

WEB PHISHING DETECTION

1.INTRODUCTION

Nowadays Phishing has become a main area of concern for security researchers because it is not difficult to create a fake website which looks so close to a legitimate website. Experts can identify fake websites but not all the users can identify the fake website and such users become the victim of phishing attack. Main aim of the attacker is to steal banks account credentials. In United States businesses, there is a loss of US\$2billion (about \$6 per person in the US) per year because their clients become victim to phishing. In the 3rd Microsoft Computing Safer Index Report released in February 2014, it was estimated that the annual worldwide impact of phishing could be as high as \$5 billion (about \$15 per person in the US). Phishing attacks are becoming successful because of lack of user awareness. Since phishing attacks exploit the weaknesses found in users, it is very difficult to mitigate them, but it is very important to enhance phishing detection techniques.

The general method to detect phishing websites by updating blacklisted URLs, Internet Protocol (IP) to the antivirus database which is also known as "blacklist" method. To evade blacklists attackers uses creative techniques to fool users by modifying the URL to appear legitimate via obfuscation and many other simple techniques including fast flux, in which proxies are automatically generated to host the webpage; algorithmic generation of new URLs; etc. Major drawback of this method is that it cannot detect zero-hour phishing attack.

Heuristic based detection which includes characteristics that are found to exist in phishing attacks in reality and can detect zero-hour phishing attack, but the characteristics are not guaranteed to always exist in such attacks and false positive rate in detection is very high.

To overcome the drawbacks of blacklist and heuristics-based methods, many security researchers now focus on machine learning techniques. Machine learning technology consists of many algorithms which require past data to make a decision or prediction on future data. Using this technique, algorithms will analyse various blacklisted and legitimate URLs and their features to accurately detect phishing websites including zero- hour phishing websites.

1.1 Project Overview

In emerging technology industry which deeply influence today's security problems has given a non-ease of mind to some employer and home users. Occurrences that exploit human vulnerabilities have been on the upsurge in recent years .In the dimension

of new era there are many security systems being developed to ensure security is given the utmost priority and prevention to be taken from being hacked by those who are involved in cyber-criminal and essential prevention is also taken as high consideration in organization to ensure network security is not being breached. Cyber security employees are currently searching for trustworthy and steady detection techniques for phishing websites detection. Due to wide usage of internet to perform various activities such as online bill payment, banking transaction, online shopping, etc. Customer face numerous security threats like cybercrime. There are many cybercrimes that are extensively executed, for example spam, fraud, cyber terrorism and phishing. Among these phishing is known as the popular cybercrime today. Phishing has become one amongst the highest 3 most current forms of law-breaking in line with recent reports, and both frequency of events and user susceptibility has enlarged in recent years, more combination the danger of economic damage.

Phishing is a type of practice done on the Internet where individual data are obtained by illegal approaches. It supplies us to obtain sensitive information, for example, usernames, passwords, and positive identification points of interest, often for malignant reasons, by taking up the looks of an electronic correspondence. Phishing attack will be enforced in varied kinds like email phishing, web site phishing, spear phishing, Whaling, Tab off his guard, Evil twin phishing etc. Phishing is known as webpage violence. Phishing is often done by email spoofing or texting, and it typically guides users to enter points of interest at a fake web site which look and feel the same. It tries to handle the increasing range of phishing got to be met by clients in awareness and alternative efforts to ascertain protection numerous anti-phishing tools. A number of sites have currently created optional instruments for applications, like maps for redirection but clients ought to not utilize similar passwords anywhere on the net. [8] The primary key feature is to allow user to inquire whether visited websites are original or fake. This paper proposes a security tool called Detecting Phishing Website Using Machine Learning.

