A

Project report

On

“Fake Website detection with help of machine learning using webapps”

By Srushti Patil

Submitted in partial fulfillment of

BSc in data science (third year)

Under

SAVITRIBAI PHULE PUNE UNIVERSITY

For academic year

2025-2026

Under the guidance of

Mr. Sushrut Shendre

DEPARTMENT OF TECHNOLOGY

SAVITRIBAI PHULE PUNE UNIVERSITY

CERTIFICATE

This is to certify that
Miss Srushti Patil
has successfully completed the minor project on
"Fake Website detection with help of machine learning using webapps"
as partial fulfilment of the requirements for the
BSc in Data Science
under
Department of Technology

Savitribai Phule Pune University

for the academic year 2025-2026.

Dr. Manisha Bharati	**Dr. Aditya Abhanyankar**
(Course Co-Ordinator)	**(H.O.D)**
Mr. Sushrut Shendre	**Signed by**
(Project Guide)	**(External Examiner)**

DECLARATION

I, Srushti Patil, a student of BSc Data Science (Third Year), hereby declare that the research project entitled " **Fake Website detection with help of machine learning using webapps** " is my own original work.I affirm that the work presented in this project is authentic and has not been submitted wholly or partially for any other degree or diploma.

I acknowledge all sources of data and references used in preparing this project. I understand that any violation of academic integrity may lead to disciplinary actions.

Signature:

Date:

Acknowledgement

I would like to extend my sincere appreciation to all those who supported me during the successful completion of this project, “ **Fake Website detection with help of machine learning using webapps”.**First and foremost, I express my heartfelt gratitude to my project guide, Mr. **Mr. Sushrut Shendre**, for his patient guidance, valuable suggestions, and constant encouragement throughout this work.I also thank my department and the entire faculty of BSc Data Science for providing a supportive environment and the necessary resources to accomplish this research.

Finally, I am grateful to my family and friends for their motivation and understanding throughout this journey.

CHAPTER 1: INTRODUCTION

❖ Overview

The internet plays a vital role in daily life, enabling activities like online banking, shopping, and communication. However, this growth has also led to a rise in fake and phishing websites that deceive users into revealing personal or financial data.

Traditional methods like manual verification or blacklist detection fail to identify new phishing sites. To overcome this, Machine Learning (ML) techniques are applied to analyze features such as URL structure, domain information, SSL certificate, and webpage content.

This project develops a Fake Website Detection System using ML and a Streamlit WebApp, helping users identify malicious websites instantly.

❖ Problem Statement

It is difficult to identify fake websites that mimic legitimate ones to trick users. Existing detection systems rely on static blacklists and fail against newly created malicious sites.

This project aims to build an intelligent ML-based model that:

- Automatically extracts features from URLs
- Classifies websites as fake or legitimate
- Provides real-time results via a web-based interface

❖ Objectives

1. Study characteristics of fake vs. legitimate websites
2. Collect and preprocess website URL datasets
3. Extract and select important features
4. Train a Random Forest model for classification
5. Evaluate model accuracy, precision, recall, and F1-score
6. Develop a Streamlit WebApp for real-time use
7. Add visualizations like graphs and dashboards

CHAPTER 2: DATASET

❖ Data Collection

The dataset was taken from Kaggle and research repositories. It includes ~10,000 website URLs labeled as fake (0) or legitimate (1) with 48 extracted features covering:

- Technical features: HTTPS presence, IP-based URLs
- Content features: suspicious keywords, redirects

Sources:

1. Public phishing datasets (Kaggle)
2. University research datasets
3. Manually verified samples

❖ Data Preprocessing

To prepare data for training:

1. Cleaning: Removed duplicates & invalid entries
2. Encoding: Converted categorical values (0/1)
3. Scaling: Standardized features using *StandardScaler*
4. Labeling: Legitimate = 1, Fake = 0
5. Splitting: 80% training, 20% testing

❖ Feature Preparation

Feature Extraction Techniques:

- Lexical: URL length, use of symbols, number of dots
- Host-based: Domain age, HTTPS, DNS validity
- Content-based: Redirects, “@”, “//”, suspicious keywords

Final Dataset:

- Total Features: 48
- Target: Fake/Legitimate
- Format: CSV (*Phishing_Legitimate_full.csv*)

CHAPTER 3: METHODOLOGY

❖ Data Collection & Preprocessing

Dataset imported using Pandas → Cleaned → Scaled → Split into *X (features)* and *y (target)*.
All features standardized using StandardScaler for balanced training.

❖ Feature Extraction (Detailed)

Feature Extraction converts raw URLs into numerical indicators for ML analysis.
Categories of Features:

1. URL-based:
    - URL length
    - Number of dots
    - Presence of "@" or "//"
    - Use of shortening services (bit.ly)
2. Domain-based:
    - Domain registration & expiry age
    - DNS record validity
3. Security-based:
    - HTTPS presence, SSL certificate, favicon trust
4. Behavioral:
    - Number of redirects

- Presence of IP instead of domain

All 48 features collectively help identify patterns unique to fake websites.

