# **URL ANALYZER FOR MALWARE AND VULNERABILITY DETECTION**

By

Wajahat Ayaan

2021-GCUF-03469

Zakar Ullah

2020-GCUF-076750

Uzaif Hadi

2021-GCUF-03468

Thesis Submitted in Partial Fulfillment

of the Requirements for the

DEGREE OF BACHELOR OF

SCIENCE

IN

DATA SCIENCE

A logo of a school

Description automatically generated

DEPARTMENT OF CENTER OF DATA SCIENCE

GC UNIVERSITY, FAISALABAD

Session: 2021-25

## **CERTIFICATE BY SUPERVISOR**

This is to certify that the Final Year Project Proposal submitted by **Mr. Wajahat Ayaan (Registration No. 2021-GCUF-03469), Mr. Zakarullah (Registration No. 2021-GCUF-076750), and Mr. Uzaif Hadi (Registration No. 2021-GCUF-034668)** has been reviewed and found satisfactory in both content and format, as per the prescribed guidelines. We hereby recommend that it be processed for evaluation.

**SUPERVISORY COMMITTEE:**

**DR. KHURRAM ZEESHAN:**

**Supervisor:**  Sig: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**SIR RAO SOHAIL:**

Member -1  Sig: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**MAM. AYESHA KHALIQ:**

Member -2 Sig: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Table of Contents

[**Abstract**](#_Toc184286628)

[**PROJECT INTRODUCTION** 1](#_Toc184286629)

[1.1 Background 1](#_Toc184286630)

[1.2 Objectives 1](#_Toc184286631)

[1.3 Problem Statement 2](#_Toc184286632)

[1.4 Work Breakdown Structure (WBS) 2](#_Toc184286633)

[1.5 Project Scope 4](#_Toc184286634)

[1.6 Document Overview 4](#_Toc184286635)

[**PRELIMINARY REQUIREMENTS** 5](#_Toc184286636)

[2.1 Data Sources 5](#_Toc184286637)

[2.2 Tools and Technologies 6](#_Toc184286638)

[**TECHNICAL APPROACH** 7](#_Toc184286639)

[3.1 Data Preprocessing Steps 7](#_Toc184286640)

[3.2 Feature Extraction 8](#_Toc184286641)

[3.3 Machine Learning Model Development 9](#_Toc184286642)

[**LITERATURE REVIEW / EXISTING SYSTEM STUDY** 10](#_Toc184286643)

[4.1 Phishing and Malware Attacks 10](#_Toc184286644)

[4.2 Machine Learning in URL Detection 11](#_Toc184286645)

[**METHODOLOGY / IMPLEMENTATION** 13](#_Toc184286646)

[5.1 System Design 13](#_Toc184286647)

[5.2 User Interface 14](#_Toc184286648)

[5.3 Interface Accessibility: 14](#_Toc184286649)

[**MANAGEMENT APPROACH** 15](#_Toc184286650)

[6.1 Project Plan 15](#_Toc184286651)

[6.2 Project Risks 16](#_Toc184286652)

[**REFERENCES** 17](#_Toc184286653)

## **Abstract**

This report discusses the development of a tool that can analyze URLs to detect malware, viruses, and vulnerabilities using machine learning (ML) and cybersecurity techniques. In today’s digital age, cybersecurity threats like phishing, malware, and hacking have become increasingly prevalent, especially through malicious URLs. This tool is designed to automatically evaluate URLs to identify these threats, providing an effective solution for real-time security. This report covers the tool’s design, implementation, and testing process, `explaining the methodologies and technologies applied to develop an efficient solution for identifying harmful URLs.

Chapter 01

## **PROJECT INTRODUCTION**

## **1.1 Background**

In recent years, the increase in internet usage has led to a corresponding rise in cybersecurity threats, including phishing attacks and malware distribution. URLs (Uniform Resource Locators) are often the primary vehicles for these attacks. Attackers exploit vulnerabilities in web applications, social engineering tactics, and deceptive URLs to compromise user security. With the advent of machine learning, there is an opportunity to analyze and predict the likelihood of a URL being malicious or containing vulnerabilities.

