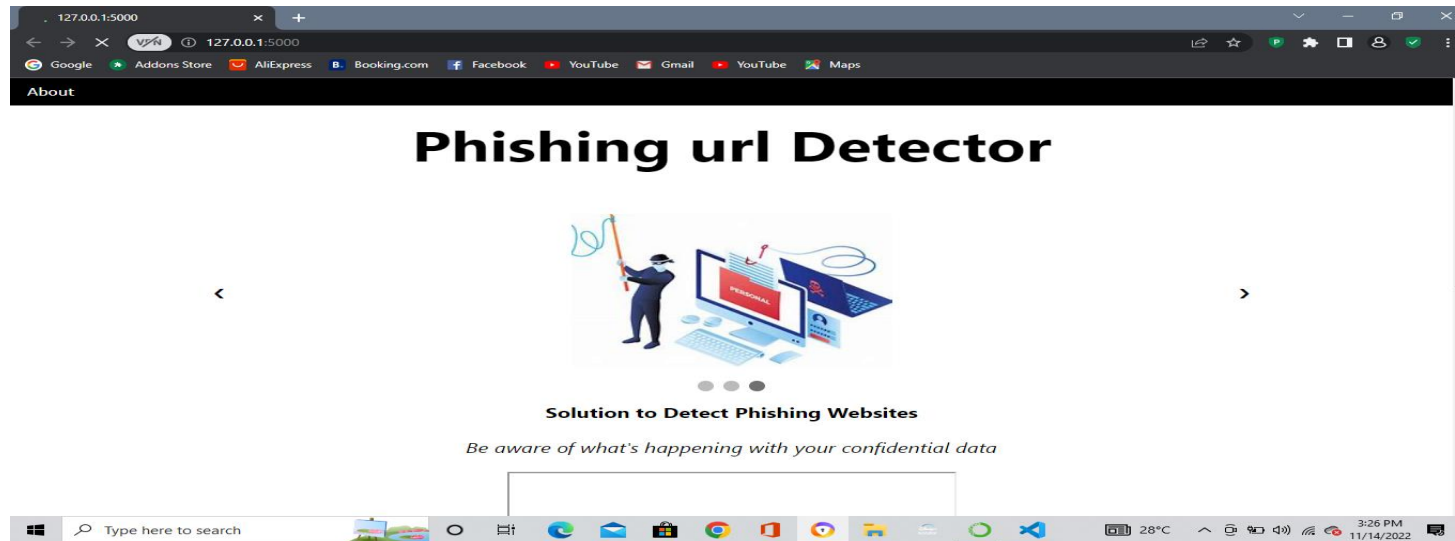
phishing_detection.py x
Flask_app > phishing_detection.py > about

10 app = Flask(__name__)
11 model = pickle.load(open(r'C:\SavedModel\Phishing_Website.pkl', 'rb'))
12
13
14 @app.route('/') #decorator
15 def phishing_detection():
16     return render_template('index.html')
17
18 @app.route('/about')
19 def about():
20     return render_template('about.html')
21
22 @app.route('/predict')
23 def predict():
24     return render_template('predict.html')
25
26 @app.route('/y_predict', methods=['POST'])
27 def y_predict():
28     url = request.form['url']

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL JUPYTER
/OneDrive/Documents/Web Phishing Detection/Flask_app/phishing_detection.py
* Serving Flask app "phishing_detection" (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 watchdog (windowsapi)
* Debugger is active!
