**REAL-TIME PHISHING DETECTION WITH MACHINE LEARNING INTEGRATION**

(PROJECT REPORT PHASE- I)

*submitted in partial fulfillment of the requirements*

*for the award of the degree in*

**BACHELOR OF TECHNOLOGY**

**in**

**COMPUTER SCIENCE AND ENGINEERING**

by

**Vetrivel M (211061101494)**

**Vishnu sah V T (211061101505)**

**Tarun Vikash B (211061101831)**



**DEPARTMENT**

**OF**

**COMPUTER SCIENCE AND ENGINEERING**

**NOVEMBER 2024**



**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**BONAFIDE CERTIFICATE**

This is to certify that this Project Report (Project Phase-I) is the bonafide work of

**Mr. VETRIVEL M** Reg. No **211061101494**, **Mr.** **VISHNU SAH VT** Reg. No **211061101505**, Mr. **TARUN VIKASH B** Reg. No **211061101831**, who carried out the project entitled “**REAL-TIME PHISHING DETECTION WITH MACHINE LEARNING INTEGRATION**” under our supervision from June 2024 to November 2024.

**Internal Guide Mrs.S.DIVYA**

Assistant Professor-CSE

Dr. M.G.R Educational and Research Institute

Deemed to be University

**Department Head Dr.S.GEETHA**

Professor and HoD of CSE

Dr. M.G.R Educational and Research Institute

Deemed to be University

**Project Coordinators**

**Dr.G.SONIYA PRIYATHARSINI**

Professor-CSE

**Mr.G.SENTHILVELAN**

Assistant Professor-CSE

Dr. M.G.R Educational and Research Institute

Deemed to be University

**Submitted for Viva Voce Examination held on \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Internal Examiner External Examiner**

**DECLARATION**

We, **VETRIVEL M (211061101494), VISHNU SAH V T(211061101504), TARUN VIKASH B (211061101831)**, hereby declare that the Project Report (Project Phase-I) entitled “**REAL-TIME PHISHING DETECTION WITH MACHINE LEARNING INTEGRATION**” is done by us under the guidance of **Mrs.S.DIVYA** is submitted in partial fulfillment of the requirements for the award of the degree in BACHELOR OF TECHNOLOGY in COMPUTER SCIENCE AND ENGINEERING.

1.

2.

3.

**DATE:**

**PLACE: CHENNAI SIGNATURE OF THE CANDIDATE(S)**

**ACKNOWLEDGEMENT**

We would first like to thank our beloved Chancellor   
**Thiru**.**Dr. A.C. SHANMUGAM, B.A., B.L.,** President **Er. A.C.S. Arunkumar, B.Tech., M.B.A.,** and Secretary **Thiru A. RAVIKUMAR** for all the encouragement and support extended to us during the tenure of this project and also our years of studies in this wonderful University.

We express my heartfelt thanks to our Vice Chancellor   
**Prof. Dr. S. GEETHALAKSHMI** in providing all the support of our Project (Project Phase-I).

We express my heartfelt thanks to our Head of the Department,   
**Prof. Dr. S.Geetha**, who has been actively involved and very influential from the start till the completion of our project.

Our sincere thanks to our Project Coordinators **Dr.G.SONIYA PRIYATHARSINI** and **Mr.G.SENTHILVELAN** and Project guide **Mrs.S.DIVYA** for their continuous guidance and encouragement throughout this work.

We would also like to thank all the teaching and non-teaching staffs of Computer Science and Engineering department, for their constant support and the encouragement given to us while we went about to achieving our project goals.

**TABLE OF CONTENTS**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO.** | **TITLE** | **PAGE NO.** |
|  | **LIST OF FIGURES** | vii |
|  | **LIST OF ABBREVIATIONS** | viii |
|  | **ABSTRACT** | ix |
| **1.** | **INTRODUCTION**  1.1. Introduction  **1.2. Problem Statement**  1.3. Objectives of the Work  1.4. Significance of the Work | 1  1  2  2  3 |
| **2.** | **LITERATURE SURVEY** | 5 |
| **3.** | **SYSTEM ANALYSIS**  **3.1. Existing Systems**  **3.2. Proposed System**  **3**.3. Functional Requirements  **3**.4. Non-Functional Requirements  **3**.5. Data Requirements  **3**.6. Technical Requirements  **3**.7. User Requirements | 7  7  8  9  9  10  11  12 |
| **4.** | **SYSTEM DESIGN**  4**.1. Data Flow in the System**  4**.2. Components of Architecture** | 13  13  14 |
| **5.** | **Implementation**  5.1. Environment Setup  5.2. Dataset Preparation  5.3. Model Development  5.4. Feature Extraction Module  5.5. Ensemble Prediction Module  5.6. API Development with Flask | 18  18  18  19  19  20  20 |
| **6.** | **Conclusion** | 22 |
| **7.** | **References** | 25 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO.** | **FIGURE NAME** | **PAGE NO.** |
| 1 | Actors involved in phishing | 4 |
| 4.1.1 | Steps involved in the system | 14 |
| 4.1.2 | Data flow in the system | 15 |
| 4.2.1 | Architecture | 19 |
| 4.2.2 | Architecture continuation | 20 |
| 5 | Life cycle of Phishing | 24 |

**LIST OF ABBREVIATIONS**

|  |  |
| --- | --- |
| **UI** | User Interface |
| **HTML** | Hyper Text Markup Language |
| **CSS** | Cascading Style Sheets |
| **AI** | Artificial Intelligence |
| **ML** | Machine Learning |
| **URL** | Unified Resource Locator |
| **REST** | REpresentational State Transfer |
| **API** | Application Programming Interface |
| **JSON** | JavaScript Object Notation |
| **RBAC** | Role-Based Access Control |

**ABSTRACT**

This work presents a real-time phishing detection system that integrates machine learning models to effectively identify and prevent phishing attacks, spam, and defaced websites. The system is designed to analyze various key URL features such as length, domain, subdomain, numeric characters, special characters, and the presence of HTTPS, providing a comprehensive approach to phishing detection. Utilizing machine learning algorithms, including Decision Tree, Random Forest, and Extra Tree classifiers, the system ensures high accuracy in distinguishing between legitimate and malicious websites. Additionally, the system incorporates behavioral analysis and real-time monitoring to enhance detection capabilities, making it adaptable to evolving phishing tactics. By offering immediate incident responses, the system strengthens user protection against cyber threats, particularly in web environments.

