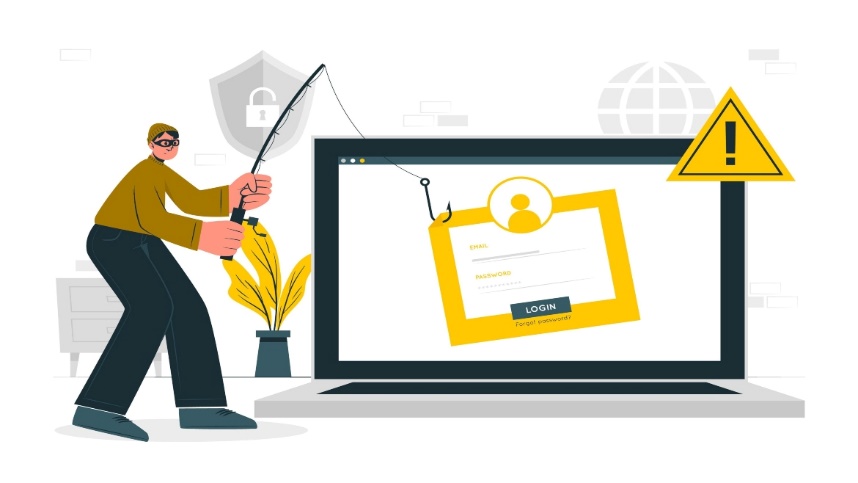
High Level Document Phishing Domain Detection



Mukhesh Kuna

# Content

1. [Introduction… 3](#_TOC_250010)
   1. Why this High Level Design Document 3
   2. [Scope 3](#_TOC_250009)
2. General Desciption 4
   1. [Product Perspective… 4](#_TOC_250008)
   2. [Problem Statement 4](#_TOC_250007)
   3. Proposed Solutions… 4
   4. Further Improvement 4
   5. Data Requirement 4
   6. [Tool Used… 4](#_TOC_250006)
   7. [Constraints… 5](#_TOC_250005)
   8. [Assumptions… 5](#_TOC_250004)
3. [Design Details… 6](#_TOC_250003)
   1. [Process Flow… 6](#_TOC_250002)
      1. Model Process flow 4
      2. Deployment Process. 7

3.4 Event Logs… 7

2.5 Error Handling 7

1. [Performance… 8](#_TOC_250001)
   1. Reusability 8
   2. Application Capability 8
   3. Resource Utilization 8
   4. Deployment 8
2. [Conclusion… 9](#_TOC_250000)

# Introduction

## Why this High-Level Design Documentation

The primary objective of this High-Level Design (HLD) Document is to augment the existing project description with essential details that provide a suitable framework for the coding phase. This document serves as a proactive means of identifying and addressing potential contradictions before the coding phase, acting as a reference manual for understanding high-level module interactions.

The HLD Document will encompass the following key aspects:

Design Aspects: It will comprehensively present and elucidate all design elements, offering detailed insights into how the system's components are organized and function.

User Interface: The document will outline the user interface that is being implemented, providing a clear representation of how users will interact with the system.

Hardware and Software Interfaces: Descriptions of the interfaces with hardware and software components will be provided, enabling a holistic view of the system's connectivity.

Performance Requirements: This section will define the performance expectations and metrics that the system is expected to meet.

Design Features and Architecture: The document will delve into the architectural aspects of the project, elucidating the structural organization and design features that underpin the system's functionality.

## Scope

Our software system is a web application specifically designed to assist users in determining the authenticity of websites. It employs a predictive model that evaluates the provided information, helping users discern whether a website is legitimate or potentially malicious.

The user interface includes several categories to fill in, such as the Length of the URL, Length of the Directory, Length of the File, Activation Domain Time, and Expiration Domain Time. The model utilizes these input features to make a prediction regarding the website's authenticity.

Our priority is to ensure that all the required features are available for the prediction process, allowing for optimal accuracy and the opportunity for companies to maximize their profits. By providing users with a reliable tool to assess the legitimacy of websites, we contribute to a safer online experience and enhanced security in the digital landscape.

# General Description

## Product Perspective

This Phishing Domain Detection System is a machine learning based model which will predict weather the website is real or malicious on the basis of user’s input values.

## Problem statement

Phishing is a type of fraud in which an attacker impersonates a reputable company or person through email or other communication channels to obtain sensitive information such as login credentials or account details. Attackers prefer phishing because convincing someone to click a seemingly authentic malicious link is easier than bypassing a computer's security measures. The primary objective is to predict the authenticity of domains, distinguishing between genuine and malicious ones.