**Table 1.1: Statistics on Cybersecurity Threats**

|  |  |  |  |
| --- | --- | --- | --- |
| Year | Phishing Attacks | Malware Attacks | Data Breaches |
| 2020 | 1.5 million | 11 million | 37 billion |
| 2021 | 2.1 million | 15 million | 40 billion |
| 2022 | 3.5 million | 20 million | 50 billion |
| 2023 | 4 million | 25 million | 60 billion |

## **1.2 Objectives**

**The primary objectives of this project are:**

* To develop an AI-powered tool that can analyze URLs to detect phishing attempts and malware.
* To utilize machine learning models to classify URLs as malicious or benign.
* To provide real-time vulnerability detection for web applications by analyzing input URLs.
* To enhance user awareness about potential threats by providing detailed reports on detected vulnerabilities.

## **1.3 Problem Statement**

Despite advancements in cybersecurity, users remain vulnerable to phishing and malware attacks due to the sophistication of these threats. Existing solutions often rely on signature-based detection methods that fail to identify new or obfuscated attacks. This project seeks to address these gaps by leveraging machine learning to provide proactive URL analysis and vulnerability detection.

## **1.4 Work Breakdown Structure (WBS)**

**Phase 1: Research and Data Collection**

This phase involves reviewing existing URL analysis tools and datasets related to phishing and malware detection. Data collection includes gathering URLs from sources like PhishTank and OpenPhish, cleaning and preprocessing the data, and addressing class imbalance. It also focuses on reviewing common web application vulnerabilities, gathering data from OWASP and NVD.

**Phase 2: Design and Development**

In this phase, the architecture for the URL analysis tool is designed, integrating machine learning models and vulnerability detection. The machine learning models are developed using Python and TensorFlow/Keras. The user interface (UI) is designed to be intuitive, allowing users to input URLs and view results.

**Phase 3: Model Optimization**

This phase focuses on fine-tuning the machine learning models for better accuracy and speed. The models are optimized to improve real-time detection, ensuring they can process URLs quickly. The vulnerability detection module is refined to identify SQL Injection, XSS, and other web application vulnerabilities.

**Phase 4: Testing and Evaluation**

Testing involves checking the accuracy of the URL detection system with various URL types and performing integration testing. The system is tested for seamless interaction between components, and user feedback is gathered to assess usability and performance.

**Phase 5: Deployment**

The tool is deployed as a web application, ensuring secure access and data protection. Documentation, including user manuals and technical guides, is prepared. Post-deployment monitoring ensures the system functions smoothly, and user feedback helps identify further improvements.