1.2 Purpose

Phishing attack is used to steal confidential information of a user. Fraud websites appear like genuine websites with the logo and graphics of genuine website. This project aims to detect fraud or phishing websites using machine learning techniques. Phishing websites are one of the internet security problems that target human vulnerabilities rather than software vulnerabilities. It can be described as the process of attracting online users to obtain sensitive information such as usernames and passwords. The objective of this project is to train machine learning models and

deep neural networks on the dataset created to predict phishing websites. Both phishing and legitimate URLs of websites are gathered to form a dataset and from them required URL and website content-based features are extracted. The performance level of each model is measured and compared.

2.LITERATURE SURVEY

Phishing Detection: Analysis of Visual Similarity Based Approaches

Abstract:

Phishing is one of the major problems faced by cyber-world and leads to financial losses for both industries and individuals. Detection of phishing attack with high accuracy has always been a challenging issue. At present, visual Similarities based techniques are very useful for detecting phishing websites efficiently. Phishing website looks very similar in appearance to its corresponding legitimate website to deceive users into believing that they are browsing the correct website. Visual Similarity based phishing detection techniques utilise the feature set like text content, text format, HTML tags, Cascading Style Sheet (CSS), image, and so forth, to make the decision. These approaches compare the suspicious website with the corresponding legitimate website by using various features and if the similarity is greater than the predefined threshold value then it is declared phishing. This paper presents a comprehensive analysis of phishing attacks, their exploitation, some of the recent visual Similarity based approaches for phishing detection, and its comparative study. Our survey provides a better understanding of the problem, current solution space, and scope of future research to deal with phishing attacks efficiently using visual Similarity based approaches.

1.Introduction

Phishing is a crime in which a perpetrator sends the fake e-mail, which appears to come from popular and trusted brand or organization, asking to input personal credential like bank password, username, phone number, address, credit card details, and so forth. The fake e-mails often look amazingly legitimate, and even the website where the Internet user is asked to input personal information also looks similar to legitimate one. Phishing messages propagate over e-mail, SMS, instant messengers, social networking sites, VoIP, and so forth, but e-mail is the popular way to perform this

attack and 65% of the total phishing attack is achieved by visiting the hyperlink attached to the e-mail. Moreover, spear phishing attack is becoming popular nowadays. Business e-mail compromise (BEC) is observed as a major Internet threat in 2015. In BEC, the intruder uses spear phishing methods to fool organizations and Internet persons. More sophisticated spear phishing attacks targeted particular individual or groups within the organization. Phishing is metaphorically similar to fishing in the water, but instead of trying to catch a fish, attackers try to steal consumer's personal information. When a user opens a fake webpage and enters the username and protected password, the credentials of the user are acquired by the attacker which can be used for malicious purposes. Phishing websites look very similar in appearance to their corresponding legitimate websites to attract large number of Internet users. Recent developments in phishing detection have led to the growth of numerous new visual Similarity based approaches. Visual Similarity based approaches compare the visual appearance of the suspicious website to its corresponding legitimate website by using various parameters. Due to different phases of phishing detection, this paper contains the following.

2. Background, History, and Statistics:

A phishing scam has attracted the attention of both academicians and corporate researchers as it is a serious privacy and web security threat. Phishing cannot be controlled by firewalls or any encryption software.

2.1. Brief History:

First phishing attack was observed on America online network systems (AOL) in the early 1990s where many fraudulent users registered on AOL website with fake credit card details. AOL passed these fake accounts with a simple validity test without verifying the legitimacy of the credit card. After activation of the fake account, attackers accessed the resources of America online system. At the time of billing, AOL determined that the accounts were fraudulent, and associated credit cards were also not valid; Therefore AOL ceased these accounts immediately. After this incident, AOL took measures to prevent this type of attack by verifying the authenticity of credit card and associated billing identity, which also enabled the attackers to change their way of obtaining AOL accounts. Instead of creating a fake account, attackers would steal the personal information of registered AOL user. Attackers contacted registered AOL users through instant messenger or e-mail and asked them to verify the password for security purposes. E-mail and instant messages appeared to come from an AOL employee. Many users provided their passwords and other personal information to the attackers.

The attackers then used the variously billed portions of America online website on behalf of a legitimate user. Moreover, an attacker no longer restricts themselves to masquerading America online website but actively masquerade a large number of financial and electronic commerce websites.

