

Palestine Technical University- Kadoorie

faculty of applied science

Department Applied Computing

**developing a Phishing Detection Tool**

**Prepared by:**

Ahmad Saleh-202012191

Nagham Maraheen-202012658

Muhannad Tomeh-202011744

**Supervised by:**

Dr..Eman Droubi

Tulkarm, Palestine

2022/2023

**Abstract**

The internet has become an integral part of our lives, and we often store sensitive personal and financial information online. However, as internet usage has increased, so has the prevalence of cyber threats such as phishing attacks. These attacks aim to steal personal information or cause damage, making it crucial for us to protect ourselves from these risks.

To address this issue, we have developed a website that provides a valuable service: detecting and protecting users from dangerous websites that pose a risk to their privacy and security. In this project, we utilize machine learning techniques to build a model that can accurately predict whether a website is a phishing site or not. The model is trained on a large dataset of known phishing and legitimate websites and incorporates feature scaling and a multilayer perceptron neural network architecture to optimize performance.

This project is vital for all internet users, particularly those with limited experience, as it is easy to fall prey to phishing attacks. The website is accessible globally and provides an accuracy rate of over 90% in identifying potentially dangerous sites, enabling users to avoid potentially harmful links.

The benefits of this project extend to businesses, institutions, and individuals alike. It can help ensure the safety of links to websites and protect against hacking attempts, providing a valuable reference in case of encountering external links that may pose a risk. Ultimately, this project has the potential to significantly reduce the incidence of cyber threats and safeguard the privacy and security of internet users worldwide.

**Contents:**

1. **Introduction** 
   1. Introduction
   2. Problem statement
   3. Project objectives
   4. Project benefits
   5. Project methodology
   6. Research plan (working plan)
   7. Project constraints (research limitations)
   8. Summary and recommendations
2. **System Analysis** 
   1. Introduction
   2. A brief history of the organization
   3. Project implementation options
   4. The proposed system
   5. System requirements
   6. Feasibility study
      1. Economic feasibility
      2. Technical feasibility
      3. Legal feasibility
      4. Schedule feasibility
   7. Project added values
   8. Project management
   9. Summary and recommendations
3. **System Functional Requirements** 
   1. Introduction
   2. Context or Use-Case Diagrams
   3. Functional requirements description
   4. Non-functional requirements description
   5. Summary and recommendations
4. **System Design and Development**
   1. Introduction
   2. Class diagram
   3. Sequence diagrams
   4. Entity Relationship Diagram (ER-model)
   5. Activity diagrams
   6. System interface (input/ output design)
   7. Summary and recommendations
5. **Coding and Implementation** 
   1. Introduction
   2. Coding programming languages
   3. Database system
   4. Explain algorithms and some technique
   5. Summary and recommendations
6. **Conclusions and Future Works** 
   1. Conclusions
   2. Future works
   3. Closing remarks

**List of figures:**

Figure 1.1. Cyber Phishing websites

Figure 3.2.1 context diagram.

Figure 3.2.2 use case.

Figure 4.2 class diagram

Figure 4.3 Sequence diagrams

Figure 4.4 ER-Model.

Figure 4.5 Activity diagram.

Figure 4.6.1 main page

Figure 4.6.2 tips page.

Figure 4.6.3 check page

Figure 4.6.3.1 check page (not phishing)

Figure 4.6.3.2 check page(phishing)

**List of Tables:**

Table 2.5.1 Functional System requirement

Table 2.5.2 Non-Functional System requirement

Table 2.6.1 Technical Feasibility (Hardware)

Table 2.6.2 Technical Feasibility(software)

Table 2.6.4 Schedule feasibility

Table 3.3 Functional Requirements

**Chapter** **1**

* 1. **Introduction:**

Phishing is a form of cybercrime that aims to deceive unsuspecting users into revealing their sensitive personal information, including usernames, passwords, financial details, and social connections. Attackers often disguise themselves as trustworthy sources using social engineering tactics and technical trickery.

Phishing domains represent one type of attack, where sensitive information is obtained without authorization through blackmail or fake websites that appear similar to legitimate ones. Security breaches occur when users input their private data into these sites, leaving their personal information vulnerable to identity theft. In the following figures, examples of malicious links are shown.



Figure 1.1. Cyber Phishing websites [1]

While financial and government institutions have improved their internet security measures to prevent phishing domains, the public's reliance on online services, such as shopping, banking, and bill payment, continues to grow. Successful phishing attempts can have a significant impact on global finances, increasing the need for protection online. The process of protecting internet-connected resources from cyber-attacks, including phishing, is known as cybersecurity. As cyber-attacks become more sophisticated and frequent, cybersecurity is becoming increasingly complex, making it challenging to identify, assess, and manage significant risk events. The Anti-Phishing Working Group (APWG) found over 51,000 distinct phishing websites, and according to the Rivest-Shamir-Adleman (RSA) analysis, phishing attacks cost global businesses $9 billion in 2016 [7]. In 2016, there were over one million phishing attacks, representing a 65% increase from the previous year [8]. The frequency of these attacks erodes users' trust in social networks and webpages.

There are many types of web fraud [9], and phishing websites are a common entry point for online social engineering attempts. A hacker creates a webpage by impersonating a reputable website and then sends these URLs to potential victims through spam chats, messages, or social media sites, hoping that unsuspecting users will believe it is a real URL [10]. If users enter their personal information, such as bank account numbers or government savings numbers, at the link sent by the hacker, their data will be compromised. Various strategies can be used to combat phishing [11]. Artificial intelligence (AI) has had a significant impact on almost every industry, including cybersecurity, as it can detect spam, phishing, spear phishing, and other attacks using past attacks in the form of datasets.

This project aims to develop and compare the effectiveness of machine learning (ML) classification models in detecting phishing domains. The goal is to improve detection by using the most accurate model out of the four to predict whether a webpage is a phish or legitimate. A phished domain is challenging to analyze and comprehend since it involves social and technical issues for which there is no one-size-fits-all analysis. As a result, all phishing domain causes and features were analyzed quantitatively and qualitatively to determine where to focus the model to better decrease the danger arising from a visit to a phishing website, particularly regarding consumer trust.

* 1. **Problem Statement:**

Phishing is one of the easiest types of cyberattacks for hackers to execute, and it is also one of the easiest types for people to fall for. This is often done to steal user data, such as login details, credit card numbers, and other personal information. The attacker poses as a trusted entity to trick the person into clicking on a specific link or opening an email or WhatsApp message. Sometimes, the links are disguised, making it impossible to know their source.

If the recipient clicks on the link, malware may be installed, the system may be frozen (a type of ransomware attack), or sensitive information may be exposed due to a security vulnerability in the operating system.

Statistics reveal that there are over 4.1 billion email users, and about 3 million emails are sent per second. This means that more than 50% of the world's population uses email, and the number is expected to rise to 4.5 billion users by 2024. These high numbers have encouraged criminals to focus their attacks on email rather than other sources. With COVID-19, many organizations have ordered their employees to work remotely for their safety, and this has led to many people relying heavily on email communication. With employees receiving a large number of messages daily, they are more vulnerable to cyberattacks. When it comes to phishing, all it takes is one click to lose everything.

Phishing attacks, in their most common form, are email messages that urge mail recipients to take action, usually to achieve one of two goals: trick the user into sharing their personal information or tricking the user into downloading malicious software. Once the software has access, hackers can access bank accounts, steal identities, or make fraudulent purchases in the victim's name. A common example of this deception is a fake letter from your bank containing a request to update your personal information by logging into your account. Clicking on the link in the message will direct you to a site that looks identical to the bank's website. When you enter the requested username and password, this data is sent to the hacker's address, not the bank.

Over the past few years, email scams have increased by over 400%. The growth and success of email phishing scams have also led to the proliferation of these scams. We will discuss more about this below:

* SMiShing, as the name suggests, is similar to phishing emails, but it tricks users via text messages. Although many people are aware of email phishing, the level of awareness about SMS fraud is lower, increasing the possibility of people falling victim to a phishing scam.
* Spear phishing uses the same methods of trickery mentioned above, but it targets a specific person. You might receive a series of emails designed to entice you to take action. Phishing attacks can also target multiple messaging platforms.
  1. **Project objectives:**

The following points summarize the main objectives of this project:

* Investigating the nature of cyberattack content and its types.
* Developing a website for detecting cyberattacks and phishing systems.
* Conducting extensive experiments to evaluate the proposed model.
  1. **Project benefits:**

**- For users:**

Building a tool for detecting cyberattacks in websites, which may be added to Facebook officially as an add-on, may play a role in:

* Making detection of phishing emails or websites easier for users.
* Making the internet safer and reducing attacks.
* Protecting people from attackers and safeguarding their personal data.

**- For developers:**

* Enhancing and expanding general knowledge about cyberattacks and improving the experience of developing useful data mining-based programs.
* Learning about the many algorithms that are used to solve a problem and which are the building blocks of the advanced digital world we see today. It is an important concept that must be understood, because in machine learning, learning algorithms - not computer programmers - create the rules.
  1. **Project methodology:**

Software development methodologies are essential and are mostly used for various software development projects. Furthermore, each of these methodologies works well in specific projects depending on the nature of the project. None of these methodologies are foolproof as each has its pros and cons. The basic purpose of these methodologies is to provide smooth software development according to the project requirements [3].

In our project, we will adopt the waterfall model because:

* The requirements are well understood, so there is no need to send a prototype for each small update.
* It is simple and easy to understand and use.
* The project status is more easily measured based on a complete schedule and resource plan.
* The risk is zero or minimum.
  1. **Research Plan :**

In this project, we are going to build a machine learning model to predict whether a website is a phishing site or not. We will start by fetching the data and then preprocess it by applying feature scaling. Next, we will create a neural network using MLP (multilayer perceptron) and set the number of layers and neurons for each layer. Finally, we will set the number of iterations for training the model.

* 1. **Project constraints:**

There are some big challenges for any machine learning project, and in this project, we need to mention some of them:

* Accuracy of the model: In any machine learning model, we need to optimize the model to achieve high accuracy. Therefore, we aim to maximize the accuracy percentile.
* Model efficiency: If the model takes a very long time to predict whether a website is phishing or not, then what is the benefit? Therefore, in this project, the performance and effectiveness of the algorithm used will be taken into account.

**1.9 Summary and Recommendation:**

In this chapter, we have discussed the introduction to this project, the problem statement of the project and its solution. We have mentioned the objectives of the project and how it will benefit both developers and users. We have also discussed some of the methodologies and explained how the work plan will be carried out. Finally, we have addressed the project constraints and limitations.

**Chapter 2**

**{System Analysis}**

**2.1 Introduction :**

In this chapter, we will cover important topics. We will provide an explanation of system analysis in general and discuss previous research related to our project. We will also explain the available implementation options and specify the one we have chosen. Furthermore, we will outline the system requirements. Finally, we will present a feasibility study for this project.

**2.2 Related work:**

In general, website users tend to overlook the URL, which can make them more vulnerable to falling prey to phishing domains. Traditional methods for detecting phishing attacks have limited accuracy, while ML techniques for phishing detection produce better results but are time-consuming and not scalable. One way to reduce features is through feature selection strategies. To train a machine learning model for phishing detection, the dataset needs to have properties relevant to phishing and legitimate website classes. Classifiers like DT, C4.5, k-NN, and SVM are important and have been used in numerous research projects. Researchers have suggested different strategies to combat phishing attacks, such as using different classifiers or ensemble learning techniques. The most effective classifiers include random forest, extra-tree base, and stacking. Other methods include using particle swarm optimization to weigh different aspects of a website to detect phishing from legitimate websites accurately.