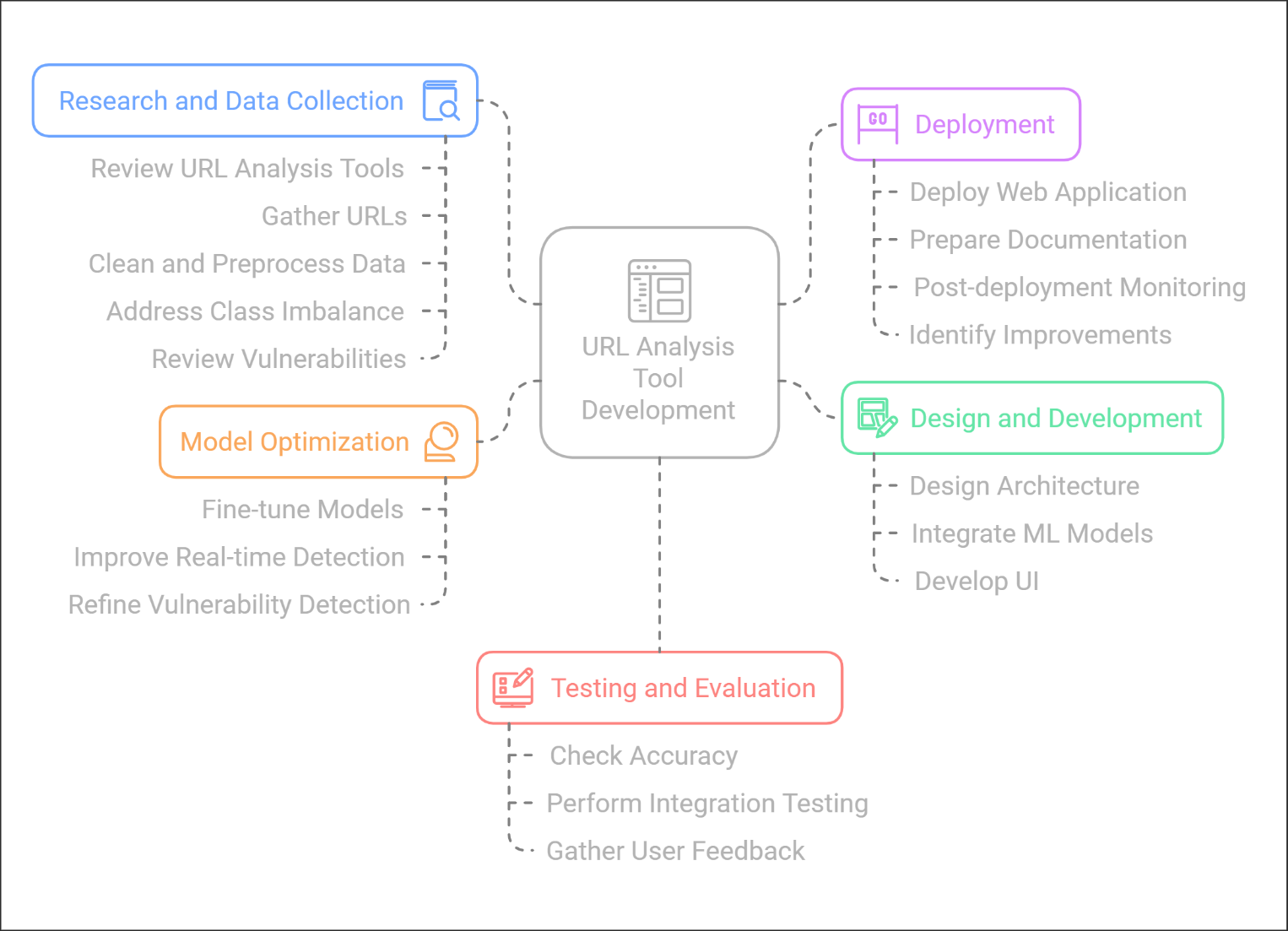


Figure 1.1 URL Analysis Tool Development Process

## **Project Scope**

This project focuses on the development of an AI-powered URL analyzer that targets the following aspects:

* **URL Analysis:** Examination of URLs to detect potential phishing and malware threats.
* **Vulnerability Detection:** Identifying vulnerabilities in web applications using
* provided URLs.
* **User Interface:** Designing an intuitive UI for users to interact with the tool.
* **Reporting:** Generating comprehensive reports summarizing findings and recommendations.

## **1.6 Document Overview**

This chapter introduces the project, focusing on the development of an AI-powered tool for detecting phishing and malware threats through URL analysis. It begins with the Background, which underscores the growing prevalence of cybersecurity threats due to increased internet usage. Statistical data highlights the alarming rise in phishing attacks, malware incidents, and data breaches over recent years, demonstrating the need for advanced cybersecurity solutions.

Chapter 02

## **PRELIMINARY REQUIREMENTS**

## **2.1 Data Sources**

To train and evaluate the machine-learning model for URL analysis and vulnerability detection, the following datasets will be used:

**1.1 Phishing and Malware URLs**

* **PhishTank**: Verified phishing URLs.
* **OpenPhish**: Real-time phishing data.
* **CVE Database**: Known security vulnerabilities.

**1.2 Legitimate URLs**

* **Alexa Top Sites**: List of trusted websites.
* **Google Safe Browsing API**: URLs flagged as safe or malicious.

**1.3 Synthetic Data Generation**

* Techniques to create synthetic phishing URLs for dataset balancing.