**Keywords:** Artificial Intelligence (AI), cybersecurity, Threat Detection, Threat Response, Machine Learning, Paradigm Shift, Evolution

|  |  |  |  |
| --- | --- | --- | --- |
| Student Group | VISHNU SAH V T  (211061101505) | VETRIVEL M  (211061101494) | TARUN VIKASH B  (211061101831) |
| Project Title | REAL-TIME PHISHING DETECTION WITH MACHINE LEARNING INTEGRATION | | |
| Program Concentration  Area | Phishing, Spamming, hacking | | |
| Constraints Example | Power Constraints | | |
| Economic | No | | |
| Environmental | No | | |
| Sustainability | Yes | | |
| Implementable | Yes | | |
| Ethical | Yes | | |
| Health and Safety | No, yes | | |
| Social | Yes | | |
| Political | No | | |
| Other | Machine Learning Integration | | |
| Standards |  | | |
| 1 | PEP 8 | | |
| 2 | ISO 50001, ISO 27001, ISO 12207, **ISO 27001** | | |
| 3 | USB to TTL | | |
| Prerequisite Courses for the Major Design  Experiences | 1. Networking 2. Machine Learning 3. Cyber Security | | |

**CHAPTER-1**

**INTRODUCTION**

* 1. **Introduction:**

In today’s interconnected world, the internet has become an integral part of daily life for billions of people. As more activities shift online, from personal communications to banking and e-commerce, the risk of cyber threats continues to rise. Among these, phishing attacks have emerged as one of the most common and dangerous forms of cybercrime.

Phishing refers to the fraudulent practice of sending fake messages or creating deceptive websites that mimic legitimate ones to lure users into disclosing sensitive information, such as usernames, passwords, and credit card details. These attacks not only compromise individuals' privacy and finances but also threaten the security of organizations and societies at large.

The sophistication and frequency of phishing attacks have increased exponentially in recent years, driven by the widespread adoption of digital platforms and the availability of phishing toolkits. Traditional security mechanisms such as firewalls and antivirus software, while essential, are often inadequate against phishing.

Unlike viruses or malware, phishing primarily exploits human trust and can evade technical defenses by using socially engineered tactics. Therefore, there is a critical need for proactive, intelligent systems that can identify and block phishing attempts before they reach unsuspecting users.

To address this need, researchers and cybersecurity professionals have increasingly turned to Machine Learning (ML) as a potential solution. ML models can analyze large datasets, recognize patterns, and make predictions based on empirical data. In the context of phishing detection, ML models can be trained to differentiate between legitimate and malicious websites or emails based on features such as URL structure, domain characteristics, and the presence of certain keywords or characters. This approach enables the development of a Real-Time Phishing Detection System that can identify phishing attempts with high accuracy, often before any harm is done.

**1.2. Problem Statement**

Phishing attacks pose a significant threat to the security of individuals and organizations, leading to data breaches, financial losses, and compromised systems. Existing phishing detection solutions are often limited by their reliance on static lists of known phishing URLs or simplistic heuristic rules. As a result, they struggle to identify new and evolving phishing techniques in real time. This lag in detection allows attackers to exploit users, creating substantial cybersecurity risks. Given the dynamic nature of phishing attacks, there is a need for a more robust and adaptive solution that can detect phishing attempts as they occur.

A system that integrates machine learning to analyze URL characteristics and user behavior can overcome the limitations of traditional methods. Such a system should be capable of identifying novel phishing patterns, providing immediate responses to potential threats, and adapting to the continuously changing landscape of phishing tactics. By incorporating advanced machine learning models and real-time monitoring capabilities, the system will provide a scalable, efficient, and user-friendly defense mechanism, offering users enhanced protection against phishing attacks.

**1.3. Objectives of the Work**

The primary objective of this work is to develop a Real-Time Phishing Detection System that leverages Machine Learning to accurately classify URLs as either "Phishing" or "Benign.".

The system is designed to offer a reliable, proactive solution capable of identifying phishing attempts based on the characteristics of a given URL, without reliance on outdated blacklists or predefined rules. Specific objectives include:

**Real-Time Detection**:

The system must be able to process and classify URLs in real-time to provide immediate feedback to users.

**Feature Engineering**:

Extract relevant features from URLs to improve model accuracy, such as URL length, domain structure, presence of special characters, and subdomain characteristics.

**Machine Learning Integration**:

Utilize multiple Machine Learning algorithms, including Decision Tree, Random Forest, and Extra Trees classifiers, to enhance detection accuracy and robustness.

**Ensemble Decision Making**:

Implement an ensemble approach that aggregates predictions from multiple models to achieve a more reliable final decision.

**API Development**:

Develop a RESTful API using Flask to enable integration with other applications or platforms for real-time phishing detection.

**User Confidence and Interpretability**:

Provide a confidence score with each prediction to help users understand the level of certainty in the classification results. Through these objectives, this work aims to build a practical, deployable phishing detection solution that addresses the limitations of existing systems and empowers users with reliable protection against phishing.

**1.4. Significance of the Work**

The Real-Time Phishing Detection System with Machine Learning integration represents a significant advancement in cybersecurity. Traditional phishing detection techniques, such as blacklist-based approaches, cannot keep up with the rapidly changing landscape of phishing threats.

Cybercriminals frequently modify URLs, use temporary domains, and exploit new tactics to bypass these basic security measures. Machine Learning, however, offers a promising alternative by enabling the system to learn from data and adapt to new patterns without extensive manual intervention.

Moreover, the real-time nature of this system provides immediate benefits for both individual users and organizations. For individuals, it enhances protection against identity theft and financial fraud.

For organizations, it reduces the risk of data breaches, preserves reputation, and protects customer trust. Financial institutions, e-commerce platforms, and other sectors that handle sensitive data can particularly benefit from deploying such a system, as they are prime targets for phishing attacks.

From a technical standpoint, this work contributes to the development of intelligent phishing detection models that are both scalable and deployable. By incorporating an ensemble approach that aggregates predictions from multiple ML models, the system achieves a high level of accuracy and robustness.

The Flask API makes it easy to integrate the system with various user interfaces, from browser extensions to mobile applications, enhancing accessibility and practical applicability.

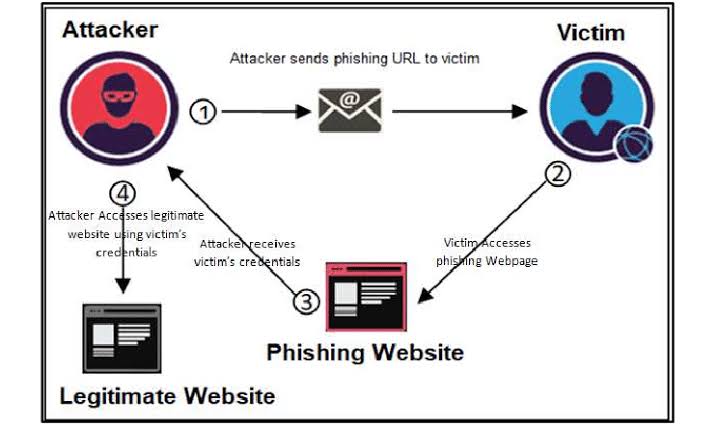


Figure 1 - Actors involved in phishing

**CHAPTER-2**

**LITERATURE SURVEY**

M. F. Ansari, P. K. Sharma, and B. Dash, “Prevention of phishing attacks using AI-based Cybersecurity Awareness Training,”

Ansari, Sharma, and Dash (2022), in their paper *“Prevention of Phishing Attacks Using AI-based Cybersecurity Awareness Training”*, examine the role of artificial intelligence in enhancing cybersecurity awareness to combat phishing threats. Published in the *International Journal of Smart Sensor and Adhoc Network*, this study discusses how AI-driven training programs can improve users' ability to recognize and respond to phishing attempts. By simulating phishing scenarios and providing targeted feedback, the AI-based training model helps users develop critical skills to identify phishing attacks, thus contributing to a more proactive and informed approach to cybersecurity.