2.2. Statistics:

According to Internet world stats, total numbers of Internet users worldwide are 2.97 billion in 2014; that is, more than 38% of the world population uses Internet. Hackers take advantage of the insecure Internet system and can fool unaware users to fall for phishing scams. Phishing e-mail is used to defraud both individuals and financial organizations on the Internet. The Anti-Phishing Working Group (APWG) is an international consortium which is dedicated to promoting research, education, and law enforcement to eliminate online fraud and cybercrime. In 2012, total phishing attack increased by 160% over 2011, signifying a record year in phishing volumes. The total phishing attacks detected in 2013 were approximately 450000 and led to financial losses more than 5.9 billion dollars. Total attack increases by 1% in 2013 as compared to 2012. The total number of phishing attacks noticed in Q1 (first quarter) of 2014 was 125,215, a 10.7 percent increase over Q4 (fourth quarter) of 2013. More than 55% of phishing websites contain the name of the target site in some form to fool users and 99.4% of phishing websites use port 80. According to the APWG report in the first quarter of 2014, second highest number of phishing attacks ever recorded was between January and March 2014 and payment services are the most targeted industry. During the second half of 2014, 123,972 unique phishing attacks were observed. In the year 2011, total financial losses were 1.2 billion, and they rose to 5.9 billion dollars in 2013. The financial losses due to phishing attack in 2014 and 2015 were 4.5 and 4.6, respectively. The growth of phishing attacks from 2005 to 2015 .

2.3. Phishing Mechanism.

The fake website is the clone of targeted genuine website, and it always contains some input fields (e.g., text box). When the user submits his/her personal details, the information is transferred to the attacker. An attacker steals the credential of the innocent user by performing following steps: Construction of Phishing Site. In the first step attacker identifies the target as a well-known organization. Afterwards, attacker collects the detailed information about the organization by visiting their website. The attacker then uses this information to construct the fake website. URL Sending. In this step, attacker composes a bogus e-mail and sends it to the thousands of users. Attacker attached the URL of the fake website in the bogus e-mail. In the case of spear

phishing attack, an attacker sends the e-mail to selected users. An attacker can also spread the link of phishing website with the help of blogs, forum, and so forth. Stealing of the Credentials. When user clicks on attached URL, consequently, fake site is opened in the web browser. The fake website contains a fake login form which is used to take the credential of an innocent user. Furthermore, attacker can access the information filled by the user. Identity Theft. Attacker uses this credential of malicious purposes. For example, attacker purchases something by using credit card details of the user.

2.4. Taxonomy of Phishing Attack:

Attacker performed the phishing attack by utilising the technical subterfuge and social engineering techniques. In social engineering techniques, attackers carry out this attack by sending bogus e-mail. Attackers often convince recipients to respond using names of banks, credit card companies, e-retailers, and so forth. Technical subterfuge strategies install malware into user's system to steal credentials directly using Trojan and keylogger spyware. The malware also misaddresses users to fake websites or proxy servers. Attackers attached malware or embedded malicious links in the fraudulent emails and when the user opens the fraud hyperlink, malicious software is installed on the user's system, which collected the confidential information from the system and sent it to the attacker (e.g., keylogger software sends the details of every key hit by the user). Attackers may also get remote access to victim's computer and collect data whenever attackers want. In this paper, we focus on social engineering schemes, as it is the most popular way to steal victim's information by phishing.

Anti-phishing Technique:

Modus Operandi. A phishing scam starts with spreading bogus e-mail. After receiving an e-mail, Anti-phishing techniques start working, either by redirecting the phishing mail in the spam folder or by showing a warning when an online user clicks on the link of phishing URL

The following steps are involved in phishing lifecycle:

Step 1. Attacker creates the fake copy of a popular organization and sends the URL of fake website to the large number of Internet users using e-mail, blog, social networking sites, and so forth.