**1.4 Vulnerability Data**

* **OWASP Top Ten**: Critical web application security risks.
* **NVD**: Information on publicly disclosed vulnerabilities.

**1.5 Threat Intelligence Feeds**

* **AlienVault OTX**: Real-time threat intelligence.
* **VirusTotal**: Aggregates information from multiple antivirus engines.

These datasets ensure a comprehensive, up-to-date training environment for the AI-powered URL analyzer.

## **2.2 Tools and Technologies**

**Table 2: Tools and Technologies**

To achieve the objectives of this project, various tools and technologies were utilized for development, analysis, and deployment. Each tool was selected to address specific project requirements and ensure efficiency, scalability, and accuracy. Table 2 outlines the key tools and their purposes:

|  |  |
| --- | --- |
| **Tool/Technology** | **Purpose** |
| **Python** | Development of machine learning models, preprocessing, and backend logic. |
| **TensorFlow/Keras** | Training and fine-tuning deep learning models for URL analysis. |
| **Scikit-learn** | Data preprocessing, feature extraction, and additional machine learning support. |
| **Burp Suite** | Scanning for vulnerabilities in web applications and analyzing threats. |
| **Flask/Django** | Building the web-based user interface (UI) for user interaction. |
| **MySQL/MongoDB** | Storing detected vulnerabilities, system logs, and generated reports. |
| **PhishTank API** | Retrieving real-time phishing URL datasets for analysis and model training. |
| **OpenPhish API** | Integrating with real-time threat intelligence for phishing detection. |
| **Matplotlib/Seaborn** | Visualizing data distributions, trends, and performance metrics. |
| **Nessus API** | Conducting vulnerability scans for advanced system analysis. |
| **Jupyter Notebook** | Prototyping machine learning models and conducting exploratory data analysis. |
| **GitHub** | Version control and collaborative development of project code. |

Chapter 03

## **TECHNICAL APPROACH**

The technical approach outlines the methodology for data handling, machine learning (ML) model development, and vulnerability detection. Each step is carefully designed to ensure robustness, scalability, and high performance.

## **3.1 Data Preprocessing Steps**

* **Data Cleaning**:
  + Removing duplicates.
  + Handling missing or inconsistent values.
* **Normalization**: Scaling numerical features like URL length and domain age.
* **Categorization**: Labeling data into phishing, malware, and safe categories.

**Diagram 3.1**: Data Collection and Preprocessing Flowchart

A diagram of data cleaning

Description automatically generated

## **3.2 Feature Extraction**

Feature extraction transforms raw data into structured input for ML models. Key features for phishing and malware detection include:

**1 URL-Based Features**

1. **Length of URL**: Longer URLs are often malicious.
2. **Presence of Special Characters**: E.g., "@" or "%".
3. **Domain Reputation**: Based on known blacklists and WHOIS records.

**2 Content-Based Features**

1. **Keywords in URL**: Words like "login", "verify", or "secure".
2. **IP-Based URLs**: Direct IP usage instead of domain names.

**3 Behavioral Features**

1. **Frequency of Access**: Higher access rates can indicate phishing campaigns.
2. **Domain Registration Age**: Shorter registration periods often correlate with malicious activity.

**Table 3.1: Example Features**

|  |  |
| --- | --- |
| Feature Name | Description |
| URL Length | Total characters in the URL |
| Has Suspicious Word | Binary: Presence of phishing keywords |
| Domain Registration Age | Days since domain registration |
| Count of Special Characters | E.g., "-", "@", or "%". |

## **3.3 Machine Learning Model Development**

The heart of the system is the ML model, which classifies URLs and identifies threats.

**3.3.1 Model Selection**

Several algorithms will be tested:

1. **Logistic Regression**: Baseline model for binary classification.
2. **Random Forest**: Robust model for non-linear feature interactions.
3. **Neural Networks**: Advanced model for complex patterns in data.

**3.4 Vulnerability Detection**

The system will include a vulnerability detection module to identify system flaws beyond phishing and malware.