In their study, Patil et al. [40] proposed three techniques to detect phishing websites. The first method involved evaluating various URL attributes, the second involved verifying the website's validity by identifying its host and manager, and the third method involved analyzing the website's appearance to determine its authenticity. To evaluate the different aspects of URLs and websites, the researchers utilized machine learning algorithms and methodologies.

Meanwhile, Joshi et al. [41] developed a binary classifier utilizing the reliefF algorithm for feature selection and an RF algorithm. The Mendeley domain provided the data used to select features. The chosen features were then used to train the RF algorithm to detect phishing attacks.

In the research by Ubing et al. [42], three ensemble learning strategies (bagging, boosting, and stacking) were utilized. Their dataset comprised 30 attributes and 5126 entries in the result column. The dataset was sourced from UCI, which is accessible to the public. Their classifiers were combined using a DT to achieve the highest possible accuracy.

Lastly, the authors of [43] proposed a novel approach that employed both URLs and HTML-related data as inputs. Following feature extraction, a stacking strategy was utilized to combine the learners.

**2.3 Project implementation options:**

There are several options for implementing projects, such as Desktop Applications, Web Applications, and Mobile Applications.

* Desktop Applications are standalone applications that run on systems and laptops.
* Web Applications are accessed through the internet via a web browser. A web browser allows you to access the app and its content, and also runs all the scripts responsible for its features. What differentiates a simple static web page from a web application is interactivity. They often allow you to create, edit, or manipulate data and content.
* Mobile Applications are designed to run on mobile devices, such as smartphones or tablet computers.

In our project, we are using Python to develop a Web Application. Python provides many useful features that make it popular and valuable compared to other programming languages. It supports object-oriented programming and procedural programming approaches, and additionally, it has many libraries that support data science and machine learning. Web Applications are easy to use for anyone and much more enjoyable than Mobile Applications.

**2.4 The proposed system :**

The main objective of the project is to discover phishing sites that can help the general public, companies, and institutions to protect them from trap sites that can cause them harm and expose them to theft or damage to their sites.

In this project, we are going to implement a website for phishing detection. The tool will have information about phishing that the admin gives it as a dataset from Kaggle that contains a lot of websites and their results, whether they are phishing or not. The tool will be trained about phishing websites and will be ready to detect new websites checked in the future.

The tool will use many algorithms like Logistic Regression, SVM Support Vector Machine, Naive Bayes, K Nearest Neighbors, Neural Network(MLP), and Multilayer Perceptron that the tool will depend on to make decisions. The tool will have an edit text that allows the user to enter a URL. Then, based on a trained ML model, it will check whether the URL is phishing or not.

**2.5 System requirements:**

The functional requirements explain how the system should work, while the non-functional requirements explain how the system should perform.

**2.5.1 The Functional requirement of the system:**

Table 2.5.1 shows the Functional System requirement.

|  |
| --- |
| Data Preprocessing 🡪 Clean Data And Convert Sentence To Vectors |
| Rate Model 🡪 Rate Model From 5 And Write Feedback |
| Import Data 🡪 Extract Data From Kaggle |
| Classification 🡪 Classification Weather URL Phishing OR Not |
| Retrain Model 🡪 Train Model From Imported Data To Improve Results |
| Show Tips🡪 Show Tips For User For Increase Awareness For User . |
| Invite Friend🡪Invited Friend Via Facebook , Linkedin , … |
| Show Results🡪Show Precision , Recall And F1 Score For Model |
| Show Rates & Feedback 🡪 Show People Rates For Model |

Table 2.5.1 Functional System requirement

**2.5.2 Non Functional Requirements:**

Table 2.5.2 shows the Non Functional System requirement.

|  |  |  |
| --- | --- | --- |
| **Description** | **Nonfunctional** | # |
| Clear, simple and easy that can be understood and used by anyone who does not have experience in the field | Understandable &Usability | 1 |
| Be available to all users at any time | Availability | 2 |
| Able to give high F1 Score result | Correctness | 3 |
| Ability to perform speedily and execute efficiently | Performance | 4 |
| Ability to cope with errors during execution and cope with erroneous input | Robustness | 5 |

Table 2.5.2 Non Functional System requirement

**2.6 Feasibility study:**

**2.6.1 Technical Feasibility:**

**-Hardware:**

|  |  |
| --- | --- |
| **Components** | **Description** |
| HP Laptop | Intel(R) Core(TM) i5-1035G1 CPU @ 1.00GHz 1.19 GHz |
|  |  |
| HP Laptop | Intel(R) Core(TM) i3-6006U CPU @ 2.00GHz 2.00 GHz  RAM : 8.00 GB |
| ASUS | Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz 1.99 GHz  RAM: 20.0 GB |

Table 2.6.1 Technical Feasibility (Hardware)

**-Software:**

|  |  |
| --- | --- |
| **Components(programs)** | **Description** |
| PyCharm | To build the Apps |
| Sql Server | To create the data base |
| EdrawMax, VP | To draw the diagrams |
| Internet Connectivity | Available at the client phone |

Table 2.6.2 Technical Feasibility(software)

**2.6.3 Legal feasibility:**

The proposed project ensures that it is legally accept and conform the legal and ethical requirements.

**2.6.4 Schedule feasibility:**

Table 2.6.4 Schedule feasibility

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Task** | **First month** | | | | **Second month** | | | | **Third month** | | | | | **Fourth month** | | | |
| **Week 1** | **Week 2** | **Week 3** | **Week 4** | **Week 5** | **Week 6** | **Week 7** | **Week 8** | **Week 9** | | **Week 10** | **Week 11** | **Week 12** | **Week 13** | **Week 14** | **Week 15** | **Week 16** |
| **Find my group** |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| **Find supervisor** |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| **Find idea** |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| **Project introduction** |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |  |
| **System**  **analysis** |  |  |  |  |  | | |  |  | |  |  |  |  |  |  |  |
| **System functional requirements** |  |  |  |  |  |  |  |  |  |  | |  |  |  |  |  |  |
| **Documentary** |  |  |  |  |  | | | | | | | | | | | | |

**2.7 Project added values:**

The website will educate users about dangerous and fraudulent links, and increase their awareness of the risks associated with them. Simple advice and guidance should be provided to help users develop sufficient knowledge and understanding of these risks.

**2.8 Project management :**

- Supervision : Dr. Iman Droubi

The team:

1- Ahmad Saleh : leader / task : backend

2- Nagham Maraheen : member / task : interfaces

3- Muhannad Tomeh : member / task : interfaces

**2.9 Summary and recommendations :**

In this chapter, we discussed system analysis, starting with a brief history of the organization, considering previous projects and how their performance evolved over time. We also discussed project implementation options and the proposed system that illustrates where and how the system will be applied. Additionally, we referred to the functional and non-functional system requirements, discussed the feasibility study, project added values, and concluded by mentioning project management.

**Chapter 3**

**{System Functional Requirements}**

**3.1 Introduction:**

Requirements analysis is critical to the success or failure of a systems or software project. The requirements should be documented, actionable, measurable, testable, traceable, related to identified business needs or opportunities, and defined to a level of detail sufficient for system design. In this chapter, we will use the "use case diagram" to show how users interact with our application. Additionally, we will present the functional and non-functional requirements of this application.

**3.2 Context or Use-Case Diagrams:**

**Context Diagram :**

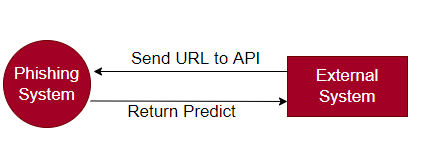
****

Figure 3.2.1 context diagram.

**Use Case Diagram:**

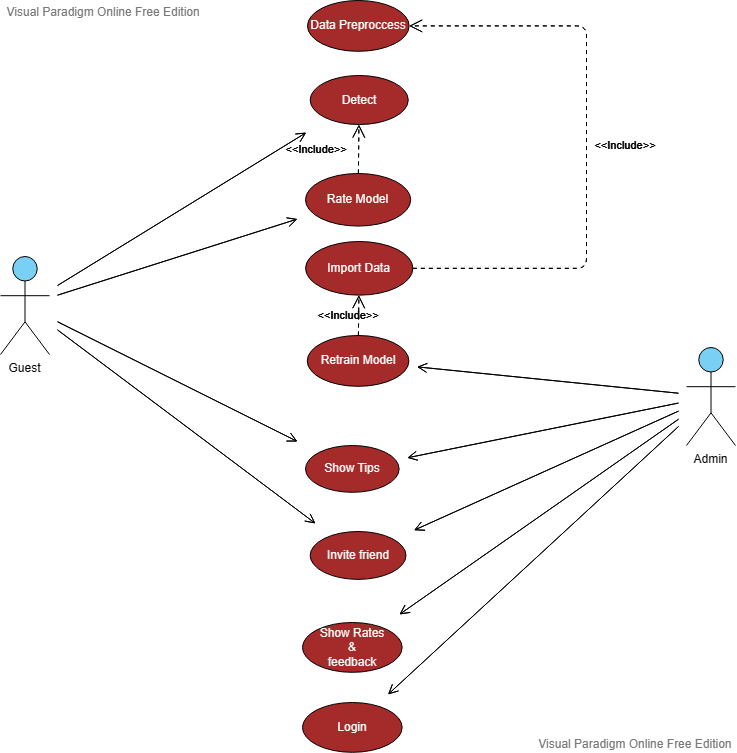


Figure 3.2.2 use case.

**3.3 - Functional Requirements:**

Table 3.3 Functional Requirements

|  |  |
| --- | --- |
| **Functional** | # |
| **Data Preprocessing**  1- Actor🡪Admin.  2- Input🡪Data OR URL.  3- Description🡪 Preprocess Data Like Convert Text To Vector Using Technique Word2Vec And Cleaning Data By Ignoring Missing Values | 1 |
| **Detect**  1- Actor🡪Guest.  2- Input🡪URL.  3- Description🡪 Check Weather URL Is Phishing OR Not  4- Reaction/Output🡪 Alter If Its Phishing OR Its Not Phishing | 2 |
| **Rate Model**  1- Actor🡪Guest.  2- Input🡪Rate From 5 And Feedback.  3- Description🡪Rate Model.  4- Reaction/Output🡪Rated Successfully OR Not | 2 |
| **Import Data**  1- Actor🡪Admin.  2- Input🡪Data  3- Description🡪Import Data From Admin Device To Train Model On It  4- Reaction/Output🡪Imported OR Not | 2 |
| **Retrain Model**  1- Actor🡪Admin.  2- Input🡪Data To Train Model On It  3- Description🡪Retrain Model For Getting Better Results  4- Reaction/Output🡪 Trained Successfully OR Not With New Information About Recall, Precision And F1 Score | 3 |
| **Show Tips**  1- Actor🡪Guest , Admin.  2- Input🡪Press Button Show Tips  3- Description🡪 Show Some Tips For Increase Awareness For User .  4- Reaction/Output🡪 Selection of appropriate and suitable data. | 4 |
| **Invite Friends**  1- Actor🡪Guest, Admin.  2- Input🡪  3- Description🡪Open Small Dialog To Share Website Using Facebook , Linkedin ,..  4- Reaction/Output🡪 | 5 |
| **Show Rates & Feedback**  1- Actor🡪Admin  2- Input🡪 Press Button Show Rates & Feedback  3- Description🡪Show People Ratings And His Feedback  4- Reaction/Output🡪 | 6 |
| **Login**  1- Actor🡪Admin  2- Input🡪 Email And Password  3- Description🡪Login Page For Admin  4- Reaction/Output🡪Login Successfully OR Failed | 7 |

**3.4 Non Functional Requirements:**

Table 3.4 Non Functional Requirements

|  |  |  |
| --- | --- | --- |
| **Description** | **Nonfunctional** | # |
| Clear, simple and easy that can be understood and used by anyone who does not have experience in the field | Understandable &Usability | 1 |
| Be available to all users at any time | Availability | 2 |
| Able to give high F1 Score result | Correctness | 3 |
| Ability to perform speedily and execute efficiently | Performance | 4 |
| Ability to cope with errors during execution and cope with erroneous input | Robustness | 5 |

**3.5 Summary and recommendations:**

In this chapter, we discussed Context, Use-Case Diagrams, Functional Requirements Description, and Non-Functional Requirements Description.

