# Phishing Website Detection Tool (Complete Version)

# Step 1: Import Required Libraries

import re

import pandas as pd

import tkinter as tk

from tkinter import messagebox

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestClassifier

from sklearn.metrics import classification\_report

# Step 2: Define Rule-Based Detection Function

def is\_phishing\_url(url):

    if re.search(r'@', url):

        return True

    if re.match(r'https?://\d{1,3}(\.\d{1,3}){3}', url):

        return True

    if len(url) > 75:

        return True

    if not url.startswith("https"):

        return True

    suspicious\_words = ['secure', 'login', 'account', 'update', 'free', 'verify', 'ebayisapi']

    for word in suspicious\_words:

        if word in url.lower():

            return True

    return False

# Step 3: Load Dataset and Train ML Model

def train\_ml\_model(csv\_file):

    df = pd.read\_csv(csv\_file)

    df['Label'] = df['Label'].map({'legitimate': 0, 'phishing': 1})

    vectorizer = CountVectorizer()

    X = vectorizer.fit\_transform(df['URL'])

    y = df['Label']

    X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

    model = RandomForestClassifier()

    model.fit(X\_train, y\_train)

    y\_pred = model.predict(X\_test)

    print("\nMachine Learning Model Evaluation:")

    print(classification\_report(y\_test, y\_pred))

    return model, vectorizer

# Step 4: Initialize ML Model

ml\_model = None

vectorizer = None

try:

    ml\_model, vectorizer = train\_ml\_model("phishing\_site\_urls.csv")

except Exception as e:

    print(f"[!] Error loading ML model: {e}")

# Step 5: Define Detection Logic (Rule + ML)

def detect\_url(url):

    if is\_phishing\_url(url):

        return "Suspicious (Rule-based)"

    elif ml\_model and vectorizer:

        features = vectorizer.transform([url])

        prediction = ml\_model.predict(features)[0]

        return "Phishing (ML)" if prediction else "Legitimate (ML)"

    else:

        return "Legitimate (Rule-based)"

# Step 6: Build GUI with Tkinter

def check\_url():

    url = entry.get()

    result = detect\_url(url)

    if "Phishing" in result or "Suspicious" in result:

        messagebox.showwarning("Alert", f"This URL is flagged: {result}")

    else:

        messagebox.showinfo("Safe", f"This URL appears safe: {result}")

root = tk.Tk()

root.title("Phishing Website Detector")

root.geometry("500x200")

tk.Label(root, text="Enter URL to check:", font=("Arial", 12)).pack(pady=10)

entry = tk.Entry(root, width=50)

entry.pack(pady=5)

tk.Button(root, text="Check URL", command=check\_url, font=("Arial", 12)).pack(pady=20)

root.mainloop()

<!DOCTYPE html>

<html lang="en">

<head>

    <meta charset="UTF-8">

    <title>Phishing URL Detector</title>

</head>

<body>

    <h2>Phishing Website Detection</h2>

    <form method="POST">

        <label>Enter URL:</label>

        <input type="text" name="url" required>

        <button type="submit">Check</button>

    </form>

    {% if result %}

    <h3>Result: {{ result }}</h3>

    {% endif %}

</body>

</html>

# Phishing Website Detection Tool (Web Version with Flask)

# Step 1: Import Required Libraries

import re

import pandas as pd

from flask import Flask, request, render\_template

from sklearn.feature\_extraction.text import CountVectorizer

from sklearn.model\_selection import train\_test\_split

from sklearn.ensemble import RandomForestClassifier

# Step 2: Rule-Based Detection Function

def is\_phishing\_url(url):

    if re.search(r'@', url):

        return True

    if re.match(r'https?://\d{1,3}(\.\d{1,3}){3}', url):

        return True

    if len(url) > 75:

        return True

    if not url.startswith("https"):

        return True

    suspicious\_words = ['secure', 'login', 'account', 'update', 'free', 'verify', 'ebayisapi']

    for word in suspicious\_words:

        if word in url.lower():

            return True

    return False

# Step 3: Train ML Model

def train\_ml\_model(csv\_file):

    df = pd.read\_csv(csv\_file)

    df['Label'] = df['Label'].map({'legitimate': 0, 'phishing': 1})

    vectorizer = CountVectorizer()

    X = vectorizer.fit\_transform(df['URL'])

    y = df['Label']

    X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

    model = RandomForestClassifier()

    model.fit(X\_train, y\_train)

    return model, vectorizer

# Step 4: Load Model and Vectorizer

ml\_model = None

vectorizer = None

try:

    ml\_model, vectorizer = train\_ml\_model("phishing\_site\_urls.csv")

except Exception as e:

    print(f"[!] Error loading ML model: {e}")

# Step 5: Detection Logic

def detect\_url(url):

    if is\_phishing\_url(url):

        return "Suspicious (Rule-based)"

    elif ml\_model and vectorizer:

        features = vectorizer.transform([url])

        prediction = ml\_model.predict(features)[0]

        return "Phishing (ML)" if prediction else "Legitimate (ML)"

    else:

        return "Legitimate (Rule-based)"

# Step 6: Flask Web Application

app = Flask(\_\_name\_\_)

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

def home():

    result = None

    if request.method == 'POST':

        url = request.form['url']

        result = detect\_url(url)

    return render\_template('index.html', result=result)

if \_\_name\_\_ == '\_\_main\_\_':

    app.run(debug=True)