**3.4.1 Methodology**

1. **Integration with Existing Tools**:
   * Using tools like **Burp Suite** or **Nessus** for vulnerability scans.
   * APIs for real-time vulnerability reporting.
2. **Types of Vulnerabilities Detected**:
   * **SQL Injection**: Malicious database queries.
   * **Cross-Site Scripting (XSS)**: Injecting scripts into web applications.
   * **Broken Authentication**: Issues with user sessions and logins.

**3.4.2 Process Flow**

1. Input: URL or application for scanning.
2. ML Model Prediction: Detect phishing or malware.
3. Vulnerability Scanning: Identify vulnerabilities in associated systems.

CHAPTER 04

## **LITERATURE REVIEW / EXISTING SYSTEM STUDY**

## **4.1 Phishing and Malware Attacks**

Phishing and malware attacks are among the most prevalent cybersecurity threats today. Phishing attacks exploit social engineering techniques to deceive users into providing sensitive information, such as login credentials or financial details, by impersonating legitimate entities. Similarly, malware attacks involve the distribution of malicious software to compromise systems, steal data, or disrupt operations.

**Table 1: Comparison of Common Cyber Threats**

|  |  |  |  |
| --- | --- | --- | --- |
| **Threat Type** | **Primary Method of Attack** | **Detection Challenges** | **Example Tools for Mitigation** |
| Phishing | Social engineering, URL spoofing | Realistic impersonations, zero-day links | Email filters, blacklist URLs |
| Malware | URL-based downloads, ads | Polymorphic malware, drive-by-downloads | Antivirus, sandboxing |

Recent studies have highlighted the increasing sophistication of phishing campaigns, leveraging realistic website impersonations and URL obfuscation techniques to evade detection. For instance, attackers often embed malicious URLs in emails or online advertisements to lure users into clicking. The global increase in malware incidents, coupled with the growing complexity of these attacks, has led to the development of various defensive mechanisms such as blacklists, heuristic detection, and signature-based tools. However, these traditional systems often fail against novel or zero-day threats.

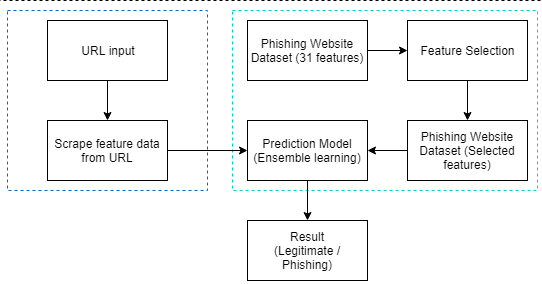
## **4.2 Machine Learning in URL Detection**

Machine learning (ML) has emerged as a powerful tool in combating phishing and malware attacks. By analyzing large datasets of URLs, machine learning algorithms can identify patterns and features indicative of malicious behavior. Supervised learning techniques, such as decision trees, support vector machines (SVM), and neural networks, have been employed to classify URLs as benign or malicious based on features like domain age, character distribution, and URL length.

Several studies have proposed hybrid approaches combining ML with heuristic methods to improve detection accuracy. For example, ensemble learning techniques, which combine predictions from multiple models, have shown promise in increasing robustness against adversarial attacks. Additionally, deep learning models, such as convolutional neural networks (CNNs) and recurrent neural networks (RNNs), are being explored for their ability to extract features automatically and adapt to new threats over time.

However, challenges remain, including the need for labeled datasets, handling imbalanced classes (as benign URLs vastly outnumber malicious ones), and maintaining real-time detection speeds without compromising accuracy.