* Debugger PIN: 416-564-543
* Running on http://127.0.0.1:5000/ (Press CTRL+C to quit)
* Detected change in 'C:\Users\nandhitha\anaconda3\lib\concurrent\__init__.py', reloading

Ln 20, Col 41 Spaces: 4 UTF-8 CRLF Python 3.9.12 (base: conda) 7:32 PM 11/14/2022
```

Home page of the web application:



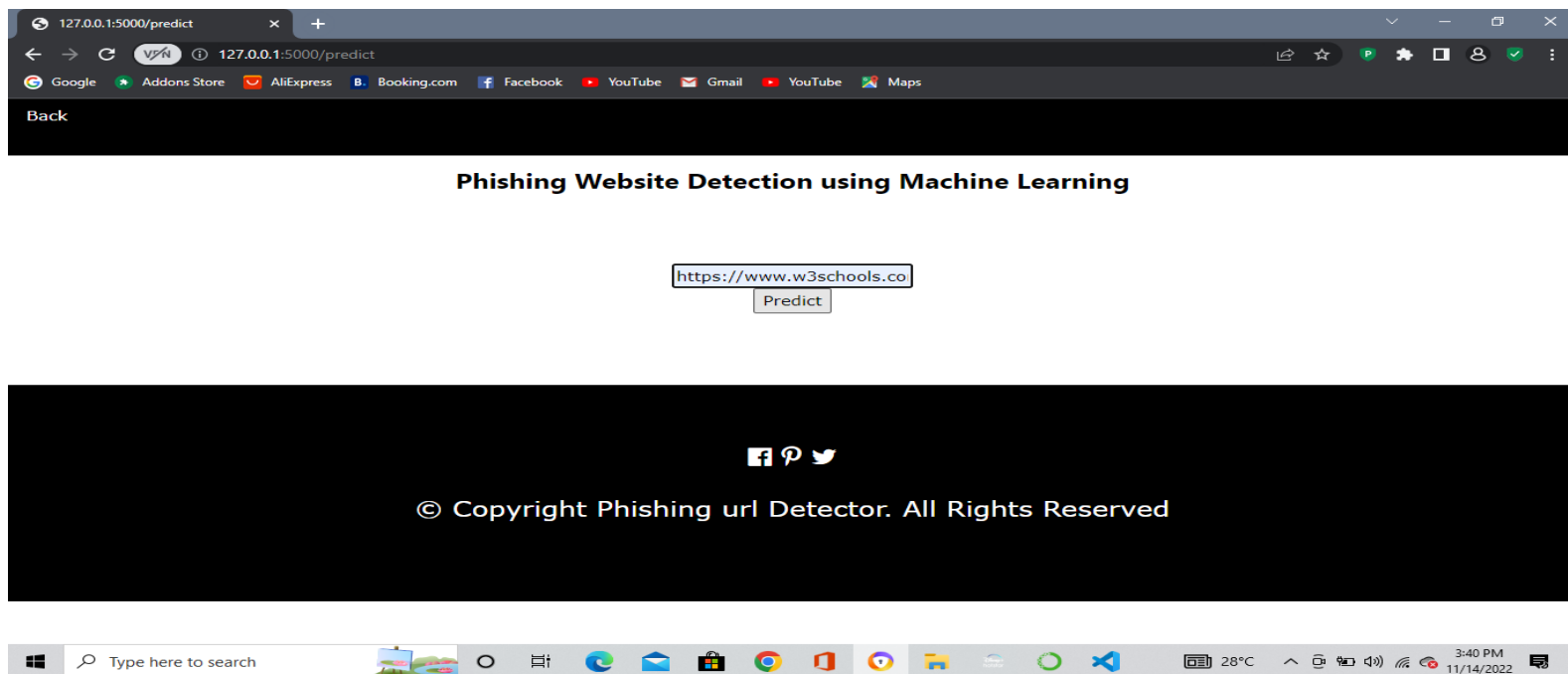
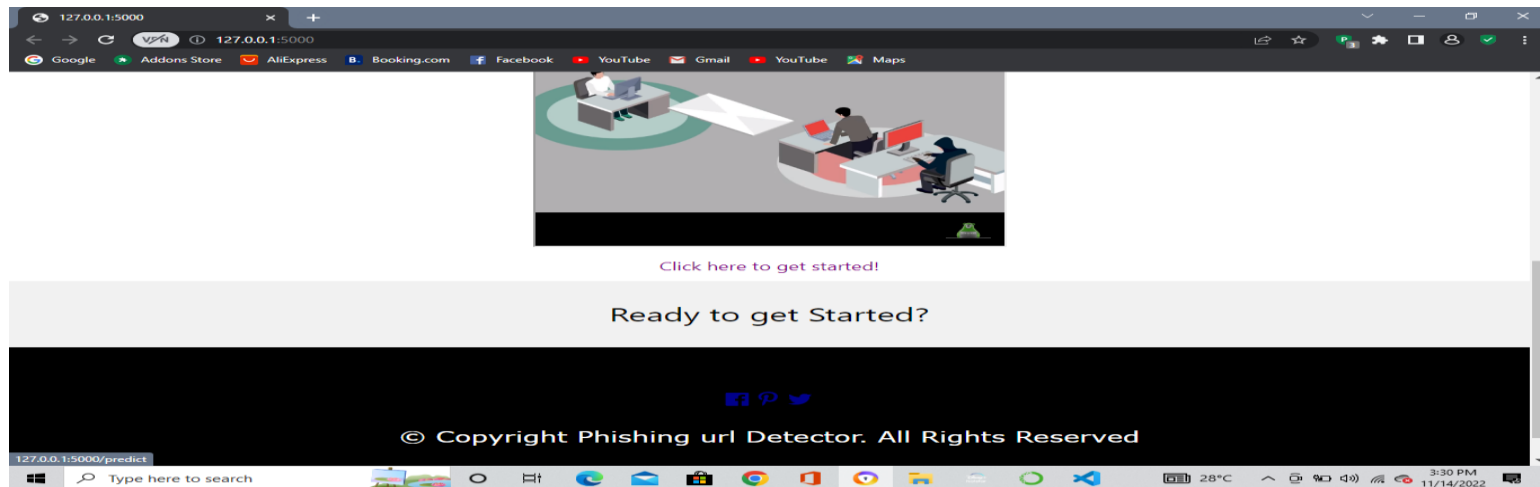
About page:

The user upon clicking the about button available in the navigation bar, the user will be redirected to the About page.



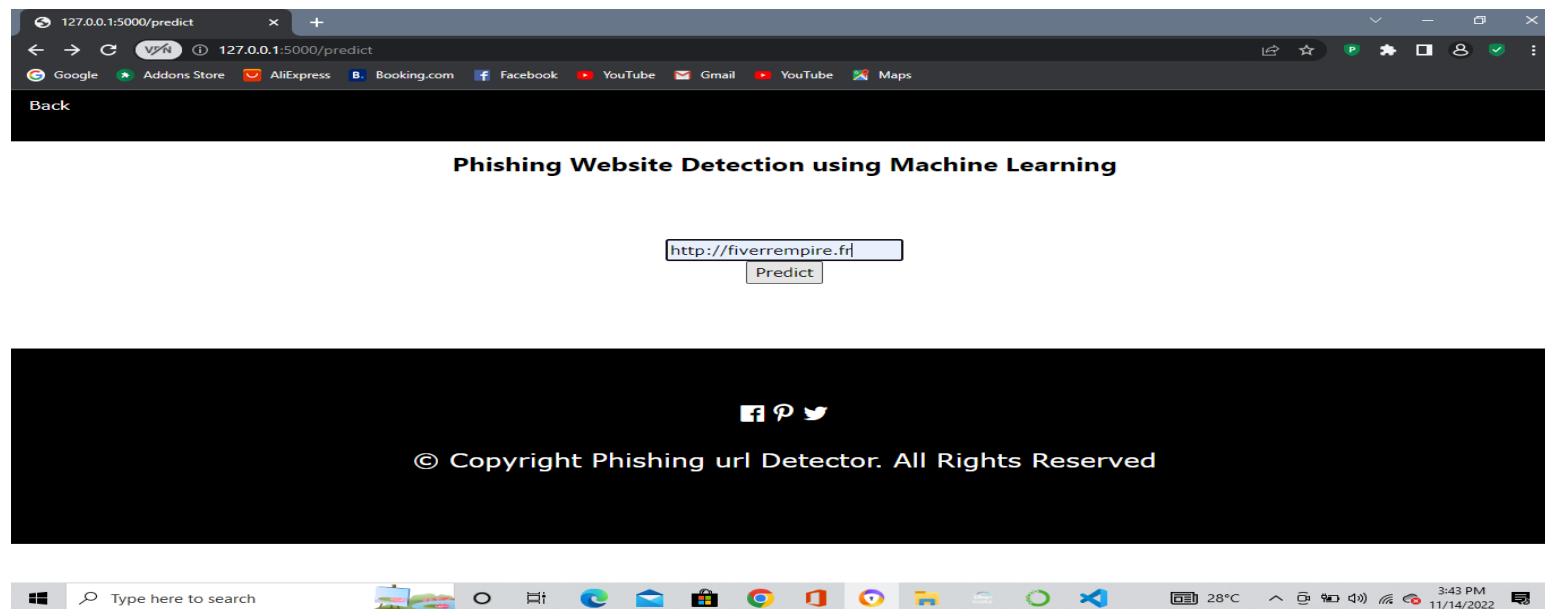
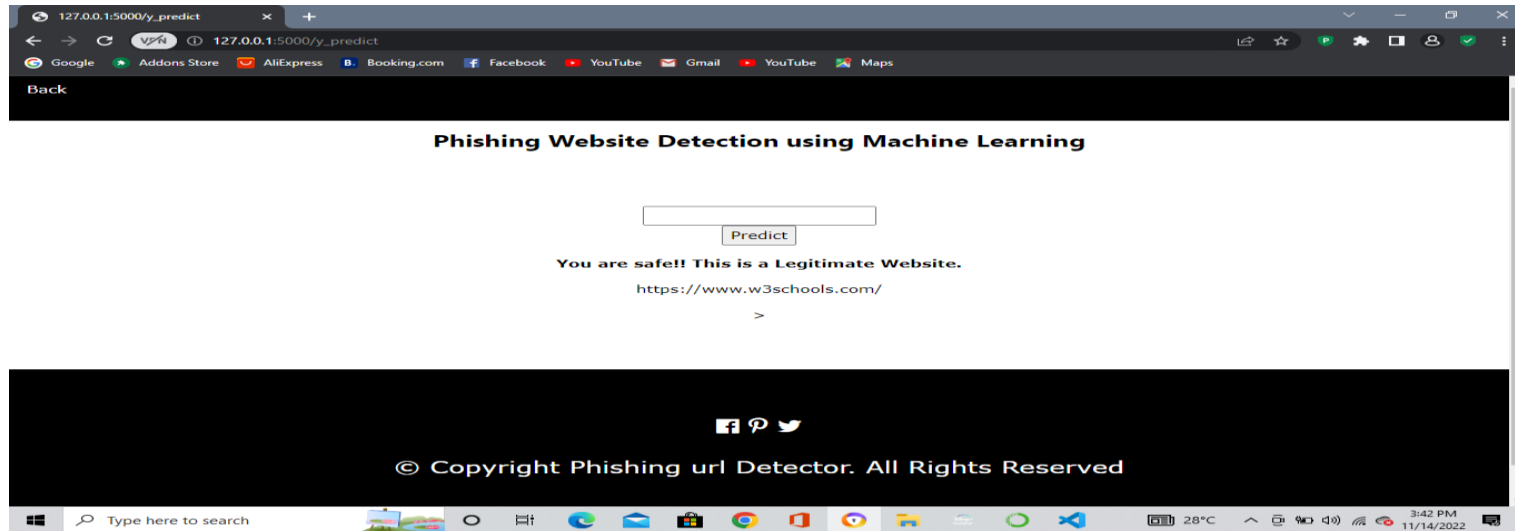
Prediction page:

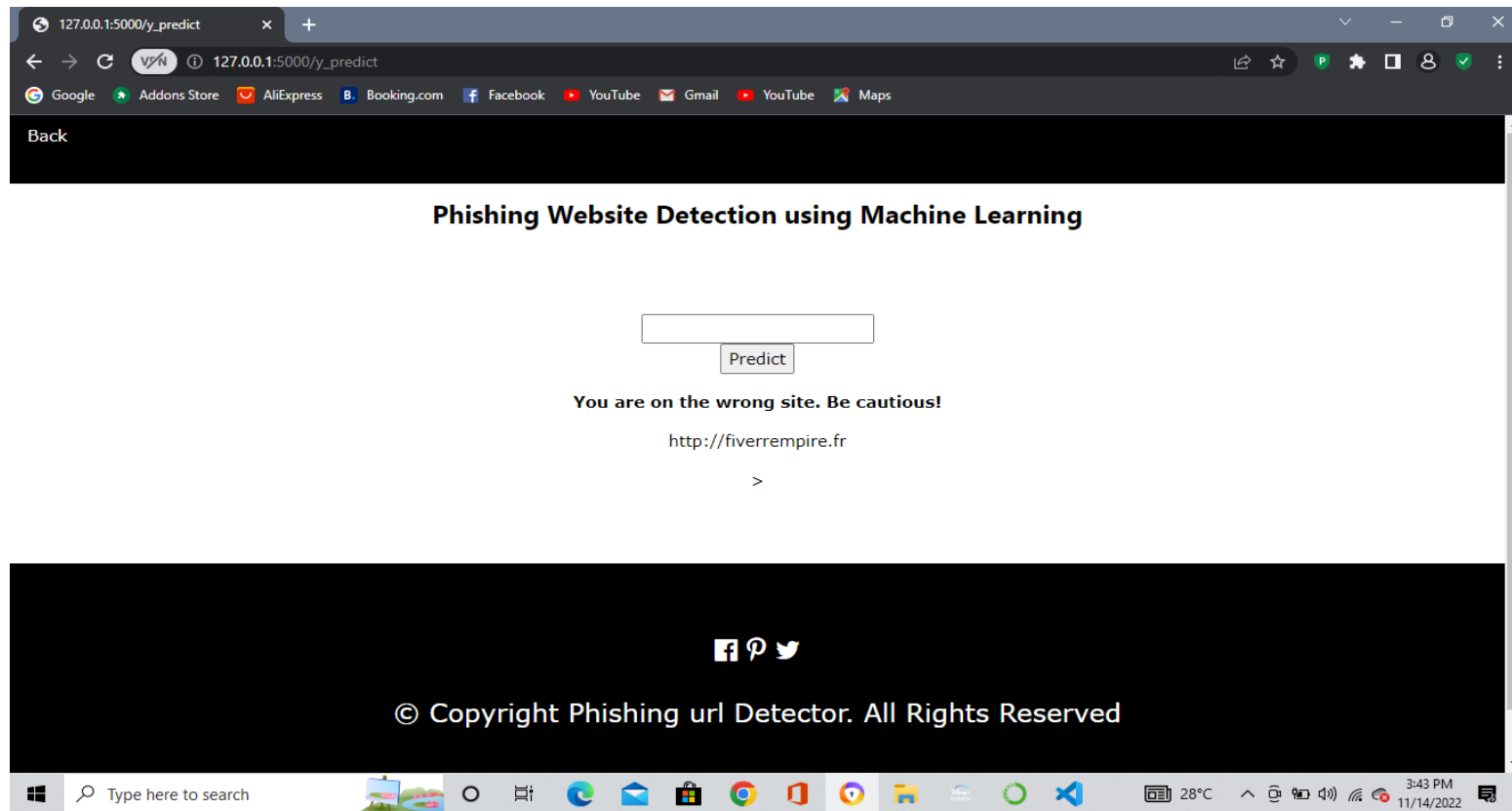
- Now when the user clicks on the “Click here to get started” link, the user will now be redirected to the prediction page.
- In the prediction page the user can enter the url in the search bar, and when he clicks on the “Predict” button the user will be redirected to the y_prediction page.



Y_prediction page:

Now in this page the output is displayed. If the url is legitimate then the message is displayed stating that “You are safe!! This is a Legitimate Website.” else if the url is a phishing url then the message is displayed as “You are on the wrong site, Be cautious!”.





Conclusion:

We have successfully built the model for predicting the phishing urls and have successfully built the web application using flask framework and the testing is done and the website works successfully as expected. We have used the Random Forest Classifier since it has produced 96.56% accuracy for producing accurate result.