W. Ali, “Phishing website detection based on supervised machine learning with wrapper features selection,”

Ali, explores the use of supervised machine learning models combined with wrapper-based feature selection to improve phishing website detection. Published in the *International Journal of Advanced Computer Science and Applications*, this study focuses on selecting the most relevant features from URLs, such as domain-specific characteristics, to enhance the accuracy and efficiency of machine learning classifiers in distinguishing phishing websites from legitimate ones. The findings demonstrate that applying wrapper feature selection methods can significantly improve model performance, making this approach effective for real-time phishing detection.

R. Alabdan, “future internet Phishing Attacks Survey: Types, Vectors, and Technical Approaches”, doi: 10.3390/fi12100168.

Alabdan (2020), in the paper *“Phishing Attacks Survey: Types, Vectors, and Technical Approaches”*, provides a comprehensive survey on the various forms of phishing attacks, their delivery methods, and the technical strategies used to combat them. Published in *Future Internet*, this paper categorizes phishing attacks based on their types and vectors, offering insights into the tactics used by attackers. Additionally, it examines the technical approaches employed in phishing detection and prevention, including advancements in AI and machine learning. Alabdan’s work serves as a valuable resource for understanding the evolving nature of phishing threats and the effectiveness of modern defense mechanisms.

R. S. Rao, Alwyn, and R. Pais, “Detection of phishing websites using an efficient feature-based machine learning framework,” Neural Computing and Applications, vol. 31, doi: 10.1007/s00521-017-3305 -0.

Rao, Alwyn, and Pais (2017), in their paper *“Detection of Phishing Websites Using an Efficient Feature-Based Machine Learning Framework”*, present a machine learning approach for detecting phishing websites based on carefully selected features. Published in *Neural Computing and Applications*, this study proposes an efficient framework that leverages URL-specific attributes—such as URL length, domain age, and presence of special characters—to improve classification accuracy. By utilizing feature-based techniques, the framework enhances the model’s ability to distinguish phishing websites from legitimate ones effectively. This work highlights the importance of feature selection in improving both the performance and computational efficiency of phishing detection systems.

A. Kumar Jain and B. B. Gupta, “Towards detection of phishing websites on client-side using machine learning based approach,” vol. 68, pp. 687–700, 2018, doi: 10.1007/s11235-017-0414-0.

Jain and Gupta (2018), in their paper *“Towards Detection of Phishing Websites on Client-Side Using Machine Learning Based Approach”*, explore a client-side solution for identifying phishing websites using machine learning techniques. Published in *Telecommunication Systems*, this study focuses on applying machine learning algorithms directly on the client-side, allowing for real-time detection without relying on server-based solutions. By analyzing features extracted from URLs, such as domain information and security indicators, the proposed approach enhances client-side security and protects users from phishing attacks at the point of interaction. This work contributes to the development of efficient, user-focused phishing detection tools that operate locally to ensure immediate protection.

**CHAPTER-3**

**SYSTEM ANALYSIS**

In order to develop a robust and effective real-time phishing detection system, a comprehensive analysis of the system requirements is essential. This phase ensures that all functional and non-functional aspects of the system are well-defined and understood, guaranteeing alignment with user needs and technical specifications.

**3.1. Existing Systems**

Several existing systems are designed to detect phishing websites and prevent users from falling victim to such attacks. These systems employ a variety of techniques, such as blacklists, heuristic-based detection, and machine learning. Some well-known existing systems include:

**Google Safe Browsing**:

Integrated into browsers like Chrome and Firefox, Google Safe Browsing uses an extensive blacklist of known phishing and malware websites. It warns users when they attempt to navigate to a malicious website. However, it primarily relies on previously identified phishing URLs, making it slower to respond to new phishing attempts.

**PhishTank**:

A community-driven system that allows users to submit suspected phishing URLs. These URLs are verified by the community and made available through an API. While effective for gathering phishing URLs, it is reliant on human verification and lacks real-time detection.

**Microsoft Defender SmartScreen**:

Built into Windows and Microsoft Edge, this system uses heuristic-based detection to warn users about suspicious websites. However, it can sometimes flag legitimate websites as phishing, resulting in a higher false positive rate.

**OpenPhish**:

Provides real-time phishing detection feeds, leveraging machine learning algorithms to detect phishing attacks. While it offers more advanced detection mechanisms, OpenPhish’s full services are primarily subscription-based, limiting access for smaller developers or organizations.

**3.2. Proposed System**

The proposed system improves upon existing solutions by integrating machine learning models and behavioral analysis for real-time phishing detection. It leverages the strengths of multiple machine learning classifiers to ensure high accuracy in detecting both known and unknown phishing threats.

**Machine Learning Integration**:

Unlike traditional blacklist or heuristic-based systems, the proposed system uses advanced machine learning models to classify URLs based on features like URL length, domain structure, and special characters.This enables more adaptive detection of phishing websites, especially newly created or dynamically generated phishing URLs.

**Real-Time URL Analysis**:

The system is designed to provide real-time analysis of URLs, ensuring immediate detection and feedback to users. This prevents delays in identifying phishing websites, reducing the likelihood of successful phishing attacks.

**Automated Response System**:

The proposed system will also provide an automated response mechanism to alert users or block access to malicious websites. This real-time feedback minimizes user exposure to potential threats.

**Continuous Learning**:

The system incorporates continuous learning, allowing it to adapt to new phishing techniques. The models are periodically retrained using newly gathered data to maintain accuracy as phishing tactics evolve. The requirement analysis for this work is divided into several key categories: **functional** requirements, **non-functional** requirements, **data** requirements, and **technical** requirements.

**3.3. Functional Requirements**

The functional requirements define the specific tasks that the phishing detection system must perform. These tasks directly contribute to the system’s core functionality, ensuring it meets its intended purpose.

**URL Scanning and Feature Extraction:**

The system must be able to scan URLs and extract relevant features such as the length of the URL, domain structure, subdomain usage, the presence of numeric characters, special characters, and whether the URL uses HTTPS.

