**Spam and Ham Email Detection – Project Report**

**📌 Project Title:**

**Spam and Ham Email Detection Using ML and NLP**

**🧠 Objective:**

The primary goal of this project is to build an intelligent system that can automatically classify incoming emails as either **Spam** (unwanted) or **Ham** (legitimate). This helps users avoid phishing, scams, and unnecessary clutter in their inbox.

**💡 Motivation:**

Email spam is a serious issue that can lead to security threats, time wastage, and poor communication. Manual filtering is impractical due to the volume of emails received daily. Therefore, an automated, reliable spam detection system is essential.

**🔍 Problem Statement:**

Given a dataset of labeled emails, the task is to train a machine learning model that can analyze the **text content of an email** and predict whether it is spam or ham.

**🗃️ Dataset:**

* **Source**: [UCI Machine Learning Repository – SMS Spam Emails Collection](https://archive.ics.uci.edu/ml/datasets/sms+spam+collection)
* **Size**: ~5,500 emails (SMS messages)
* **Features**:
  + Label: spam or ham
  + Message: Text of the email or message

**🔧 Technologies Used:**

* **Programming Language**: Python, Flask, HTML, CSS, Java Script
* **Libraries**:
  + Numpy, Pandas– for data handling
  + sklearn– for ML models and evaluation
  + nltk– for text preprocessing
  + matplotlib, seaborn– for visualization
* **Modeling Algorithms**:
  + Naive Bayes (MultinomialNB)
  + Support Vector Machine (SVM)
  + Logistic Regression
  + Random Forest
  + XG Boost

**🔧 Project Structure:**

Spam\_and\_Ham\_Email\_Detection\_project/

│

├── static/

├─── css

│ └──── styles.css # CSS for styling

│ ├───── js

└──── script.js # JavaScript for interactivity

│

├── templates/

│ └── index.html # HTML page served by Flask

│

├── app.py # Flask application

├── Logistic Regression\_model.pkl # Trained ML model

├── tfidf\_vectorizer.pkl # TF-IDF vectorizer

**🔄 Methodology:**

**1. Data Preprocessing**

* Removal of special characters, numbers, punctuation
* Conversion to lowercase
* Stop word removal
* Tokenization and stemming (using NLTK)
* Vectorization using **TF-IDF**

**2. Exploratory Data Analysis (EDA)**

* Visualizing class distribution (spam vs ham)
* Word clouds for spam and ham messages
* Common spam keywords (like “win”, “free”, “cash”)

**3. Model Training & Testing**

* Dataset split into training and testing (80:20 ratio)
* Model trained on TF-IDF features
* Accuracy, Precision, Recall, F1-score, and Confusion Matrix used for evaluation

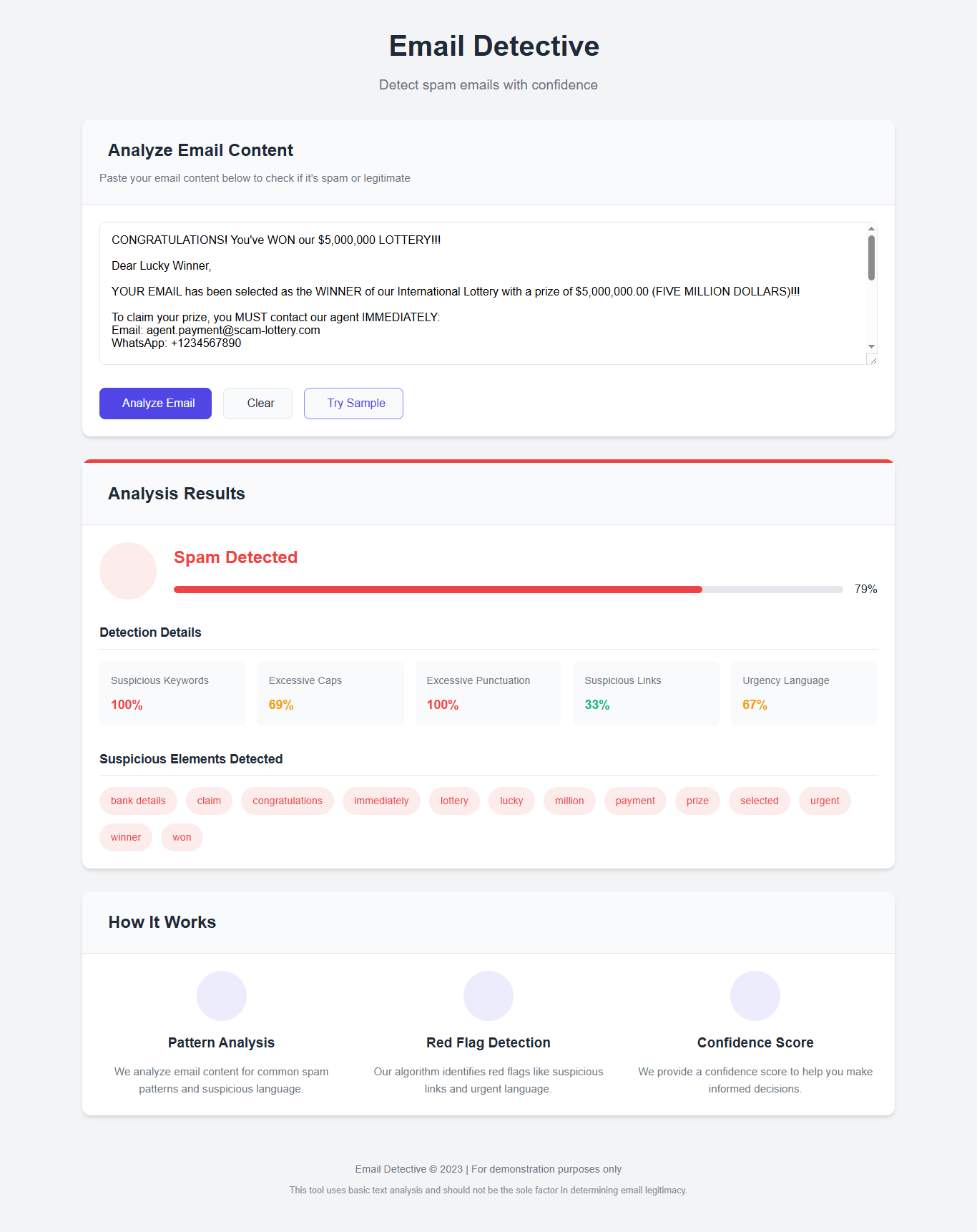
**📊 Results:**

| **Classes** |  |  | **Spam** |  |  | **Ham** |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Model** | **Accuracy** | **Precision** | **Recall** | **F1-Score** | **Precision** | **Recall** | **F1-Score** |
| Naive Bayes | 97.2% | 100% | 80% | 89% | 97% | 100% | 98% |
| SVM | 97.3% | 98% | 82% | 89% | 97% | 100% | 99% |
| Logistic Regression | 95.5% | 98% | 68% | 80% | 95% | 100% | 97% |
| Random Forest | 97.4% | 98% | 83% | 90% | 97% | 100% | 99% |
| XG Boost | 97.2% | 95% | 84% | 89% | 98% | 99% | 98% |

**Best Performing Model**: ✅ *Multinomial Naive Bayes*

**✅ Conclusion:**

This project demonstrates how machine learning can effectively detect spam emails with high accuracy. The **Naive Bayes model**, due to its performance and efficiency with text data, proved to be the best choice for this task.

**Screenshot of Email detective website **