## Proposed Solution

This system relies on a set of critical features for phishing detection, encompassing Length of URL, Length of Directory, Length of File, Activation Domain Time, Expiration Domain Time, and sender policy framework (SPF). By leveraging these essential features, the system is equipped to make predictions regarding the authenticity of websites, differentiating between genuine and potentially malicious domains.

The predictive model analyzes these features to determine whether a website is legitimate or malicious. It is this amalgamation of factors that empowers the system to safeguard users by providing timely alerts and ensuring their online security.

The inclusion of these key features is essential to achieve optimal accuracy and to maximize the system's efficacy and value for the company and its users.

## Further Improvements

The dataset consists of URL values rather than complete URL links. Our primary objective is to address this use case using a machine learning algorithm as the most optimized solution. In the future, as the need arises to work with URL sites and different categories, we may explore the implementation of deep-learning algorithms to achieve the best possible solutions.

Our approach is flexible, allowing us to adapt to evolving requirements and employ the most suitable algorithms for the task at hand. The initial focus is on efficient machine learning, and deep learning becomes an option for more complex scenarios. This adaptability ensures our ability to provide optimal solutions while staying at the forefront of technology and innovation.

## Data Requirements

This data set consist of 88,647 websites labelled as legitimate or phishing and allow the researchers to train their classification models, build phishing detection systems.

For more details of the dataset visit:

[Datasets for phishing websites detection - ScienceDirect](https://www.sciencedirect.com/science/article/pii/S2352340920313202)

[Datasets for phishing websites detection - ScienceDirect](https://www.sciencedirect.com/science/article/pii/S2352340920313202)

## Tool Used

1) Socket 2) Pandas 3) Numpy 4) Sklearn 5) Matplotlib / Seaborn 6) Github 7) Visual Studio

8) Jupyter Notebook 9) Joblib 10) Streamlit

## Constraints

In our system, we will focus on using a limited set of features, selected through the Feature Selection method. This selection process ensures that we are working with the most relevant and influential features for the task at hand. The specific method we will employ for feature selection is the Variance Inflation Factor (VIF) method.

Our system's primary objective is to predict whether a website is genuine or fraudulent. By carefully selecting features through the VIF method, we aim to streamline the prediction process, improving efficiency and accuracy. This approach ensures that we are utilizing the most essential data points for making informed predictions.

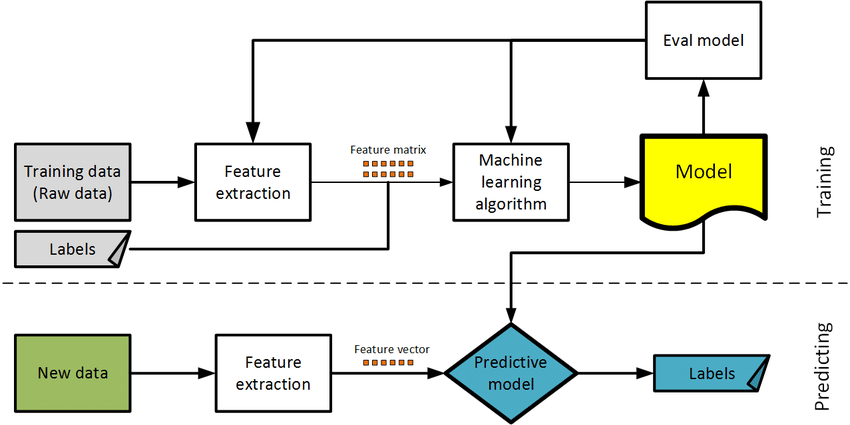
## Assumptions

# The values in our dataset have already been extracted through web scraping. Our primary responsibility is to apply a series of data processing steps, including Data Pre-processing, Feature Selection, Feature Scaling, Model Building, and ultimately deploying the model using Streamlit on local server.

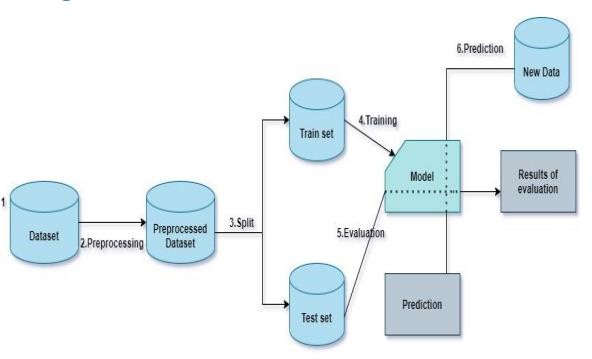
# This process involves preparing and optimizing the data, selecting the most relevant features, scaling them as necessary, building a predictive model, and finally making the system accessible through a web application created with Streamlit on local server. Our role is to ensure that the entire pipeline, from data to deployment, operates seamlessly and effectively.