**Machine Learning Integration:**

The system must utilize machine learning models (e.g., Decision Tree, Random Forest, Extra Trees) to classify URLs as either phishing or legitimate based on the extracted features.

**Real-Time Detection:**

The system must process and classify URLs in real-time, providing immediate feedback to users on the legitimacy of the URL.

**Automated Response:**

Upon detecting a phishing URL, the system should automatically provide alerts to the user, potentially preventing access to malicious websites.

**Continuous Learning:**

The system should be designed to adapt to new phishing patterns through retraining or online learning, allowing it to stay effective as new phishing techniques evolve.

**3.4. Non-Functional Requirements**

Non-functional requirements describe the attributes the system must satisfy beyond its basic functionality. These attributes ensure the system operates effectively under real-world conditions.

**Performance:**

The system must be capable of analyzing and classifying URLs with minimal latency to ensure real-time detection. The response time should not exceed a few milliseconds per URL to prevent delays in user activity.

**Scalability:**

The system must be scalable to handle an increasing number of URLs as well as a larger volume of users. This ensures that the detection service can expand without degrading performance.

**Security:**

The system must ensure that the user’s interaction with the system, such as URL submissions and responses, is handled securely. Data exchanges must be encrypted, especially when dealing with sensitive information.

**Accuracy:**

The machine learning models must be trained to maintain a high level of accuracy, minimizing both false positives (legitimate URLs flagged as phishing) and false negatives (phishing URLs not detected).

**Usability:**

The system must be user-friendly, allowing users with minimal technical knowledge to easily interact with the URL checking tool. Feedback and alerts should be clear and concise, enabling users to understand the results of the phishing detection process.

**Availability:**

The system must be available for use 24/7 to ensure that users can continuously rely on it for protection against phishing attacks.

**3.5. Data Requirements**

The effectiveness of the phishing detection system relies heavily on the quality and variety of the data it processes. The system must have access to various data sources to accurately detect phishing URLs.

**URL Data:**

The system must be capable of analyzing various types of URLs, including those from legitimate websites and phishing websites. The data should include diverse patterns of phishing URLs to ensure that the model can generalize across different forms of phishing attacks.

**Feature Set:**

The key features extracted from the URLs should include:

* + - URL length
    - Domain name and subdomain structure
    - The presence of numeric characters and special characters
    - The use of HTTPS
    - Query parameters and path complexity

**Training Data:**

A substantial and balanced dataset of both phishing and legitimate URLs must be used to train the machine learning models. The dataset should be periodically updated to include new phishing patterns and website structures.

**3.6. Technical Requirements**

The technical requirements outline the necessary tools, frameworks, and infrastructure needed to develop, deploy, and maintain the phishing detection system.

**Machine Learning Frameworks:**

The system will use machine learning libraries such as **scikit-learn** for model training and prediction. Additionally, **joblib** will be used to serialize and load the trained models efficiently in a production environment.

**Web Framework:**

The system will be deployed as a web application using **Flask**, a lightweight and flexible web framework. Flask will handle user requests, including URL submissions, and return results after running the URL through the trained models.

**API Development**:

A RESTful API will be developed to allow third-party applications or external systems to access the phishing detection service. This API will take URLs as input and return phishing detection results in real-time.

**Database Management:**

The system will require a database to store URL records, user interactions, and detected phishing URLs. A lightweight and scalable database such as **SQLite** or **PostgreSQL** will be used.

**Security Measures:**

Proper security measures such as **HTTPS** for data transmission, secure API keys, and role-based access control (**RBAC**) will be implemented to protect the system from potential threats.

**3.7. User Requirements**

The system must be designed with the user’s expectations and ease of use in mind. It should be intuitive and accessible to non-technical users who need real-time protection from phishing attacks.

**Instant Feedback:**

Users should receive immediate results when submitting a URL, indicating whether the URL is safe or potentially malicious.

**Clear Notifications:**

If a phishing URL is detected, the system should provide clear notifications to the user, detailing the risks and recommending actions (e.g., avoiding the link or blocking access).

**Mobile Compatibility:**

The system must be accessible from both desktop and mobile devices to provide flexibility for users on different platforms. This requirement analysis serves as the foundation for the design and development of the phishing detection system. By clearly defining the functional, non-functional, data, and technical requirements, the system will meet its objectives effectively, providing users with a powerful tool to combat phishing threats in real-time.

**CHAPTER-4**

**SYSTEM DESIGN**

The system design outlines the architecture and components of the real-time phishing detection system. The system consists of several key modules, each responsible for specific tasks, including data processing, feature extraction, machine learning classification, and real-time URL analysis. By designing a modular architecture, the system is structured to perform real-time phishing detection with high accuracy and scalability.