**Chapter 4**

**{System Design and Development}**

**4.1 Introduction:**

This chapter will illustrate various diagrams for our system, namely Class Diagram, Sequence Diagram, ER Diagram, and Activity Diagram. Additionally, we will showcase the expected interfaces of our application through a prototype.

**4.2 Class diagram:**

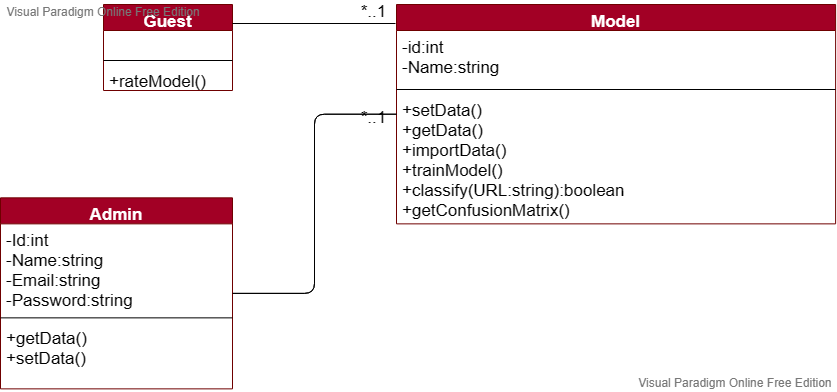
****

Figure 4.2 class diagram

**4.3 Sequence diagrams:**

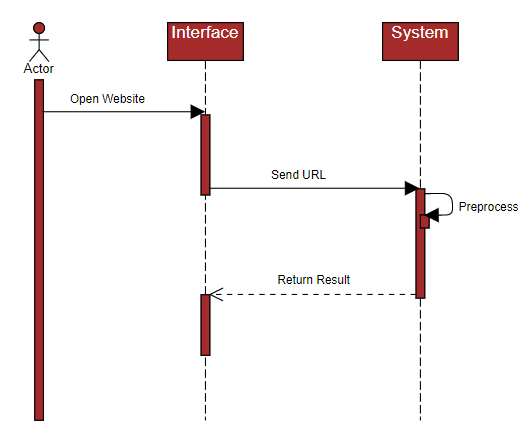


Figure 4.3 Sequence diagrams

**4.4 Entity Relationship Diagram (ER-Model)**

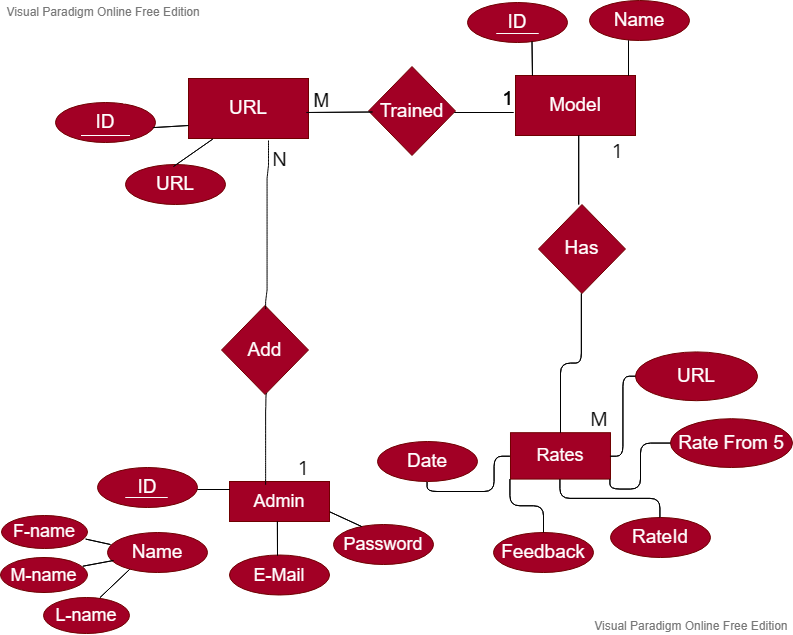
****

Figure 4.4 ER-Model.

**4.5 Activity diagrams:**

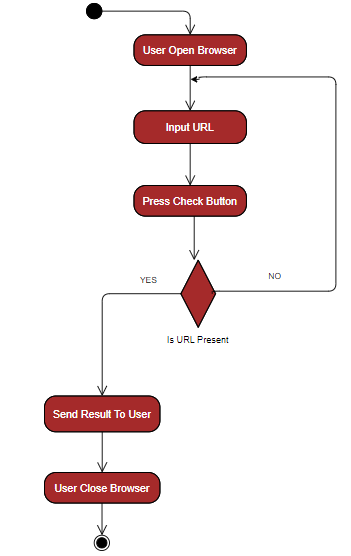
****

Figure 4.5 Activity diagram.