**4.1 Diagram A flowchart showing the ML-based URL detection process:**



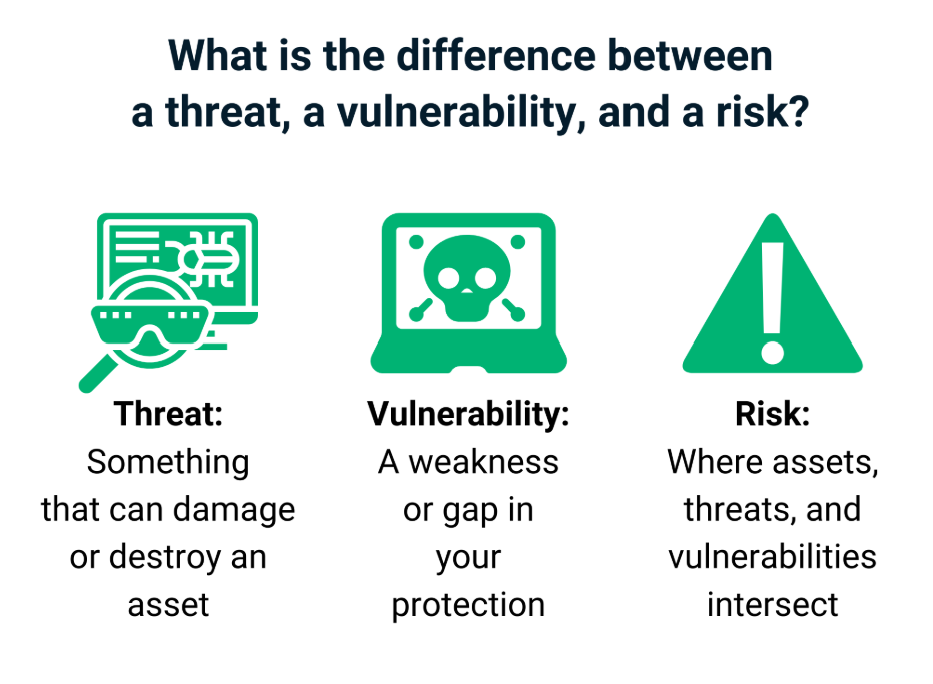
**4.3 Key Differences Between Vulnerability Detection and Malware Identification:**

* + Vulnerability Detection involves identifying weaknesses or flaws in systems, networks, or applications that attackers could exploit. It focuses on spotting potential entry points before an attack occurs.
  + Malware Identification is the process of detecting and analyzing malicious software (malware) designed to harm or compromise systems, such as viruses, Trojans, or ransomware.

***Examples*** *of issues in vulnerability detection include missing security patches, insecure configurations, and open ports.*

***Examples*** *of malware identified include ransomware, spyware, worms, and rootkits.*

***Vulnerability detection is a proactive step***while **Malware *identification is a reactive step.***Since the infection was allowed to happen by a vulnerability in the website/network.



Chapter 05

## **METHODOLOGY / IMPLEMENTATION**

The implementation of the AI-Powered URL Analyzer for Malware and Vulnerability Detection involves the development of a comprehensive system to address the challenges outlined in earlier chapters. This chapter details the system design and user interface components that form the backbone of the solution.

## **5.1 System Design**

The system design is structured to ensure efficient processing, real-time analysis, and accurate detection of malicious URLs and vulnerabilities. The architecture integrates multiple components working in unison to deliver optimal performance.

1. **System Architecture**  
   The architecture follows a modular design, incorporating:
   * **Data Acquisition Module**: Collects data from sources such as web traffic logs, vulnerability databases, and user inputs.
   * **Feature Extraction Layer**: Analyzes URLs for attributes like domain age, length, entropy, and WHOIS data.
   * **Machine Learning Engine**: Trains and deploys models for URL classification and vulnerability prediction. This engine leverages both supervised learning techniques and ensemble methods for improved accuracy.
   * **Threat Intelligence Integration**: Incorporates external threat feeds to enhance detection capabilities and ensure the system remains updated on emerging threats.
   * **Reporting and Visualization Module**: Presents insights in a user-friendly manner, including actionable recommendations for remediation.
2. **Workflow**  
   The workflow begins with data ingestion, followed by preprocessing to clean and structure the data. The processed data is then fed into the machine learning engine for analysis. Results are passed to the visualization module for user interpretation and response.
3. **Security Considerations**  
   The system includes robust measures to ensure the integrity of the analysis process, such as secure data storage, encryption of sensitive information, and mechanisms to prevent model tampering or adversarial attacks.