**4.1. Data Flow in the System**

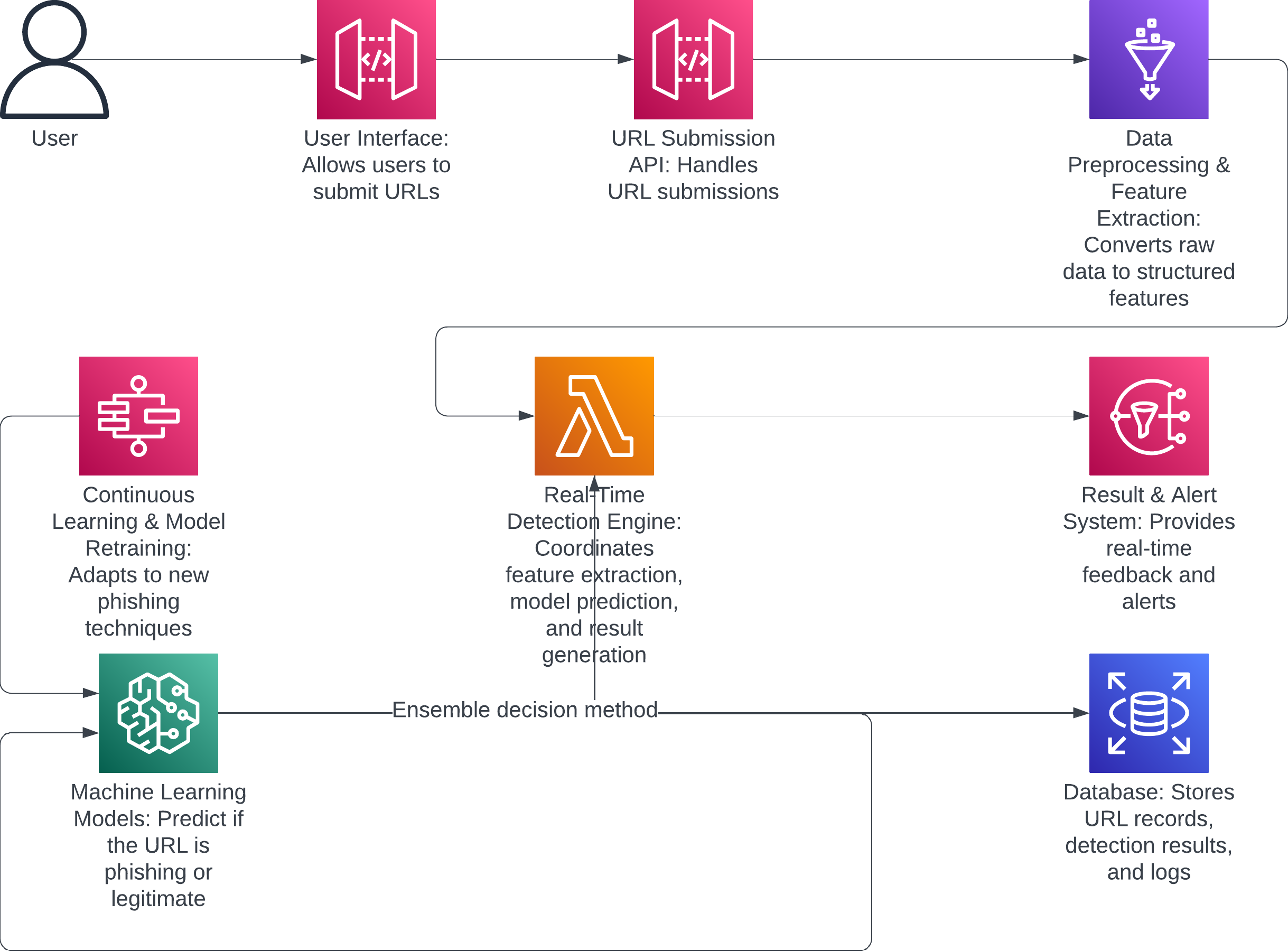
****

FIGURE 4.1.1-STEPS INVOLVED IN THE SYSTEM

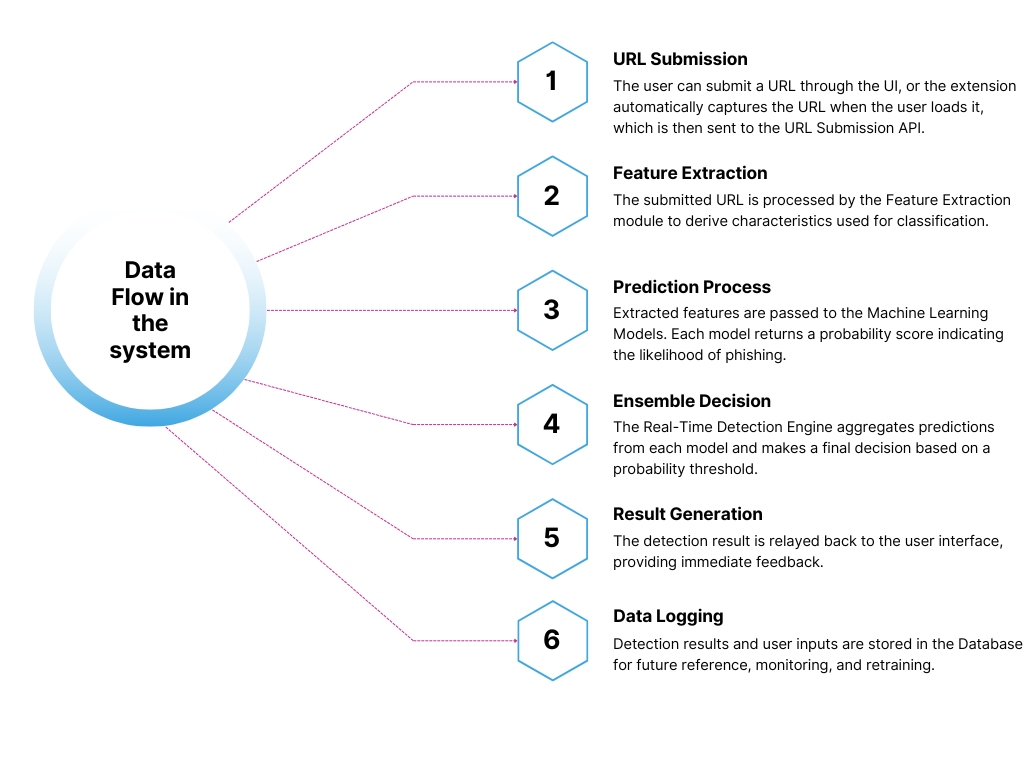


FIGURE 4.1.2 - DATA FLOW

**4.2. Components of Architecture**

**User Interface (UI)**

The front-end interface that allows users to submit URLs for phishing detection. Provides users with a simple and intuitive way to interact with the system, displaying results in real time.

**URL Submission API**

A RESTful API built with Flask that handles URL submissions from users. Receives URLs from the UI, forwards them to the feature extraction module, and returns detection results.

**Data Preprocessing and Feature Extraction**

The component responsible for cleaning and extracting features from the submitted URLs.

**Features Extracted**:

URL length, domain structure, subdomain count, numeric characters, special characters, HTTPS usage, and query parameters.Converts raw URL data into structured features for input to the machine learning models.

**Machine Learning Models**

A set of pre-trained machine learning models, including Decision Tree, Random Forest, and Extra Trees classifiers. Each model analyzes the extracted features and predicts whether the URL is phishing or legitimate.

**Ensemble Decision**:

The system combines results from each model (ensemble method) to determine a final prediction based on an average probability.

**Real-Time Detection Engine**

The core processing unit that coordinates feature extraction, model prediction, and result generation in real time. Ensures fast and efficient processing, generating immediate feedback to the user on URL legitimacy.

**Database**

A lightweight database (e.g., SQLite or PostgreSQL) for storing URL records, detection results, and model performance logs. Keeps records of analyzed URLs, which are useful for retraining models and improving detection accuracy over time.

**Continuous Learning and Model Retraining**

A pipeline for updating and retraining the machine learning models with newly gathered data. Enables the system to adapt to new phishing techniques by periodically retraining models with recent data.

**Result and Alert System**

Provides real-time feedback and alerts to users on the status of submitted URLs. Sends a clear phishing alert if the URL is flagged as malicious or a confirmation if it’s legitimate, enhancing user security.

**Cloud Deployment**

Cloud infrastructure (e.g., AWS, Google Cloud) for scalability and availability. Ensures the system can handle high traffic and large volumes of URL requests, supporting continuous access for multiple users.