**4.6- System interface (input/ output design) :**

**-Main Page :**

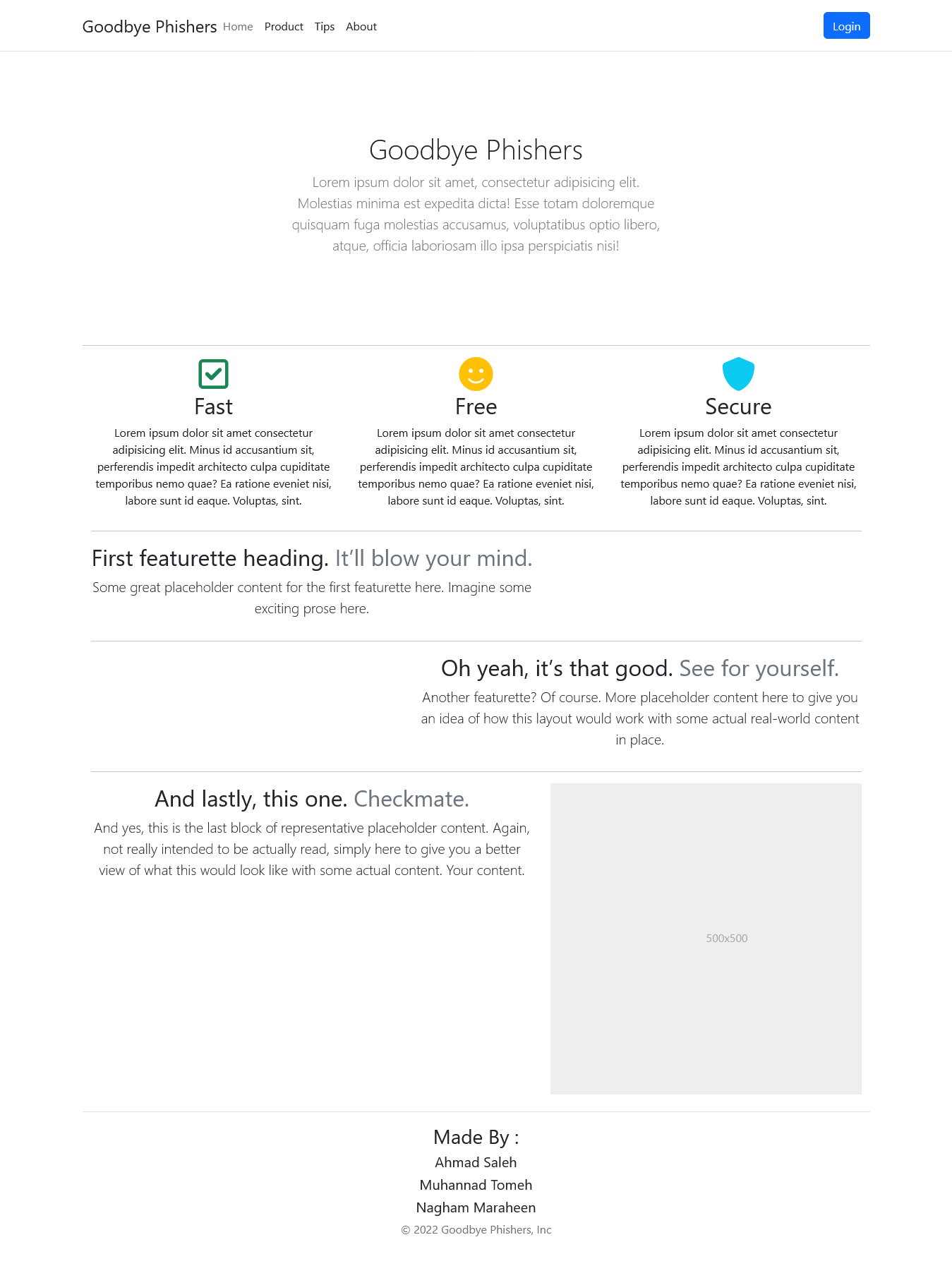
****

Figure 4.6.1 main page

**-Tips Page :**

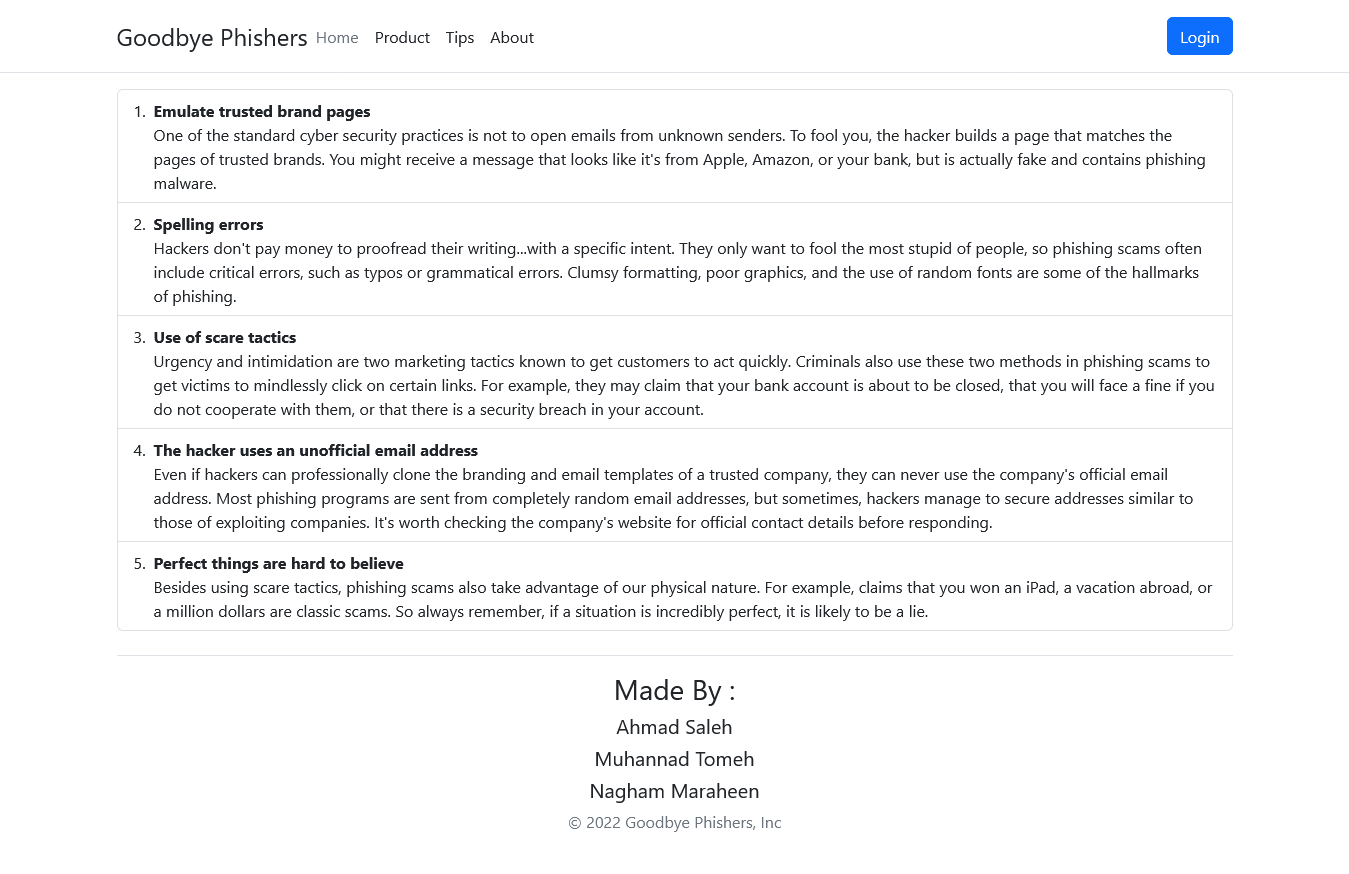


Figure 4.6.2 tips page.

**- Check Page :**

when User Not Test Product At Least One Time  
He Cant Make Rate

So Button Rate Model Is Disabled

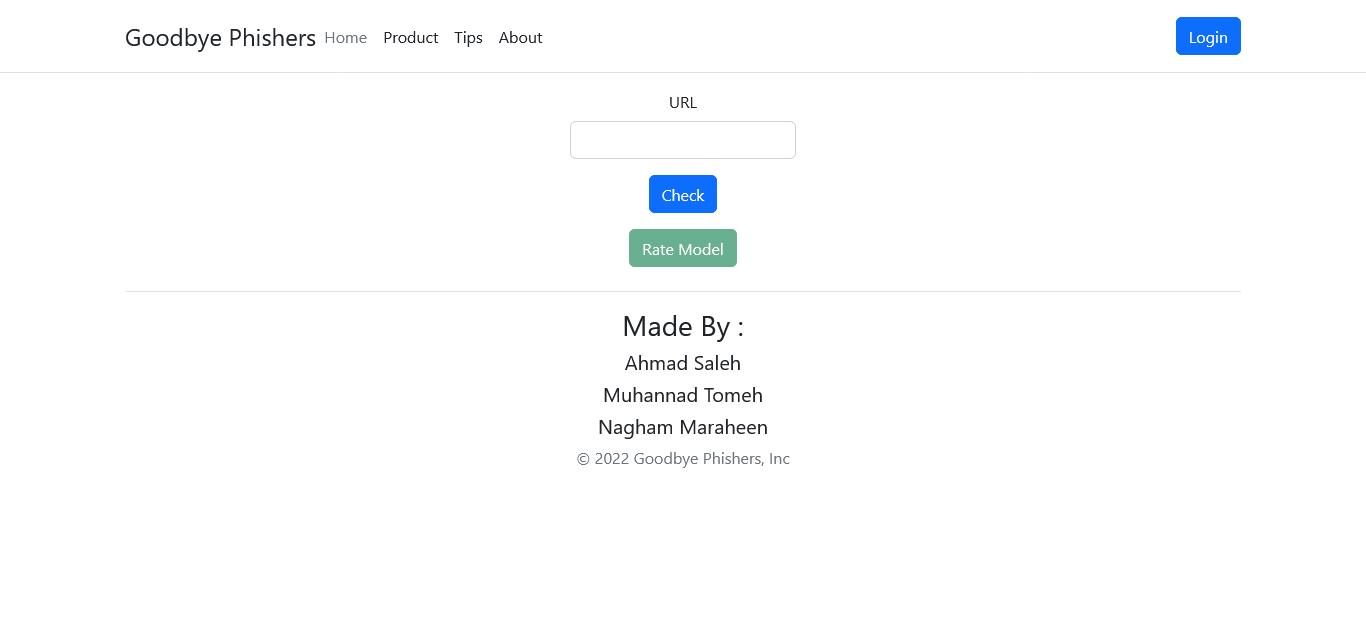


Figure 4.6.3 check page

**-Check Page(URL Is Not Phishing) :**He Can Make Rate Because He Use Model

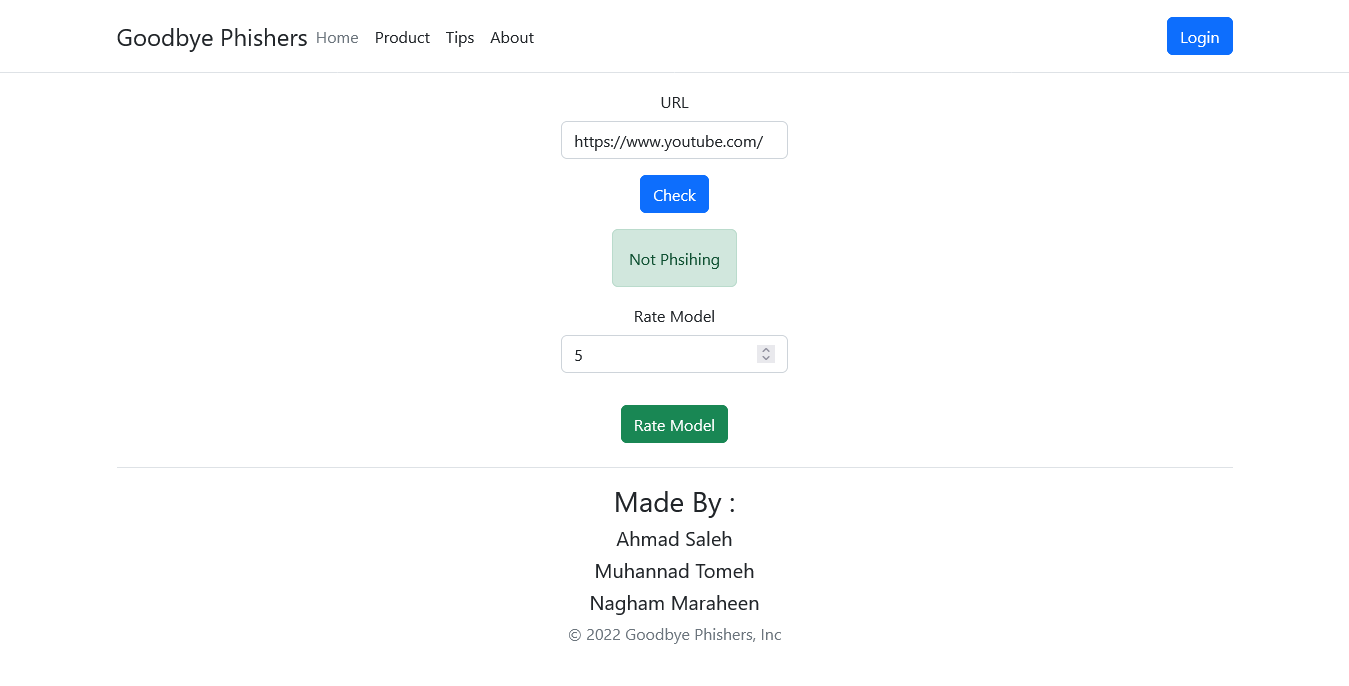
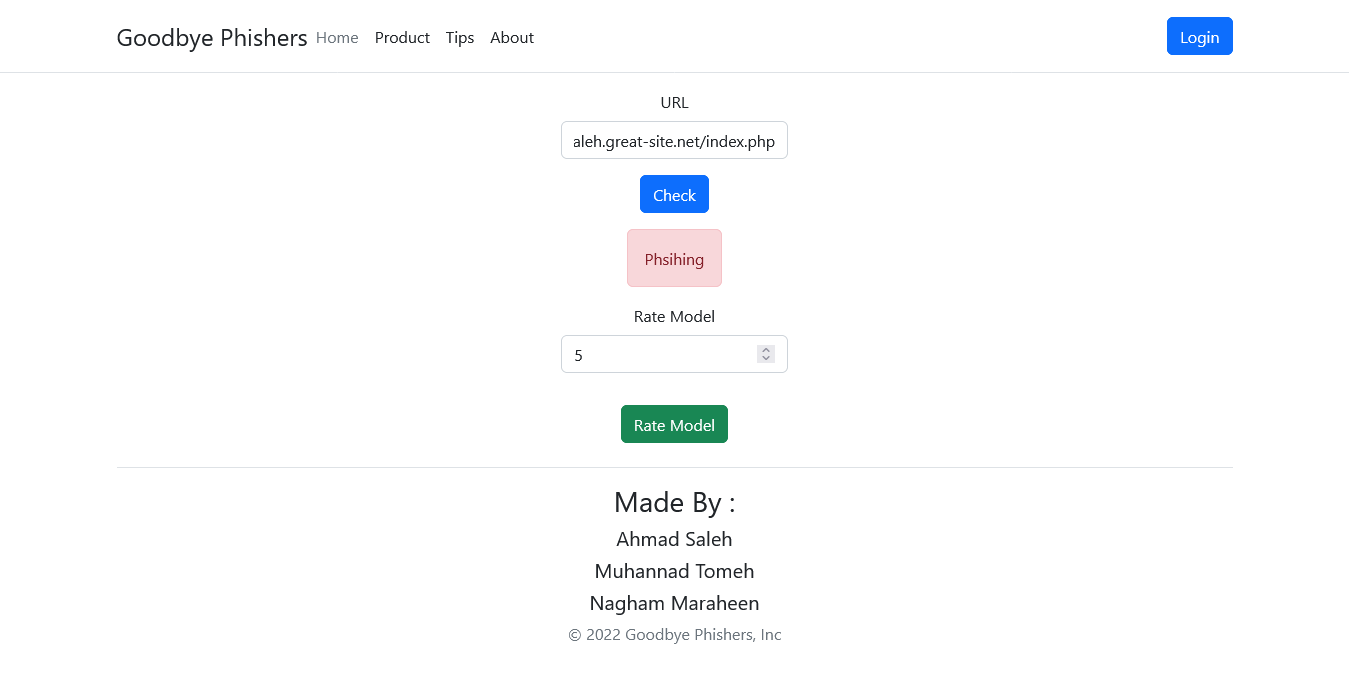
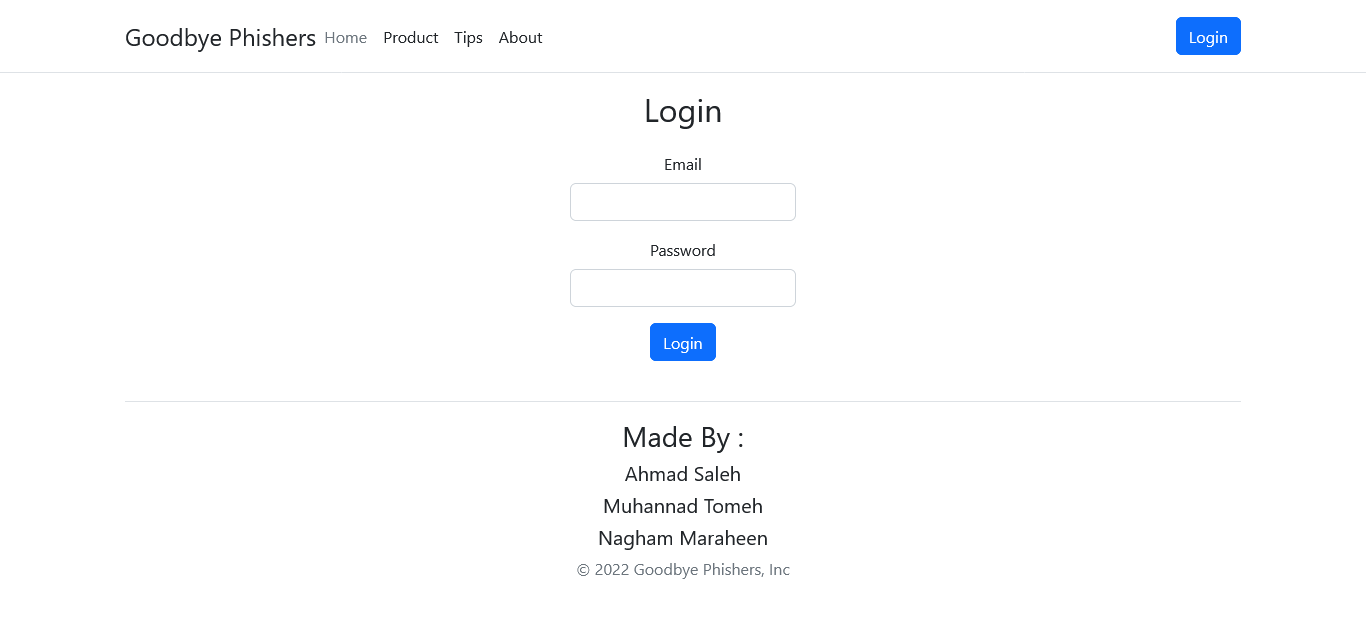