Step 2. In the case of fake e-mail, every e-mail is first to pass through the DNSbased blacklist filters. If the domain is found in the blacklist, then e-mail is blocked before it reached to SMTP mail server. There are also various solutions available which block the fake e-mail based on structural features of mail.

Step 3. If a fake e-mail bypasses the blacklist and features based solutions and if

the user opens attached link in the Email then some browser based blacklist techniques block the site at client side.

Step 4. Some other solutions like the heuristic and visual Similarities based approaches also blocked the webpage only when the browser requests for any suspicious webpage.

Step 5. If the phishing attack bypasses all the Solutions then it steals the credential of innocent users and sends it to the attacker. The attacker uses this information for financial or some other benefits.

3. Visual Similarity Based Phishing Detection and Filtering Approaches :

A user could become the victim of the phishing attack by looking the high visual resemblance of phishing website with the targeted legitimate site, such as page layouts, images, text content, font size, and font colour. The fake and genuine webpages of PayPal are shown in Figure 6, and both pages have same visual appearance but different URLs. It is not always necessary that the people carefully notice on URL and SSL (Secure Socket Layer) certificate of websites. If an attacker does not copy the visual appearance of targeted website well, then chances of inputting credentials by Internet users are very less.

An attacker fools the user by the following ways:

(1) Visual Appearance. The phishing website looks similar to its legitimate website. Attackers used to copy the HTML source code of genuine website to build the fake website.

(2) Address Bar. Attackers also cover the address or URL bar of website by script or image. The user would believe that they are inputting information on the right website.

(3) Embedded Objects. Attackers use embedded objects (images, scripts, etc.) to hide the textual content and HTML coding from the phishing detection approaches.

(4) Favicon Similarity. Favicon is an image icon associated with the particular website. An attacker may copy the favicon of targeted website. If the favicon shown in the address bar is other than the current website, then it is considered as a phishing attempt.

Dharmic et al. conducted a survey on various participants to identify whether a website is phishing or genuine. Participants were unable to identify 90% of phishing sites. Many participants wrongly judged the site on the basis of their text content and visual appearance. They also found that even an experienced user could also be fooled by the visual appearance of a fake website, and 23% of the users do not look at the address bar of a website. Therefore , we can say that if the appearance of a phishing

site is similar to its legitimate one and domain is different then also users can easily be trapped by the attackers.

6. Open Issues and Challenges:

Various types of Anti-phishing techniques based on visual similarity approach have been given in the literature. However, still there is no single technique that can detect all types of phishing attacks (i.e., zero-hour phishing attack, embedded objects, DNS poisoning, etc.). Day by day phishing attack is increasing continuously and becomes the most popular e-crime. Consistently, when researchers design a new technique to control phishing attack, attackers change their way to perform attack or exploit the vulnerability in the solution. Hence, there is the tight race between attackers and Anti-phishing developers. There are various issues which have to take care while designing a new anti phishing technique. The first problem is the zero-hour phishing attack. Most of the anti-phishing techniques compare the suspicious website from the pool of legitimate sites using feature set including URL, keyword, and visual appearances. These techniques required a large dataset and still fail to detect zero-hour phishing attack. If attacker designs a new webpage and its target (corresponding legitimate page) is not available in the dataset, then technique fails to detect new fake webpages (zero- hour attack). Liu et al. Presented a technique which can detect zero-hour phishing attack; However this technique depends on the TF-IDF algorithm and hyperlinks. Therefore, detection of zero-hour phishing attack with high accuracy is still an open challenge. The second issue is the language independence. Various text languages are worldwide used in the websites, and the ecommerce and banking websites also have different text languages in various countries for example, Amazon, eBay, and Antitank. The layout of ecommerce and banking sites is almost similar in different languages. Heuristics based phishing detection Techniques use the keywords, and they are language dependent. As we discussed, some of the visual Feature based techniques can detect this attack because they utilise the webpage features like the logo of the company, CSS Structure, DOM tree, and so forth. Such techniques only detect the attack if the layout of phishing website is similar to the real one. However, these techniques are unable to detect a new phishing attack (zero-hour) because they compare the current website with the stored database. The third issue is the embedded objects present in the webpage as attackers use images, JavaScript, and so forth, to bypass the anti-phishing system. As we discussed, image Processing based techniques can detect the embedded objects present in suspicious webpage because these techniques take the snapshot of the webpage and compare it with the corresponding legitimate webpage. But identifying the correct corresponding legitimate webpage is the