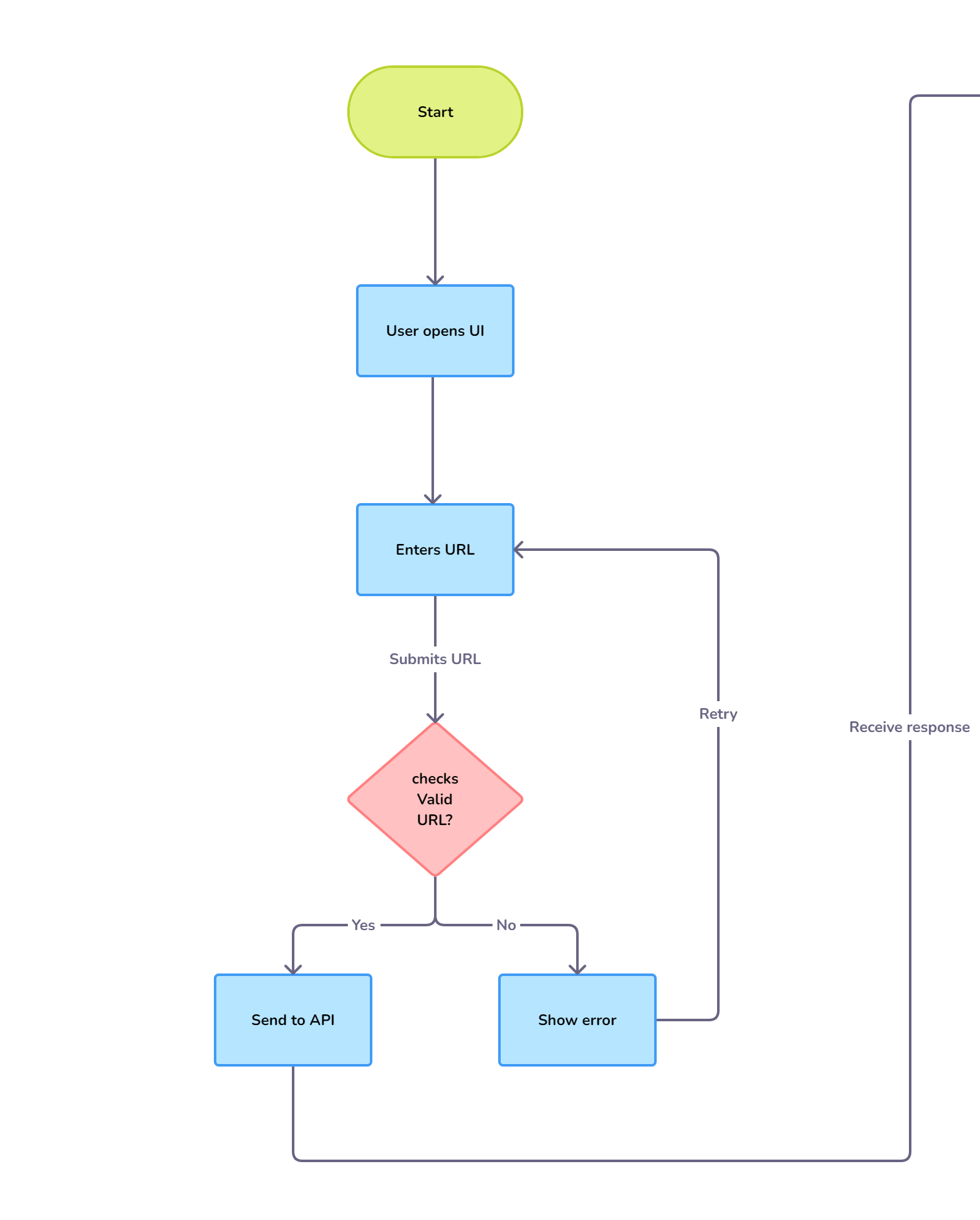


FIGURE 4.2.1 - ARCHITECTURE

A diagram of a flowchart

Description automatically generated

FIGURE 4.2.2 – ARCHITECTURE continuation

**CHAPTER-5**

**IMPLEMENTATION**

This chapter provides an in-depth overview of each module within the Real-Time Phishing Detection System, covering the coding frameworks, machine learning models, and methodologies applied throughout development.

**5.1. Environment Setup**

**Programming Language:**

The system was developed using Python due to its extensive libraries and support for machine learning.

**Frameworks and Libraries:**

The following libraries were used: **Flask:** For building the RESTful API to interact with the phishing detection model. **scikit-learn:** For implementing machine learning algorithms. **joblib:** For efficient model serialization. **NumPy and Pandas:** For data handling and manipulation.

**Additional Tools:**

**tldextract:** To parse and extract domain details from URLs. **re:** For regular expressions in feature extraction. **seaborn and matplotlib:** Used for visualizations during model evaluation to interpret data and performance.

**5.2. Dataset Preparation**

**Dataset Source:**

The dataset for training and testing was sourced from public databases such as [source, e.g., Kaggle or PhishTank].

**Data Cleaning:**

Preprocessing steps included handling missing values, removing duplicates, and correcting any anomalies to improve data quality.

**Feature Engineering:**

Features extracted from URLs included: URL length, Domain and subdomain characteristics, Presence of special characters (e.g., "-", "@"), Numerical and alphabetical patterns

**Train-Test Split:**

The data was divided into training and testing sets with a typical 80/20 split, ensuring a balanced and representative dataset for model training and evaluation.

**5.3. Model Development**

**Models Used:** Decision Tree Classifier**,** Random Forest Classifier**,** Extra Trees Classifier

**Model Selection:**

These models were selected based on factors such as interpretability, speed, and accuracy, making them suitable for real-time detection.

**Training Process:**

Each model was trained on the engineered URL features, and performance metrics (accuracy, precision, recall) were evaluated to select optimal parameters.

**Serialization:**

Trained models were saved as `.pkl` files using `joblib`, enabling seamless integration with the Flask API for real-time inference.

**5.4. Feature Extraction Module**

**Objective:**

To extract relevant characteristics from each URL, providing structured input features for the machine learning models.

**Key Features Extracted:** URL length, domain length**,** Number of certain characters (e.g., "-", "@", "//")**,** Domain suffix and the number of digits

**Implementation:**

The `extract\_features\_single(url)` function was developed to process URLs, extracting features and returning them as a vector for model input.

**5.5. Ensemble Prediction Module**

**Ensemble Decision Process:**

Combines predictions from the Decision Tree, Random Forest, and Extra Trees classifiers. Each model returns a probability score for the URL being phishing, and the average score is calculated. If the average probability exceeds a threshold (e.g., 0.4 for heightened sensitivity), the URL is classified as "Phishing."

**Confidence Score:**

This average score acts as a confidence level, providing users with an additional measure of certainty in the prediction.

**5.6. API Development with Flask**

**API Overview:**

A RESTful API was developed using Flask to allow seamless access to phishing detection functionality.

**Endpoints:**

**`/predict`:**

Accepts a URL input and returns a classification result (Phishing or Benign) along with a confidence score.