Figure 4.6.3.1 check page (not phishing)

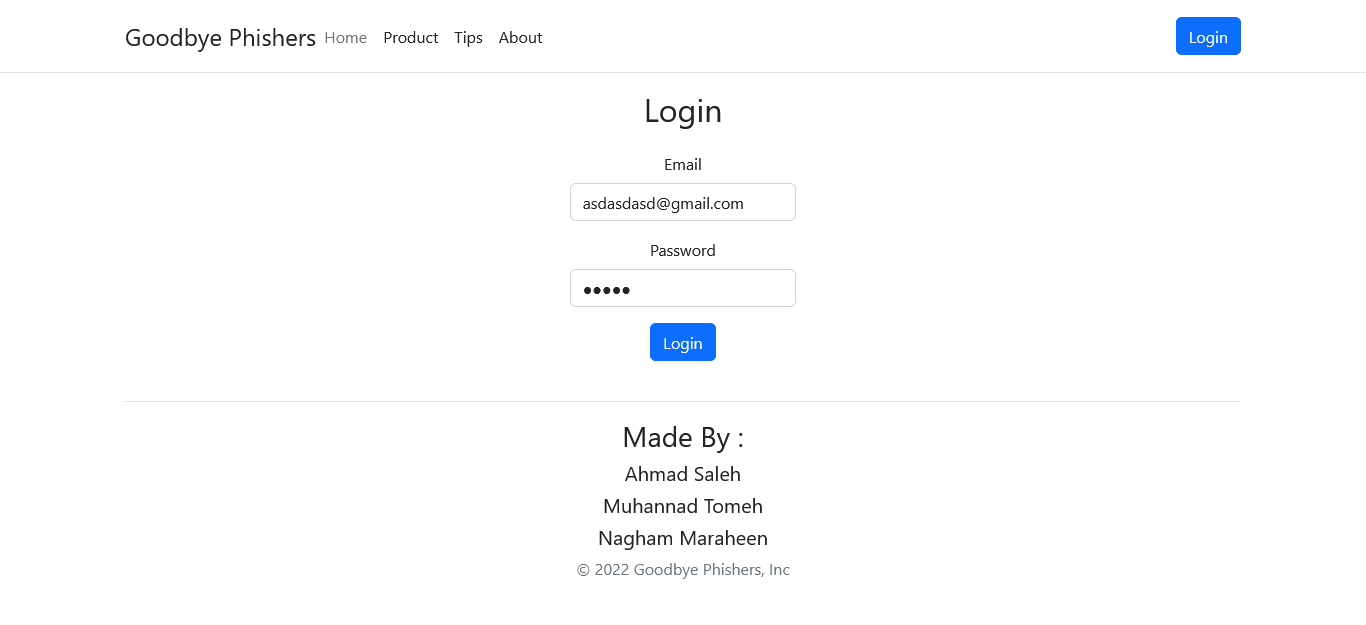
**- Check Page(URL Is Phishing) :**He Can Make Rate Because He Use Model