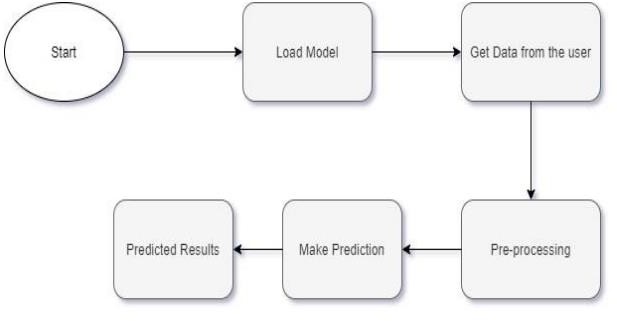
# Design Details

## Process Flow

Based on the use-case, we will use a machine learning base model. Below is the process flow diagram is as shown below.

## Model Training and Evaluation



* 1. **Deployment Proces**

## Event log

In our system, it is imperative to maintain a comprehensive log of events to keep users informed about the internal processes that are running. The initial steps for achieving this are as follows:

Identification of Logging Steps: The system intelligently identifies the specific steps at which logging is required to ensure that it captures crucial events.

Logging of System Flow: Our system is designed to log each and every step of the system's flow. This comprehensive approach ensures that no critical event or action goes unrecorded.

Logging Method Selection: As a part of system flexibility, developers have the autonomy to select their preferred logging method. For our system, we have chosen to implement File logging, which offers an effective means of capturing and storing log data.

Maintaining System Performance: Importantly, our system is designed in a way that ensures it does not hang or experience performance degradation due to the use of file logging. Logging is not just an optional feature; it is a mandatory component of our system. Its primary purpose is to facilitate efficient debugging and issue resolution, ensuring that our system operates smoothly and reliably.

## Error Handling

In the event of an error, our system is designed to provide an explanatory message that clarifies the nature of the issue. An error, in this context, is defined as any occurrence that deviates from the normal and intended usage of the system.

This approach ensures that users are not left in the dark when errors occur. Instead, they receive clear and informative explanations, enabling them to understand what went wrong and take appropriate actions. The system's commitment to transparency and user support extends to error handling, making it a valuable tool for both resolving issues and enhancing user experience.

# Performance

## Reusability

The code and components we've developed are designed with reusability in mind. They have the inherent capacity to be reused seamlessly when faced with a similar problem statement. This means that the solutions we've crafted can be efficiently adapted and applied to address analogous challenges and scenarios.

Our commitment to reusability not only optimizes development time and effort but also fosters a culture of efficiency and innovation. By creating solutions that are versatile and easily transferable, we ensure that our work has a lasting and valuable impact, extending beyond the current problem statement to tackle future challenges.

## Application Compatibility

For this project, various components will be seamlessly interconnected using Python as the interface. Each component is assigned a specific task, and Python plays a crucial role in orchestrating the smooth exchange of information between these components.

Python serves as the linchpin that ensures the efficient flow of data and functionality across the project. It acts as the central hub, enabling cohesive collaboration and interaction among the diverse components, each working toward a common objective.

## Resource Utilization

When a task is initiated, it will typically utilize all available processing power until the function is successfully completed. This maximizes system resources and optimizes the efficiency of task execution.

## Deployment

Deployment has been done using Streamlit on local server.

# Conclusion

## This project harnesses data-driven techniques, feature extraction, and machine learning to create a robust tool for detecting and guarding against phishing domains. The XGBoost model, trained on a diverse dataset, holds the potential to offer real-time, dependable protection against malicious websites, thereby enhancing the safety of online experiences for users.

## The results from phishing domain detection, utilizing the random forest model, are highly promising, showcasing an accuracy rate of 96.7%, precision of 95%. These metrics reflect the model's outstanding performance in accurately identifying phishing domains. They affirm the model's high accuracy in recognizing malicious domains and its reliability in distinguishing whether a domain is involved in phishing activities or not.

## Overall, this model serves as a valuable tool for shielding users against phishing threats. Its exceptional accuracy and precision make it a compelling choice for deployment in real-world scenarios.

# 