**Integration:**

Each request triggers the `ensemble\_classify` function, which loads models, extracts features, and delivers a final classification.

**Error Handling:**

Implemented mechanisms to manage invalid URLs and unexpected inputs, ensuring a robust and user-friendly API experience.

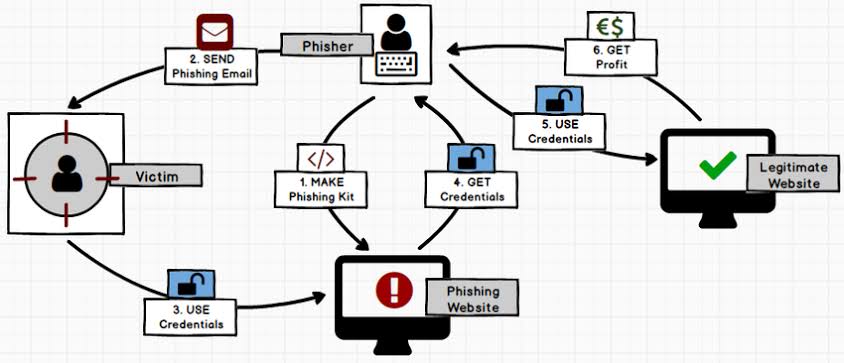


FIGURE 5 - LIFE CYCLE OF PHISHING

**CHAPTER-6**

**CONCLUSION**

The Real-Time Phishing Detection System developed in this work offers a sophisticated approach to mitigating the risks associated with phishing attacks. By leveraging machine learning, specifically an ensemble of Decision Tree, Random Forest, and Extra Trees classifiers, the system efficiently identifies and classifies URLs as either "Phishing" or "Benign."

This classification process relies on the extraction of specific URL features, such as length, structure, and the presence of special characters or digits, which are commonly associated with phishing attempts. The system’s design and implementation aim to provide a solution that is both accurate and user-friendly, allowing it to serve as a practical tool for individuals, organizations, and web service providers seeking to enhance their cybersecurity measures.

The Flask API-based architecture further contributes to the system’s accessibility by allowing easy integration into a variety of applications, such as web platforms, browser extensions, or mobile apps. This ensures that users can receive immediate feedback on the legitimacy of URLs they encounter in real-time, reducing the risk of falling victim to phishing scams.

**Outcomes and Achievements**

**High Detection Accuracy and Reliability:**

Through an ensemble approach combining multiple machine learning models, this system achieves high accuracy rates in detecting phishing URLs. The use of diverse classifiers improves reliability by reducing the likelihood of misclassification, ensuring that most phishing URLs are accurately flagged.

**Real-Time Processing Capabilities:**

The system has been optimized for real-time performance, allowing for rapid prediction by using a streamlined feature extraction process. This makes the system suitable for real-world applications where immediate detection is crucial to prevent phishing attacks.

**User-Friendly Integration and Scalability:**

The implementation of the system as a Flask API allows it to be easily integrated into a range of client applications. This design choice not only enhances accessibility but also enables scaling up to meet the demands of larger user bases or enterprise environments with minimal adjustments.

**Future Enhancements**

While the current system performs well in terms of detection accuracy and ease of integration, there are several areas where it could be enhanced to ensure greater robustness, adaptability, and user protection. These potential future improvements include:

**Behavioral Analysis for Enhanced Detection**:

Integrating behavioral analysis, such as tracking suspicious redirects, examining page content, or monitoring server response behaviors, could provide additional context that enhances detection accuracy. Behavioral patterns of phishing websites often differ significantly from legitimate sites, and capturing these can further reduce false positives and false negatives.

**Continuous Learning and Model Adaptation:**

As phishing tactics evolve, so must detection systems. A potential enhancement would be to incorporate continuous learning mechanisms, allowing the system to adapt to new phishing patterns by periodically retraining models with fresh data. This could be accomplished by integrating with a dynamic dataset that updates regularly with recent phishing URLs.

**Deployment on Scalable Cloud Infrastructure**:

To meet the needs of larger user bases and enterprise applications, deploying the system on a cloud infrastructure, such as AWS or Google Cloud, would enhance scalability. Cloud deployment offers additional benefits, including automated load balancing, scalability to handle high traffic, and ease of maintenance. Additionally, cloud-based deployment can support serverless architectures, which could further optimize costs and performance for large-scale operations.

**Enhanced Security and Privacy Measures:**

Since the system processes URLs that could potentially contain sensitive information, implementing strong data privacy practices, such as encryption and anonymization, is critical. Protecting user data will increase trust in the system, making it more attractive for widespread adoption in security-conscious environments.

**Final Reflections**

The Real-Time Phishing Detection System presented in this work showcases the effectiveness of machine learning in addressing the persistent challenge of phishing attacks. By building a system that is not only accurate but also responsive and accessible, we have contributed a practical tool for improving cybersecurity.

This tool offers significant benefits for a wide range of users, from individual internet users looking for safer browsing to enterprises and web service providers aiming to protect their customers. The integration of machine learning in cybersecurity highlights the vast potential of technology to combat online threats in a proactive manner. As cyber threats evolve, the need for adaptive, intelligent, and scalable security solutions will only grow.

The insights gained from this work provide a foundation for future advancements, laying the groundwork for enhanced protection against phishing and potentially other forms of cyber-attacks. This work has the potential to evolve further, integrating cutting-edge technologies and adapting to new threat landscapes. With continuous improvements, the Real-Time Phishing Detection System could become a comprehensive cybersecurity solution that not only detects phishing attacks but also actively contributes to a safer internet environment for all users.

**CHAPTER-7**

**REFERENCE**

[1] Vegesna, V.V., (2023). Enhancing Cyber Resilience by Integrating AI-Driven Threat Detection and Mitigation Strategies. Transactions on Latest Trends in Artificial Intelligence, 4(4).

[2] Bonfanti, M.E., (2022). Artificial intelligence and the offence-defence balance in cyber security. Cyber Security: Socio-Technological Uncertainty and Political Fragmentation. London: Routledge, pp.64-79.

[3] Vegesna, V.V., (2023). Comprehensive Analysis of AI-Enhanced Defense Systems in Cyberspace. International Numeric Journal of Machine Learning and Robots, 7(7).

[4] Guembe, B., Azeta, A., Misra, S., Osamor, V.C., Fernandez-Sanz, L. and Pospelova, V., (2022). The Emerging Threat of Ai-driven Cyber Attacks: A.

[5] Zeadally, S., Adi, E., Baig, Z. and Khan, I.A., (2020). Harnessing artificial intelligence capabilities to improve cybersecurity. Ieee Access, 8, pp.23817-23837.

[6] Sjöblom, C., (2021). Artificial Intelligence in Cybersecurity and Network security.

[7] Kaloudi, N. and Li, J., (2020). The ai-based cyber threat landscape: A survey. ACM Computing Surveys (CSUR), 53(1), pp.1-34.