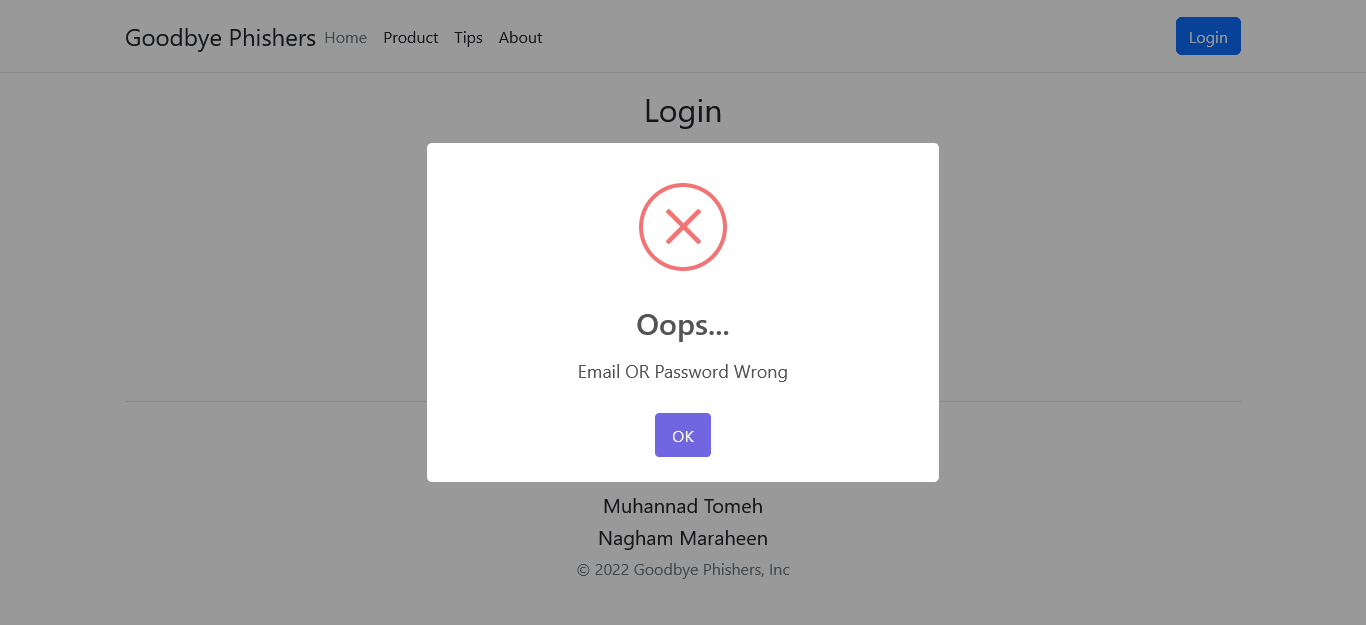
**-Login Page :**



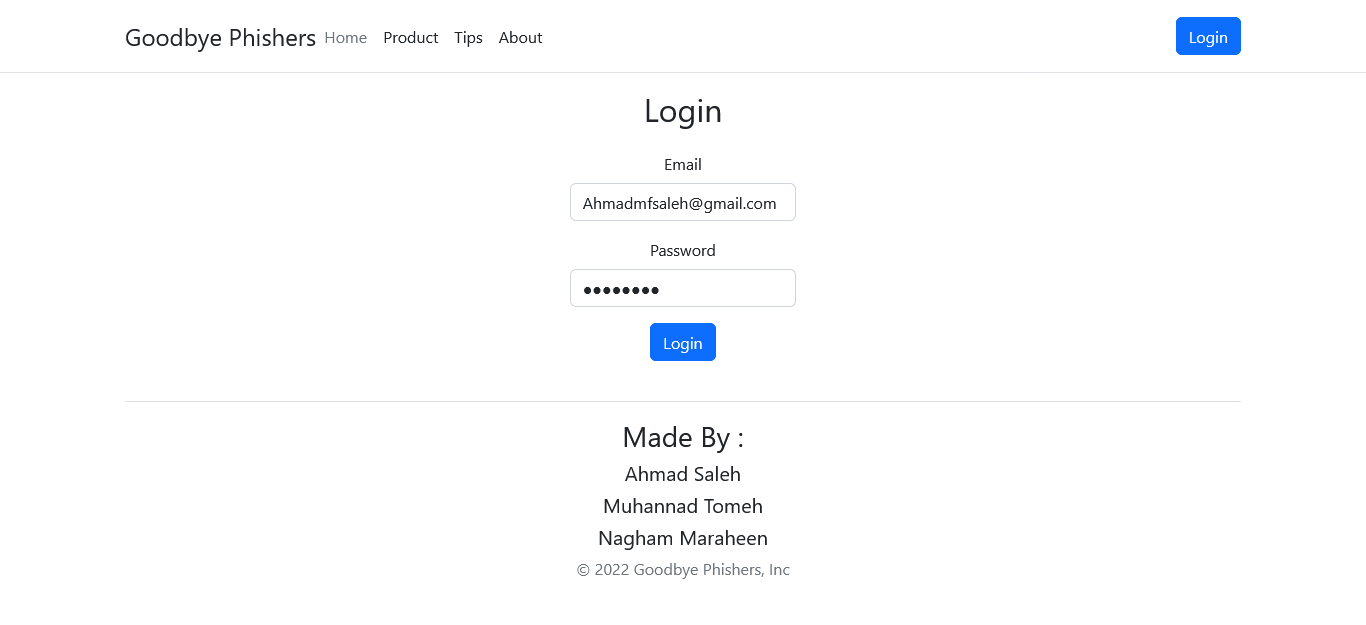
**When Enter Wrong Informations :**

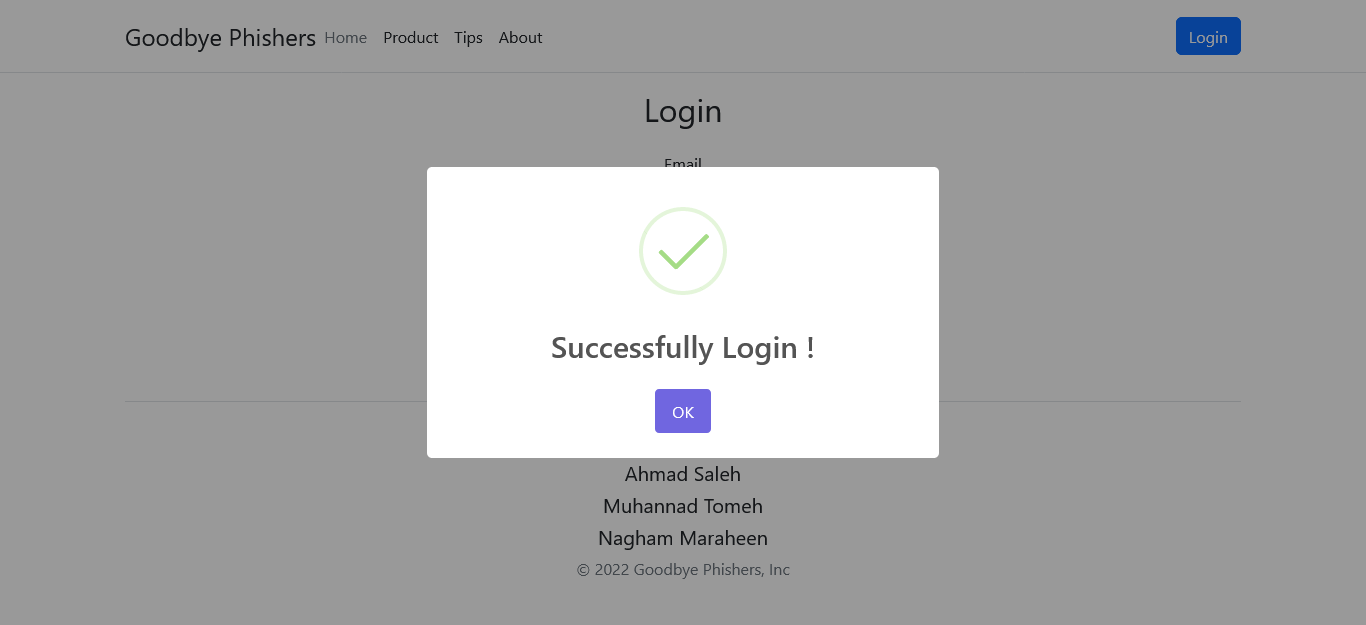


**Reaction :**

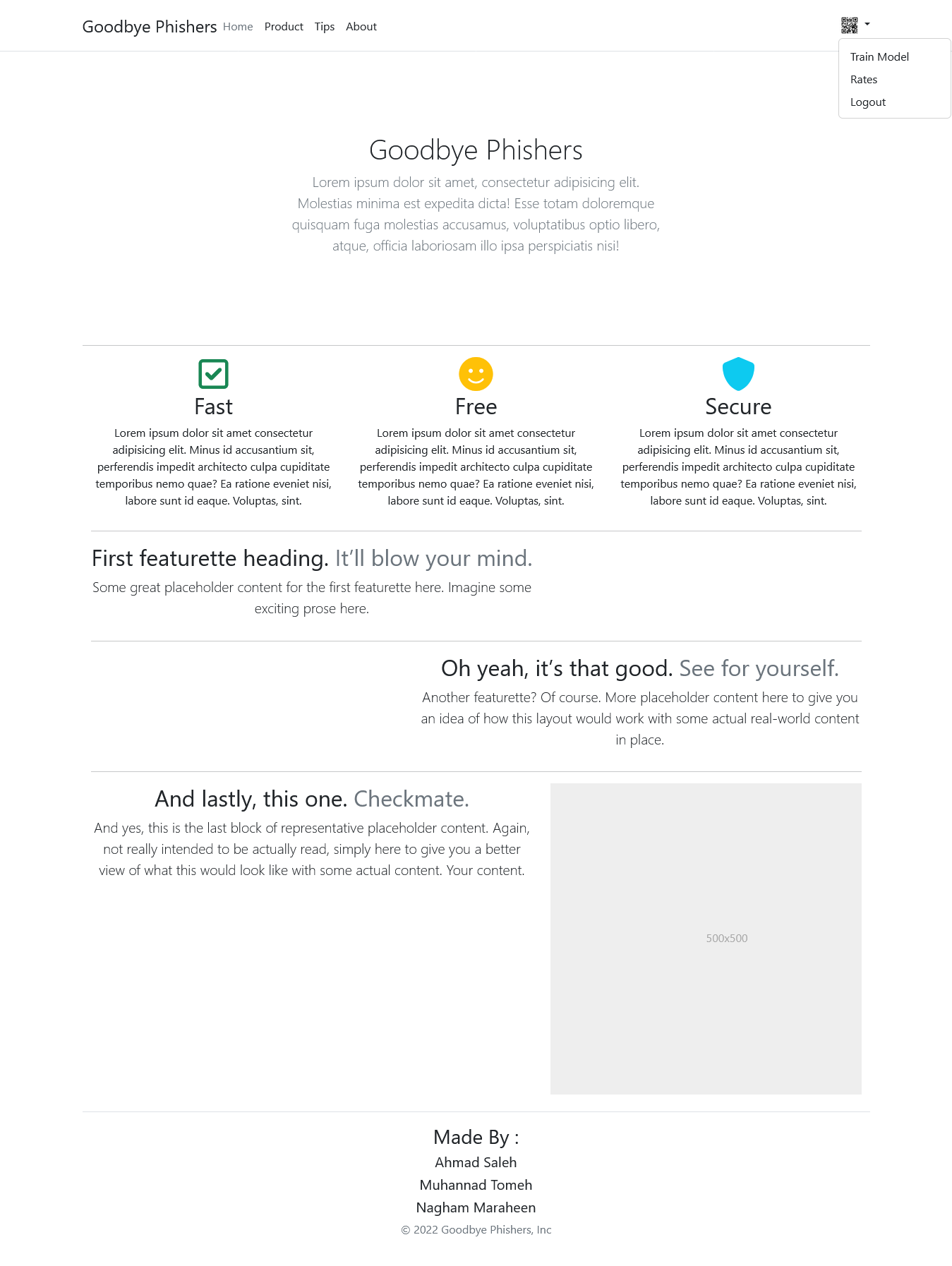


**When Enter Correct Information :**

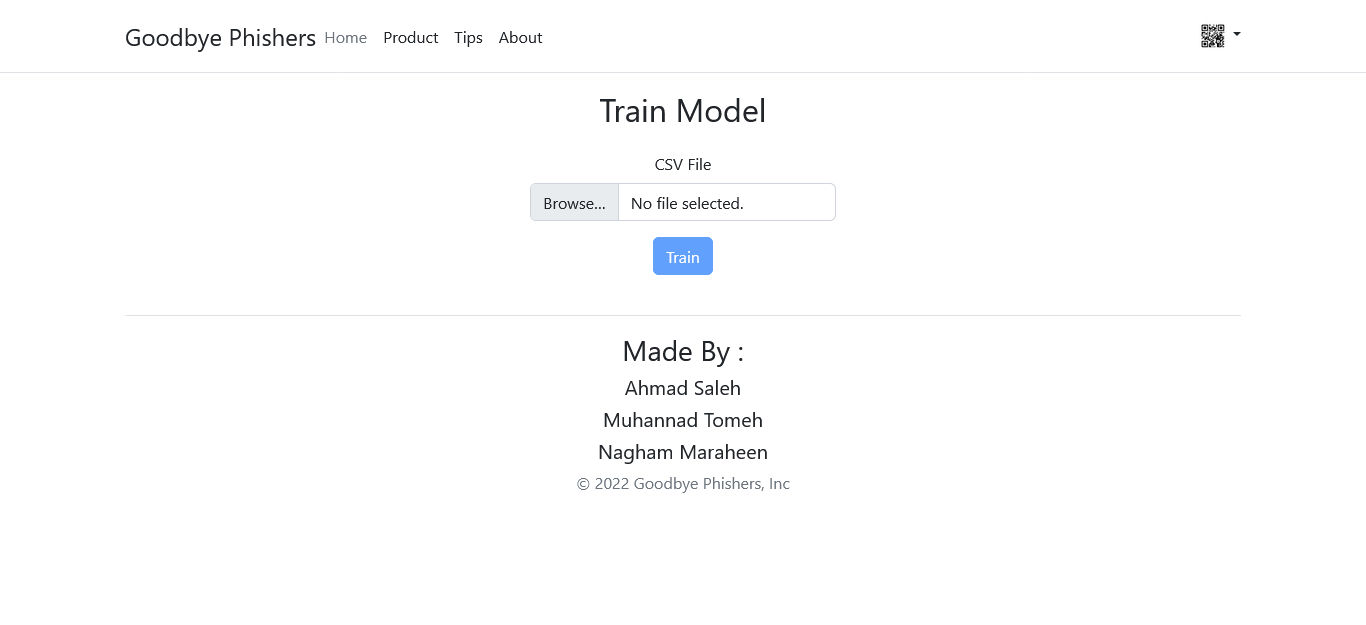


**Reaction :**   


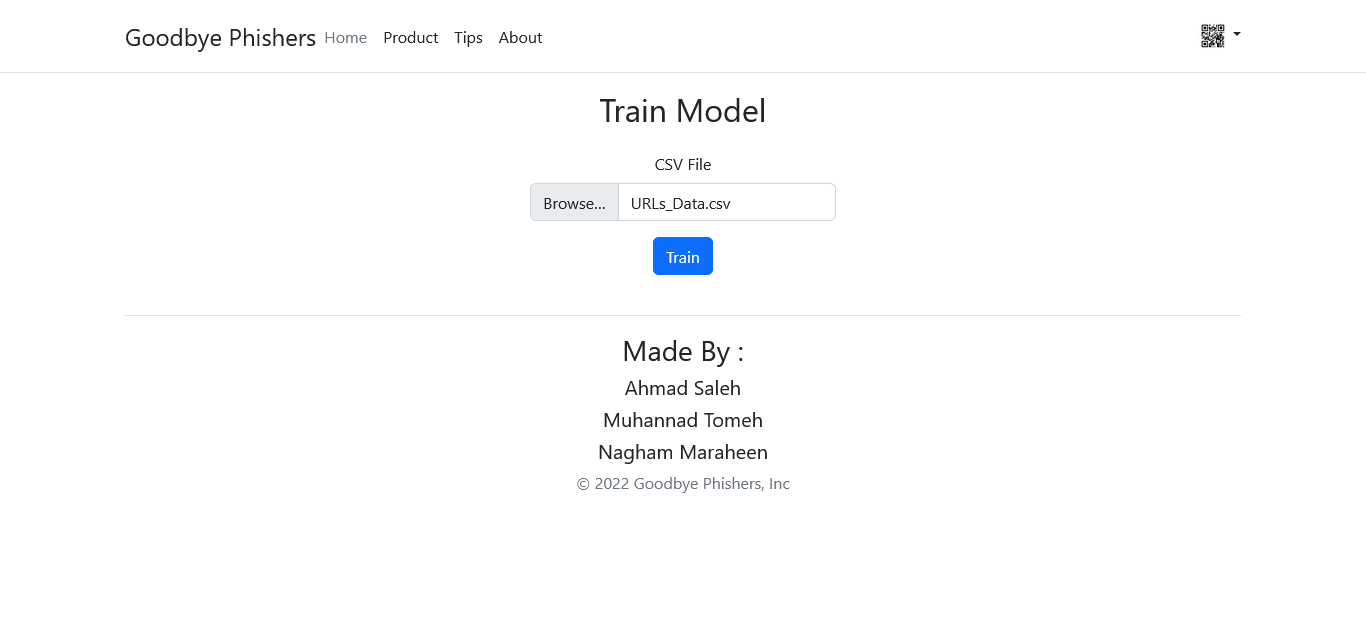
**When Admin Successfully Login :**



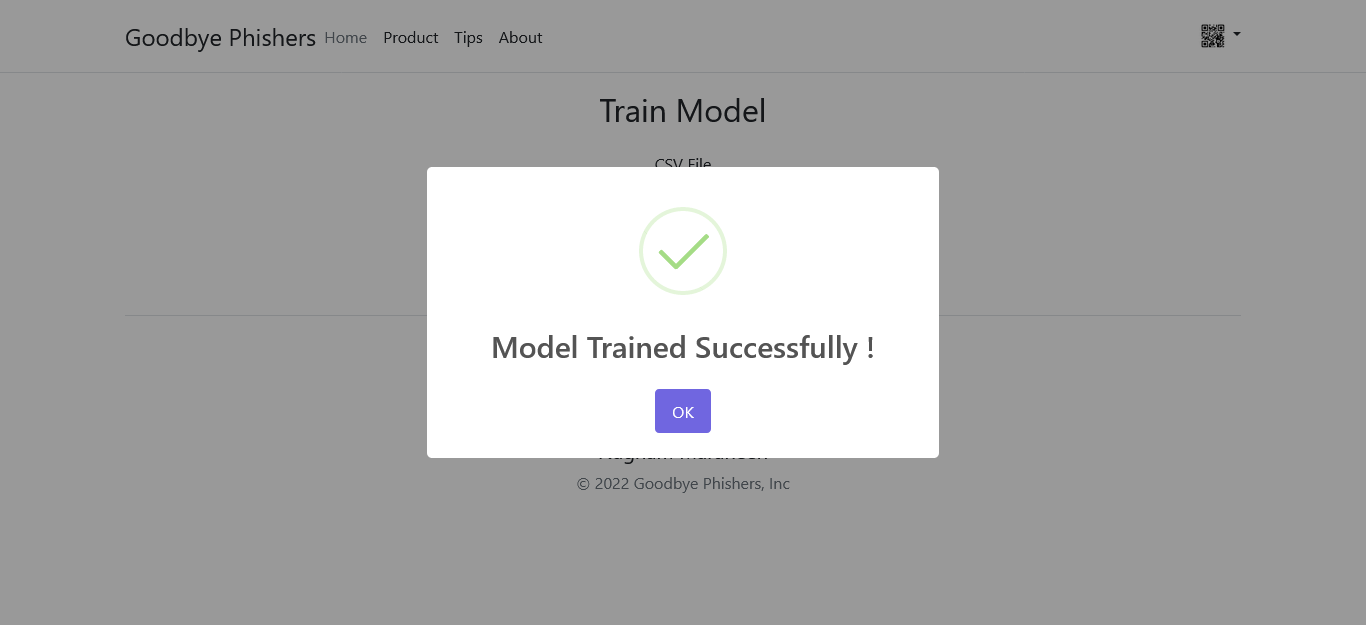
**When Click Train Mode :**



**When Upload Data :**



**Reaction After Train It :**



**When Click Rates :**

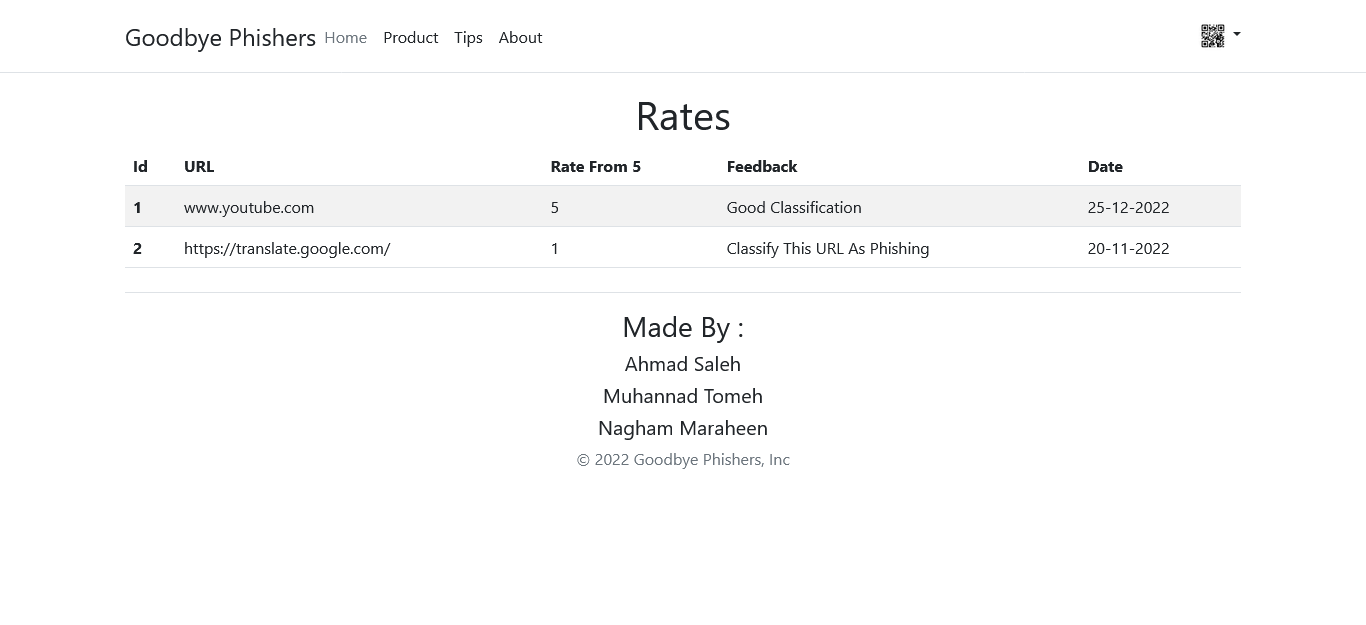


Figure 4.6.3.2 check page(phishing)

**4.7 Summary and recommendations :**

In this chapter, we talked about Class diagram , Sequence diagrams, Entity Relationship Diagram (ER-model) , Activity diagrams and System interface (input/ output design).

**Chapter 5**

**{Coding and Implementation }**

**5.1 Introduction**

In this project, we employed a methodology that involved utilizing datasets from PhishTank and openDataCommons. These datasets provided us with a wide range of phishing instances, enabling us to develop robust phishing detection models. To achieve this, we implemented three different algorithms, namely LinearSVC, Logistic Regression, and Naive Bayes, to construct our models.

To further enhance the accuracy of these models, we incorporated the TF-IDF scoring technique as a preprocessing strategy. By leveraging TF-IDF scoring, we aimed to improve the feature representation and subsequently enhance the overall performance of the phishing detection models.

Following sections discuss the coding and implementation of our phishing URL detection tool using machine learning algorithms and natural language processing techniques. The tool is designed to detect phishing URLs using a machine learning model trained on a dataset of phishing and legitimate URLs.

**5.2 Coding Programming Language**

We chose to use Node.js with the Express framework for creating the backend of our tool. Express is a minimal and flexible Node.js web application framework that provides a set of robust features for web and mobile applications. It allows us to create server-side APIs that enable us to perform various functions such as adding rates to the model, login for admins, and adding new URLs to train the model.

On the frontend, we used HTML, CSS, JavaScript, Bootstrap 5, and Font awesome. We also used the React framework to build a modern and dynamic user interface.

For the machine learning model, we used the Python programming language and the FastAPI library to create an API for accessing the model from the Node.js backend. Python is widely used for machine learning tasks due to its powerful libraries and packages such as NumPy, pandas, and Scikit-learn.

Pros:

* Node.js with Express framework provides a fast and scalable backend
* React framework provides a modern and dynamic frontend
* Python with Scikit-learn provides a wide range of machine learning algorithms and libraries
* FastAPI provides a simple and fast way to create APIs in Python

**5.3 Database System**

We chose to use MySQL as our database system for this project. MySQL is a free, easy-to-use, and widely supported database system by hosting services. We used MySQL to store the URL dataset and to persist the training data and parameters of the machine learning model.

On the other hand, PhishTank is utilized for training the model. The dataset from PhishTank specifically focuses on phishing URL detection. It consists of URLs that have been identified and verified as phishing attempts, providing valuable resources for researchers and cybersecurity professionals. By leveraging this dataset, algorithms and machine learning models can be developed to accurately detect and prevent phishing attacks, ensuring user safety and protection against online scams.