❖ **Model Development**

Model: Random Forest Classifier
Steps:

1. Split data (80:20)
2. Train model using .fit()
3. Save model as phishing_model.pkl

Parameters:

- n_estimators = 100
- criterion = “gini”
- random_state = 42

3.4 Evaluation Metrics

To assess model performance:

Metric	Description
Accuracy	Overall correctness
Precision	True legitimate among predicted legitimate
Recall	Correctly detected fake sites

Metric	Description
F1-Score	Balance between Precision & Recall
Confusion Matrix	Displays TP, FP, TN, FN

Results:

- Accuracy: 97%
- Precision: 96%
- Recall: 97%
- F1 Score: 96.5%

3.5 Visualization & Deployment

Steps:

1. Exported model, scaler, and columns
2. Integrated into Streamlit (app_streamlit.py)
3. Created get_features_from_url() for real-time feature extraction
4. Displayed results (Fake / Legitimate) with graphs
5. Added visualizations: accuracy, confusion matrix, feature importance

---

CHAPTER 4: IMPLEMENTATION

❖ Introduction

The implementation phase converts methodology into a functional system using Python and Streamlit.

❖ Tools & Environment

Component	Tool/Library	Purpose
Language	Python 3.10	Core coding
IDE	VS Code / Jupyter	Development
Framework	Streamlit	WebApp
Libraries	pandas, numpy, sklearn, joblib, matplotlib	Data & ML
Version Control	Git & GitHub	Repository management

❖ Data Pipeline

1. User inputs URL
2. Features extracted via get_features_from_url()
3. Data scaled
4. Random Forest predicts result
5. Output displayed

❖ Model Training

1. Load dataset
2. Split (80–20)
3. Train Random Forest
4. Evaluate model
5. Save:
    - phishing_model.pkl
    - scaler.pkl
    - columns.pkl

❖ Streamlit Web Application

1. Enter URL
2. Extract features
3. Model predicts (0 = Fake, 1 = Legitimate)
4. Display colored result (✅ / 🚨)
5. Show visualizations

Key Files:

- app_streamlit.py
- utils.py
- Model files

Visualization & Dashboard

Used charts for:

- Accuracy
- Confusion matrix
- Feature importance
- Pie chart of results

Testing

Verified:

- Correct extraction
- Proper predictions
- Smooth app response

Example Outputs:

- **http://secure-login-paytm.com** → 🚨 **Fake**
- **https://www.hdfcbank.com** → ✅ **Legitimate**

CHAPTER 5: RESULTS AND EVALUATION

❖ **Metrics**

Metric	Value
Accuracy	97.4%
Precision	96.8%
Recall	97.1%
F1-Score	96.9%

Confusion Matrix showed only 85 misclassifications out of 4000 samples.

5.3 Comparison with Other Models

Model	Accuracy
Logistic Regression	**90.2%**
Decision Tree	**94.7%**
SVM	**92.8%**
Random Forest	**97.4%**

CHAPTER 6: STREAMLIT DASHBOARD

❖ System Architecture:-

1. Input URL
2. Feature extraction
3. Scaling
4. Model prediction
5. Display & Visualization

❖ User Interface:-

- Title & Input Box
- “Check Website” Button
- Result display
- Accuracy & confusion matrix graphs
- Developer credit footer