[8] Guembe, B., Azeta, A., Misra, S., Osamor, V.C., Fernandez-Sanz, L. and Pospelova, V., (2022). The Emerging Threat of Ai-driven Cyber Attacks: A.

[9] Bokhari, S.A.A. and Myeong, S., (2023). The influence of artificial intelligence on e-Governance and cybersecurity in smart cities: A stakeholder’s perspective. IEEE Access.

[10] Kumar, S., Gupta, U., Singh, A.K. and Singh, A.K., (2023). Artificial intelligence: revolutionizing cyber security in the digital era. Journal of Computers, Mechanical and Management, 2(3), pp.31-42.

[11] Zeadally, S., Adi, E., Baig, Z. and Khan, I.A., (2020). Harnessing artificial intelligence capabilities to improve cybersecurity. Ieee Access, 8, pp.23817-23837.

[12] Kaloudi, N. and Li, J., (2020). The ai-based cyber threat landscape: A survey. ACM Computing Surveys (CSUR), 53(1), pp.1-34.

[13] Rangaraju, S., (2023). AI SENTRY: REINVENTING CYBERSECURITY THROUGH INTELLIGENT THREAT DETECTION. EPH-International Journal of Science And Engineering, 9(3), pp.30-35.

[14] Soni, V.D., (2020). Challenges and Solution for Artificial Intelligence in Cybersecurity of the USA. Available at SSRN 3624487.

[15] Markevych, M. and Dawson, M., (2023). A review of enhancing intrusion detection systems for cybersecurity using artificial intelligence (ai). In International conference KNOWLEDGE-BASED ORGANIZATION (Vol. 29, No. 3, pp. 30-37).

[16] Jian, Y.L. and Luaus, C., (2023). Enhancing Power Grid Security: A Comprehensive Study on Cybersecurity Measures and Fault Diagnosis Strategies Amid Dynamic System Variations. Revista Espanola de Documentacion Cientifica, 17(2).

[17] Padilla-Vega, R., Sanchez-Rivero, C. and Ojeda-Castro, A., (2023). Navigating the business landscape: challenges and opportunities of implementing artificial intelligence in cybersecurity governance. Issues in Information Systems, 24(4).

[18] Almeida, G. and Vasconcelos, F., (2023). Self-Healing Networks: Adaptive Responses to Ransomware Attacks.

[19] BOTEZATU, U.E., (2023). AI-Centric secure outer space operations. BULLETIN OF" CAROL I" NATIONAL DEFENCE UNIVERSITY, 12(3), pp.205-221.

[20] Jabbarova, K., (2023). AI AND CYBERSECURITY-NEW THREATS AND OPPORTUNITIES. Journal of Research Administration, 5(2), pp.5955-5966.

[21] Nobles, C. and Mcandrew, I., (2023). The Intersectionality of Offensive Cybersecurity and Human Factors: A Position Paper. Scientific Bulletin, 28(2), pp.215-233.

[22] Chandana, P. and Gulzar, C.M., (2023). Securing Cyberspace: A Comprehensive Journey through AI's Impact on Cyber Security. Tuijin Jishu/Journal of Propulsion Technology, 44(2).

[23] Iqbal, S., Rizvi, S.W.A., Haider, M.H. and Raza, S., (2023). Artificial Intelligence in Security and Defense: Explore the integration of AI in military strategies, security policies, and its implications for global power dynamics. INTERNATIONAL JOURNAL OF HUMAN AND SOCIETY, 3(4), pp.341 353.

[24] Mohsin, A., Janicke, H., Nepal, S. and Holmes, D., (2023). Digital Twins and the Future of their Use Enabling Shift Left and Shift Right Cybersecurity Operations. arXiv preprint arXiv:2309.13612.

[25] Fujima, H., Kumamoto, T. and Yoshida, Y., (2023). Using chatgpt to analyze ransomware messages and to predict ransomware threats.

[26] Ghasemshirazi, S., Shirvani, G. and Alipour, M.A., (2023). Zero Trust: Applications, Challenges, and Opportunities. arXiv preprint arXiv:2309.03582.

[27] Vasconcelos, F.E. and Almeida, G.S., 2023. LLaMa Assisted Reverse Engineering of Modern Ransomware: A Comparative Analysis with Early Crypto-Ransomware.

[28] Steingartner, W., Galinec, D. and Kozina, A., (2021). Threat defense: Cyber deception approach and education for resilience in hybrid threats model. Symmetry, 13(4), p.597.

[29] Bokhari, S.A.A. and Myeong, S., (2023). The influence of artificial intelligence on e-Governance and cybersecurity in smart cities: A stakeholder’s perspective. IEEE Access.

[30] Srivastava, G., Jhaveri, R.H., Bhattacharya, S., Pandya, S., Maddikunta, P.K.R., Yenduri, G., Hall, J.G., Alazab, M. and Gadekallu, T.R., (2022). XAI for cybersecurity: state of the art, challenges, open issues and future directions. arXiv preprint arXiv:2206.03585.

[31] Kasowaki, L. and Burak, A., (2023). Cybersecurity Essentials for Robotics Process Automation Deployments (No. 11351). EasyChair.

[32] Rich, M.S. and Aiken, M., (2023). An Interdisciplinary Approach to Enhancing Cyber Threat Prediction Utilizing Forensic Cyberpsychology and Digital Forensics.

[33] Dacholfany, M.I., Miswar, M., Erliana, C.I., Abdullah, D. and Indrawati, I., (2023). EXPLORING THE INTEGRATION OF QUANTUM MACHINE LEARNING ALGORITHMS IN HIGHER EDUCATION TO ENHANCE CURRICULUM DEVELOPMENT AND CYBERSECURITY PROGRAMS. International Journal of Teaching and Learning, 1(1), pp.71-85.

[34] Abdulrahman, Y., Arnautović, E., Parezanović, V. and Svetinovic, D., (2023). AI and Blockchain Synergy in Aerospace Engineering: An Impact Survey on Operational Efficiency and Technological Challenges. IEEE Access.

[35] Sarker, I.H., Janicke, H., Maglaras, L. and Camtepe, S., (2023). Data-Driven Intelligence can Revolutionize Today's Cybersecurity World: A Position Paper. arXiv preprint arXiv:2308.05126.

[36] Hassan, S.M.U.H., (2023). STUDY OF ARTIFICIAL INTELLIGENCE IN CYBER SECURITY AND THE EMERGING THREAT OF AI-DRIVEN CYBER ATTACKS AND CHALLENGE. Available at SSRN 4652028.

[37] Fotiadou, K., Velivassaki, T.H., Voulkidis, A., Skias, D., Tsekeridou, S. and Zahariadis, T., (2021). Network traffic anomaly detection via deep learning. Information, 12(5), p.215.