On the other hand, the dataset from Open Data Commons provides a broader perspective, encompassing both good and bad URLs. It includes a diverse collection of web addresses, ranging from legitimate and safe URLs to malicious or potentially harmful ones. This dataset contains classifications that indicate whether a URL is considered good or bad, enabling researchers to study and analyze the characteristics and patterns of different types of URLs. By utilizing this dataset, researchers can develop robust machine learning models to classify URLs accurately, enhancing web security and safeguarding users from various cyber threats.

**5.3 Explain algorithms and some technique**

We employed multiple machine learning algorithms to train our phishing URL detection model, including Linear SVC, Logistic Regression, Naive Bayes, Neural Network, and Random Forest Trees.

1. Linear SVC: Linear Support Vector Classification is a linear model used for classification. It creates a hyperplane in a high-dimensional feature space to separate classes. The algorithm aims to maximize the margin between the two classes by finding the optimal hyperplane.
2. Logistic Regression: Logistic Regression is a classification algorithm that models the probability of a binary target variable based on independent variables. It utilizes a logistic function to estimate the parameters through maximum likelihood estimation.
3. Naive Bayes: Naive Bayes is a probabilistic classification algorithm that assumes independence between features. It calculates the probability of each class given the features and selects the class with the highest probability.
4. Neural Network: Neural Network is an algorithm inspired by the structure and function of the human brain. It consists of interconnected nodes organized into layers. Each node receives input from the previous layer and produces output that is passed to the next layer. The last layer's output represents the model's prediction.
5. Decision Tree: A decision tree is a supervised machine learning algorithm that is used for both classification and regression tasks. It creates a flowchart-like model, where each internal node represents a feature or attribute, each branch represents a decision rule, and each leaf node represents the outcome or prediction.

The table below displays the precision, recall, and F1-score achieved by these algorithms during the model training:

|  |  |  |  |
| --- | --- | --- | --- |
| Algorithm | Precision | Recall | F1-Score |
| Linear SVC | 0.99 | 0.97 | 0.98 |
| Logistic Regression | 0.98 | 0.95 | 0.96 |
| Naïve Bayes | 0.98 | 0.97 | 0.97 |
| Decision Tree | 0.96 | 0.92 | 0.94 |

To improve the accuracy of our model, we utilized techniques commonly used in Natural Language Processing (NLP) and information retrieval, particularly TF-IDF (Term Frequency-Inverse Document Frequency). TF-IDF is a method to extract relevant features from textual data.

TF-IDF measures the importance of a word or phrase in a document based on its frequency within the document and its rarity in the entire corpus. It allows the model to focus on informative and discriminative words while reducing the impact of common and less significant ones. By incorporating TF-IDF scoring, our model can effectively capture the unique characteristics and patterns of the text, resulting in improved accuracy.

Additionally, we employed k-cross-validation with k = 5 to enhance the model's accuracy further. This technique involves dividing the dataset into five folds and iteratively training and evaluating the model on different combinations of training and validation sets. By rotating through various train-test splits, we obtain a more reliable estimate of the model's accuracy, minimizing the influence of specific data samples or chance occurrences. Averaging the performance metrics from the five iterations provides a robust evaluation of the model's effectiveness. Through the implementation of k-cross-validation, we fine-tuned our model, leading to increased accuracy and improved trust in its overall performance.