## **5.2 User Interface**

The user interface (UI) is designed to be intuitive and accessible to users with varying levels of technical expertise. Key elements include:

1. **Dashboard**
   * A central hub displaying real-time analytics, system alerts, and summaries of detected threats.
   * Graphical representations, such as pie charts for URL classifications and line graphs tracking vulnerability trends over time.
2. **Interactive Analysis Panel**
   * Allows users to upload URLs or log files for immediate analysis.
   * Provides detailed reports, including detected threats, risk levels, and recommended actions.
3. **Customizable Settings**
   * Users can configure scanning parameters, integrate external threat intelligence feeds, and schedule automated scans.
4. **Reports and Logs**
   * Includes downloadable reports in formats such as PDF or CSV for compliance and audit purposes.
   * Logs of all past scans and analyses are accessible for review and traceability.

## **5.3 Interface Accessibility:**

The UI is designed with responsive design principles to ensure compatibility across devices, including desktops, tablets, and smartphones. A dark mode option is included for better usability in different lighting conditions.

Chapter 06

## **MANAGEMENT APPROACH**

This chapter focuses on the strategic management of the project, addressing aspects such as project planning, risk management, and other key elements necessary for ensuring successful project execution. The details of the management approach ensure that the project stays on track, meets its objectives, and handles potential challenges effectively.

## **6.1 Project Plan**

The **Project Plan** is the foundation for managing the project’s timeline, resources, and deliverables. It outlines the various tasks, milestones, and expected outcomes for each phase of the project.

**Table 6.1: Project plan**

|  |  |  |
| --- | --- | --- |
| Phase | Task Description | Timeline |
| Phase 1: Planning | Requirement analysis, feasibility study | Week 1 |
| Phase 2: Data Collection | Collect phishing and malware data, preprocess | Week 2–4 |
| Phase 3: Model Development | Train, validate, and optimize ML models | Week 5–7 |
| Phase 4: Integration | Develop the UI and integrate ML and tools | Week 8–9 |
| Phase 5: Testing | Evaluate system accuracy and functionality | Week 10–11 |
| Phase 6: Documentation | Prepare reports and presentations | Week 12 |

These milestones ensure that the project progresses in a structured manner, allowing for regular evaluations of the work.

## **6.2 Project Risks**

The **Project Risks** section is a critical component for anticipating potential issues that may arise during the project and preparing strategies to mitigate them. The identification of risks early in the project lifecycle allows the team to develop contingency plans and minimize the impact of unforeseen challenges.

1. **Overfitting:**

* The model might perform well on the training data but not on new data, which means it won't work properly in real life.
* **Solution:** We will use testing methods like cross-validation to make sure the model works well on new data.

1. **Data Privacy and Security:**

* The tool might handle sensitive user data, and there’s a risk it could be exposed or misused.
* **Solution:** We will make sure all data is protected, anonymized, and stored safely.

1. **User Adoption:**

* If the tool is difficult to use, people might not want to use it.
* **Solution**: We will design the tool to be user-friendly and test it with real users to make sure it’s easy to use.

## **REFERENCES**

* + **Anderson, R., & Agarwal, D.** **(2019).** *The Role of Social Engineering in Phishing Attacks*. Journal of Cybersecurity, 5(2), 134-145.
  + **Zetter, K. (2016).** *Inside the Hacking of the U.S. Election*. Wired. Retrieved from https://www.wired.com/story/inside-the-hacking-of-the-u-s-election/**:**
  + **Bhuyan, M. H., & Guha, S. (2021).** *Machine Learning for URL Phishing Detection: A Survey*. International Journal of Computer Applications, 177(14), 1-10.
  + **Zong, B., Zhang, C., & He, L. (2019)**. *A Survey on Machine Learning Methods for Phishing Detection*. Journal of Cybersecurity, 8(4), 1-10.
  + **Kaur, H., & Gupta, D. (2017**). *Comparative Study of Vulnerability Detection Techniques for Web Applications*. International Journal of Computer Applications, 164(1), 1-6.
  + **Raluca, C., & Doru, C. (2015).** *The Evolution of Malware Detection Techniques*. Proceedings of the International Conference on Security and Privacy in Computing and Communications, 184-191.