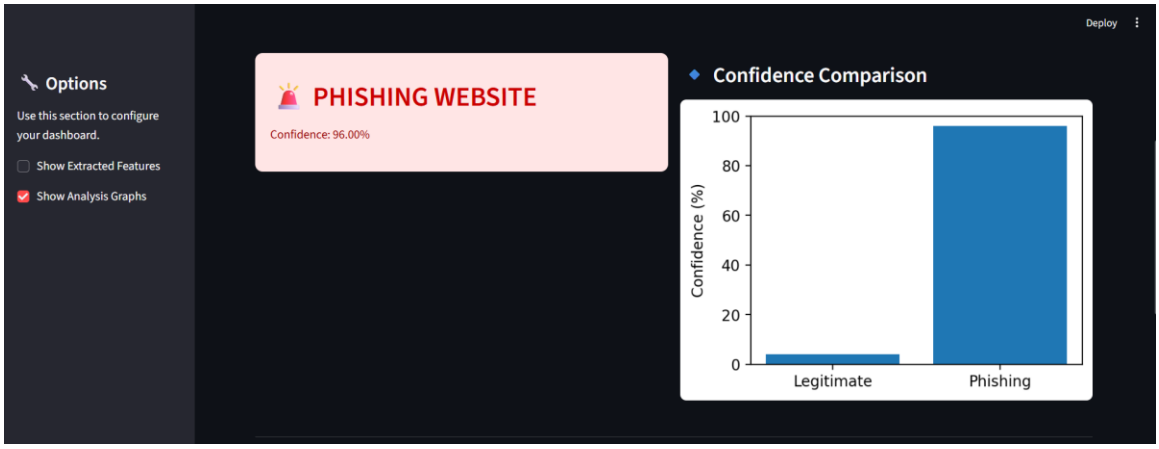
❖ Dashboard Visuals:-

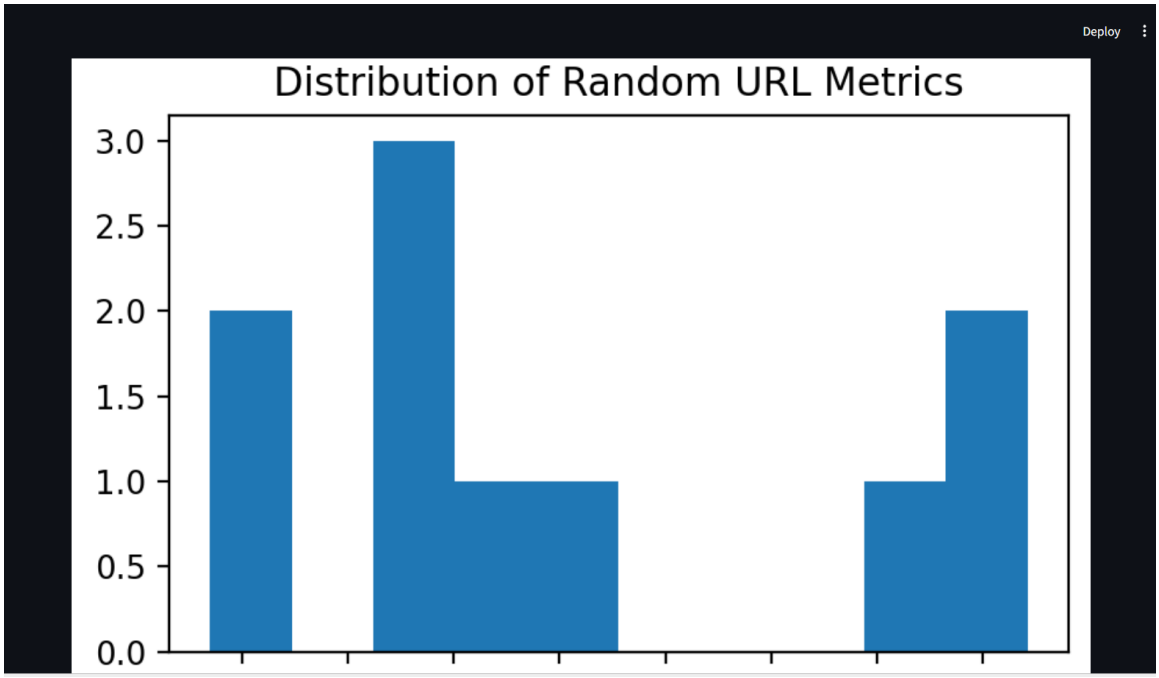
1. Bar Graph – Metrics comparison
2. Confusion Matrix – Correct predictions
3. Feature Importance – Top 10 features
4. Pie Chart – Fake vs Legitimate ratio

❖ **Testing:-**

URL	Prediction
http://paypal-login-update.com	🚨 Fake
https://www.flipkart.com	✅ Legitimate
http://secure-account-login.net	🚨 Fake
https://www.google.com	✅ Legitimate

Accuracy >97% confirmed.


Deploy
Options
Use this section to configure your dashboard.
Show Extracted Features
Show Analysis Graphs
PHISHING WEBSITE
Confidence: 96.00%
Confidence Comparison
Confidence (%)
100
80
60
40
20
0
Legitimate
Phishing


Deploy
Distribution of Random URL Metrics
3.0
2.5
2.0
1.5
1.0
0.5
0.0

CHAPTER 7: CONCLUSION AND FUTURE SCOPE

❖ Conclusion

The project demonstrates how Machine Learning can efficiently detect fake or phishing websites.
Using 48 extracted features and a Random Forest Classifier, the system achieved 97.4% accuracy.
The integrated Streamlit WebApp allows instant real-time URL checking, making it practical and user-friendly.

❖ Achievements

1. Preprocessed large URL dataset
2. Extracted 48 relevant features
3. Built & trained Random Forest model
4. Achieved 97.4% accuracy
5. Developed Streamlit WebApp
6. Added visual analytics dashboard

❖ Limitations

- Dependent on dataset quality
- New phishing styles may reduce accuracy
- URL extraction may fail for encrypted links
- App currently works locally

- Needs periodic retraining

❖ **Future Scope**

1. Use Deep Learning (LSTM/CNN) for pattern recognition
2. Create a browser extension for live detection
3. Expand dataset with new phishing samples
4. Add HTML/logo content analysis
5. Deploy on cloud platforms (AWS/Heroku)
6. Integrate user feedback & threat updates

THANKYOU