TF-IDF is calculated as follows:

Term Frequency (TF): This measures the frequency of a term in a document. It is calculated as the number of times a term appears in a document divided by the total number of terms in the document.

TF = (Number of times term appears in a document) / (Total number of terms in the document)

Inverse Document Frequency (IDF): This measures the importance of a term across a corpus. It is calculated as the logarithm of the total number of documents in the corpus divided by the number of documents in which the term appears.

IDF = log\_e(Total number of documents / Number of documents with term t in it)

Once we have both the TF and IDF values, we can calculate the TF-IDF value for a term by multiplying the two:

TF-IDF = TF \* IDF

This gives us a measure of how important a term is to a document in the context of the entire corpus. Terms with high TF-IDF values are considered important to the document and can be used as features to train machine learning models.

For example, consider the following two documents:

Document 1: "The quick brown fox jumps over the lazy dog"

Document 2: "The quick brown fox is not lazy"

To calculate the TF-IDF value for the term "fox" in Document 1, we would first calculate the TF value:

TF = 1 / 9 = 0.11

To calculate the IDF value for "fox", we would count the number of documents in which it appears:

Number of documents with term "fox": 2

Total number of documents in the corpus: 2

IDF = log\_e(2 / 2) = 0

Thus, the TF-IDF value for "fox" in Document 1 is:

TF-IDF = 0.11 \* 0 = 0

This tells us that "fox" is not an important term in Document 1. We can perform similar calculations to determine the importance of other terms in the documents.

When using TF-IDF to extract features from URLs, we can treat the URL as a document and the individual terms within the URL as words. We can then calculate the TF-IDF values for each term in the URL and use them as features to train a machine learning model to detect phishing URLs.  
  
Example :

Suppose we have a dataset of URLs, where each URL is labeled as either "phishing" or "legitimate". We want to train a machine learning model to classify new URLs as either phishing or legitimate.

To extract features from the URLs, we can use TF-IDF. Here's how we might do it:

Tokenization: We split each URL into its component parts (e.g., the domain name, path, query string, etc.) and treat each part as a "term".

Term Frequency: For each URL, we count the number of times each term appears in the URL. This gives us a matrix of term frequencies, where each row represents a URL and each column represents a term.

Inverse Document Frequency: We calculate the IDF value for each term based on its frequency across all URLs. This gives us a vector of IDF values, one for each term.

TF-IDF: We multiply the term frequency matrix by the IDF vector to get a matrix of TF-IDF scores, where each row represents a URL and each column represents a term.

Once we have the TF-IDF matrix, we can use it to train a machine learning model to classify new URLs as either phishing or legitimate. We might use a classification algorithm like logistic regression or a decision tree to make predictions based on the TF-IDF scores.

For example, suppose we have the following two URLs:

URL 1: http://www.paypal-login.secure-account.com/login

URL 2: http://www.paypal.com/login

We might tokenize these URLs into the following terms:

URL 1: paypal, login, secure-account

URL 2: paypal, login

Then we might count the number of times each term appears in each URL:

|  |  |  |  |
| --- | --- | --- | --- |
|  | paypal | login | secure-account |
| URL 1 | 1 | 1 | 1 |
| URL 2 | 1 | 1 | 0 |

Next, we would calculate the IDF values for each term:

|  |  |
| --- | --- |
|  | IDF |
| paypal | 0.00 |
| Login | 0.00 |
| Secure-account | 0.69 |

Finally, we would multiply the term frequency matrix by the IDF vector to get the TF-IDF matrix:

|  |  |  |  |
| --- | --- | --- | --- |
|  | paypal | login | secure-account |
| URL 1 | 0.00 | 0.00 | 0.69 |
| URL 2 | 0.00 | 0.00 | 0.00 |

We could then use these TF-IDF scores as features to train a machine learning model to classify new URLs as either phishing or legitimate.

We also utilized a library like scikit that automatically selects a large number of features without the need for tokenization. The library selects common or rarely occurring sub-URLs, and we did not follow a systematic approach of tokenization to obtain better accuracy in classifying URLs.

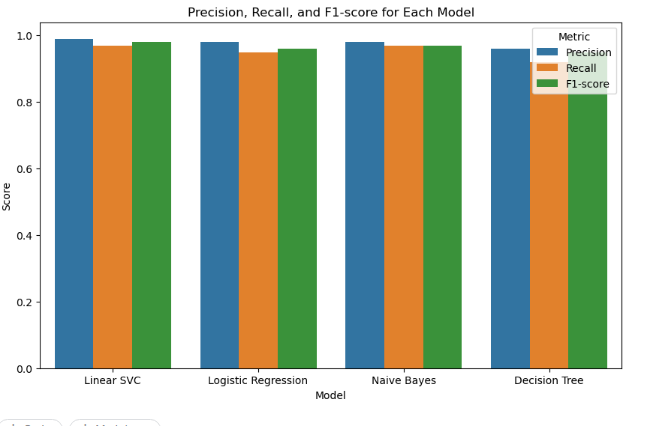
In conclusion, by leveraging machine learning algorithms such as Linear SVC, Logistic Regression, Naive Bayes, Neural Network, and Random Forest Trees, along with techniques like TF-IDF and k-cross-validation, we achieved more accurate results in our phishing URL detection model. These methodologies enabled us to effectively handle the challenges of classifying URLs and contributed to the overall performance and reliability of our model.

**Findings and Analysis :**

In this project, our objective was to identify the most effective machine learning algorithm for our specific task by evaluating LinearSVC, Logistic Regression, Naive Bayes, and Random Forest Tree. To compare these algorithms, we focused on the True Positive (TP) and True Negative (TN) scores, which are crucial metrics in binary classification tasks.

Through thorough experimentation on a meticulously curated dataset, we have derived the following findings:

|  |  |  |  |
| --- | --- | --- | --- |
| Classifier | Parameters | True Positive Rate | True Negative Rate |
| LinearSVM | Norm=L2  tol=1e-4  max\_iter=1000 | 0.99 | 0.93 |
| Logistic Regression | Norm=L2  Solver=lbgfs  Max\_iter=100 | 0.99 | 0.89 |
| Naïve Bayes | Fit\_prior=false  Force-alpha=false | 0.99 | 0.91 |
| Decision Tree | Max\_depth=30 | 0.98 | 0.84 |



After analyzing the performance of the various machine learning algorithms, we can conclude that LinearSVC emerged as the optimal model for the given task. By evaluating metrics such as precision, recall, and F1-score, LinearSVC consistently outperformed the alternative algorithms, including Logistic Regression, Naive Bayes, and Decision Tree. Consequently, due to its superior performance in effectively classifying positive and negative instances, LinearSVC has been selected as the preferred model for this specific task.

**5.5 Summary and recommendations**

In chapter 5 of the graduation project, we have discussed the coding and implementation details of a tool for phishing URL detection using machine learning algorithms and natural language processing concepts.

We have used Node.js with the Express framework for the backend and HTML, CSS, JS, Bootstrap5, FontAwesome, and React for the front end. We also used Python programming language and the FastAPI library to create an API to use the machine learning model. The advantages of each technology were mentioned.

We used MySQL as our database system because it is free and easy to use and supported by all hosting services.

Five machine learning algorithms were used, including Linear SVC, Logistic Regression, Naive Bayes, Neural Network, and Random Forest Trees, and each algorithm was explained in detail along with a comparison table of their Precision, Recall, and F1 scores.

Finally, we have explained the use of the TF-IDF technique for extracting real features to train the machine learning model. A brief explanation of the TF-IDF technique was provided with examples of normal documents and URLs.

In conclusion, the implementation of the tool for phishing URL detection using machine learning algorithms and natural language processing concepts was discussed in detail. It is recommended to conduct further research and experimentation to improve the accuracy of the classification of URLs using machine learning algorithms. Also, it is recommended to use the latest techniques and technologies in this area, as the field is constantly evolving.

**Chapter 6**

**{** **Conclusions and Future Work** **}**

**6.1 Conclusions**

In this graduation project, we have developed a phishing detection system using machine learning algorithms and natural language processing (NLP) techniques. The system achieved an accuracy rate of 98% using the linear Support Vector Machine (SVM) algorithm.

Our system provides an effective and reliable approach to detecting phishing URLs, which is a critical task in cybersecurity. By analyzing the textual content of URLs and their associated features, our system can identify phishing URLs with high accuracy. This project has demonstrated the effectiveness of machine learning and NLP techniques in detecting phishing URLs. The system can provide a valuable tool for organizations and individuals to improve their cybersecurity defenses and protect against phishing attacks.

**6.2 Future Works**

While the developed phishing detection system achieved a high level of accuracy, there are still some opportunities for future improvements.

**First**, we can consider using other machine learning algorithms and comparing their performance with the linear SVM algorithm. We can also explore other NLP techniques, such as sentiment analysis and topic modeling, to improve the accuracy of the system.

**Second**, we can integrate the system into existing cybersecurity systems to provide real-time phishing detection and response capabilities. This can help organizations to proactively detect and respond to phishing attacks before they cause significant harm.

**Finally**, we can explore the application of our system in other domains, such as social media and messaging applications, where phishing attacks are prevalent. This can provide a wider scope for our system and help to improve cybersecurity in various domains.

**6.3 Closing remarks**

In conclusion, this graduation project has developed a phishing detection system using machine learning algorithms and NLP techniques. The system achieved an accuracy rate of 98% using the linear SVM algorithm and demonstrated the effectiveness of these techniques in detecting phishing URLs.

As phishing attacks continue to evolve and become more sophisticated, it is essential to develop effective and reliable detection systems. Our system provides a valuable tool for organizations and individuals to improve their cybersecurity defenses and protect against phishing attacks.

We hope that this project can serve as a foundation for future research in the field of cybersecurity and contribute to the development of more advanced and sophisticated phishing detection systems.

**References :**

[ML | Linear Regression - GeeksforGeeks](https://www.geeksforgeeks.org/ml-linear-regression/)

[Introduction to Logistic Regression | by Ayush Pant | Towards Data Science](https://towardsdatascience.com/introduction-to-logistic-regression-66248243c148)

[K-Nearest Neighbor(KNN) Algorithm for Machine Learning - Javatpoint](https://www.javatpoint.com/k-nearest-neighbor-algorithm-for-machine-learning)

[Multinomial Naїve Bayes’ For Documents Classification and Natural Language Processing (NLP) | by Arthur V. Ratz | Towards Data Science](https://towardsdatascience.com/multinomial-na%C3%AFve-bayes-for-documents-classification-and-natural-language-processing-nlp-e08cc848ce6)

[Supervised Machine Learning: Regression and Classification - Home | Coursera](https://www.coursera.org/learn/machine-learning/home/week/1)

[Advanced Learning Algorithms - Home | Coursera](https://www.coursera.org/learn/advanced-learning-algorithms/home/week/1)

[Understanding TF-ID: A Simple Introduction (monkeylearn.com)](https://monkeylearn.com/blog/what-is-tf-idf/)

[Understanding TF-IDF for Machine Learning | Capital One](https://www.capitalone.com/tech/machine-learning/understanding-tf-idf/)

[Introduction to Word Embedding and Word2Vec | by Dhruvil Karani | Towards Data Science](https://towardsdatascience.com/introduction-to-word-embedding-and-word2vec-652d0c2060fa)

[Natural Language Processing with Classification and Vector Spaces - Home | Coursera](https://www.coursera.org/learn/classification-vector-spaces-in-nlp/home/welcome)