major problem in image Processing based solutions. Image Processing based approaches also consumed a lot of time to compare a suspicious website with the pool of websites. Therefore detection of phishing site which uses embedded objects is still an open challenge. The fourth issue is determining an appropriate threshold to take appropriate decision. The threshold is the matching score between two websites. As we discussed, attacker constructs a phishing website which looks similar to legitimate one. If the phishing website is partially copied (less than 50%) from the legitimate website, then none of the visual similarity based approach can detect it. Therefore, adjusting the appropriate threshold to detect a maximum number of phishing websites is a challenging task. If anti-phishing system increases the threshold then the false negative rate increases and if it decreases the threshold then false positive rate increases. A good anti-phishing system requires that both false negative and false positive rate should be as minimal as possible.

7. Conclusion:

Phishing is an appalling threat in the web security domain. In this attack, the user inputs his/her personal information to a fake website which looks like a legitimate one. We have presented a survey on phishing detection approaches based on visual similarity. This survey provides a better understanding of phishing website, various solution, and future scope in phishing detection. Many approaches are discussed in this paper for phishing detection; however most of the approaches still have limitations like accuracy, the countermeasure against new phishing websites, failing to detect embedded objects, and so forth. These approaches use various features of a webpage to detect phishing attacks, such as text similarity, font colour, font size, and images present in the webpage. Text based similarity approaches are relatively fast, but they are unable to detect phishing attack if the text is replaced with some image. Image processing based approaches have high accuracy rate while they are complex in nature and are time-consuming. Furthermore, most of the work is done offline. These involve data collection and profile-creation phases to be completed first. A comparative table is prepared for easy glancing at the advantages and drawbacks of the available approaches. No single technique is enough for adopting it for phishing detection purposes. Detection of phishing websites with high accuracy is still an open challenge for further research and development.

References

[1] M. Khonji, Y. Iraqi, and A. Jones, "Phishing detection: a literature survey," IEEE Communications Surveys & Tutorials, vol. 15, no. 4, pp. 2091–2121, 2013.

[2] R. Islam and J. Abawajy, "A multi-tier phishing detection and filtering approach," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 324–335, 2013.

[3] A. K. Jain and B. B. Gupta, "Comparative analysis of features based machine learning approaches for phishing detection," in *Proceedings of the 10th INDIACOM*, New Delhi, India, 2016.

[4] G. Weaver, A. Furr, and R. Norton, *Deception of Phishing: Studying the Techniques of Social Engineering by Analysing Modern-Day Phishing Attacks on Universities*, 2016.

[5] Kaspersky Lab, "Spam in January 2012 love, politics and sport," 2013, [http://www.kaspersky.com/about/news/spam/2012/Spam in January 2012 Love Politics and Sport](http://www.kaspersky.com/about/news/spam/2012/Spam%20in%20January%202012%20Love%20Politics%20and%20Sport).

[6] APWG Q1-Q3 Report, 2015, [http://docs.apwg.org/reports/lapwing trends report q1-q3 2015.pdf](http://docs.apwg.org/reports/lapwing_trends_report_q1-q3_2015.pdf).

[7] B. Parma r, "Protecting against spear-phishing," *Computer Fraud & Security*, vol. 2012, no. 1, pp. 8–11, 2012.

[8] W. Jingguo, T. Hereat, C. Rui, A. Vishwanath, and H. R. Rao, "Phishing susceptibility: an investigation into the processing of a targeted spear phishing email," *IEEE Transactions on Professional Communication*, vol. 55, no. 4, pp. 345– 362, 2012

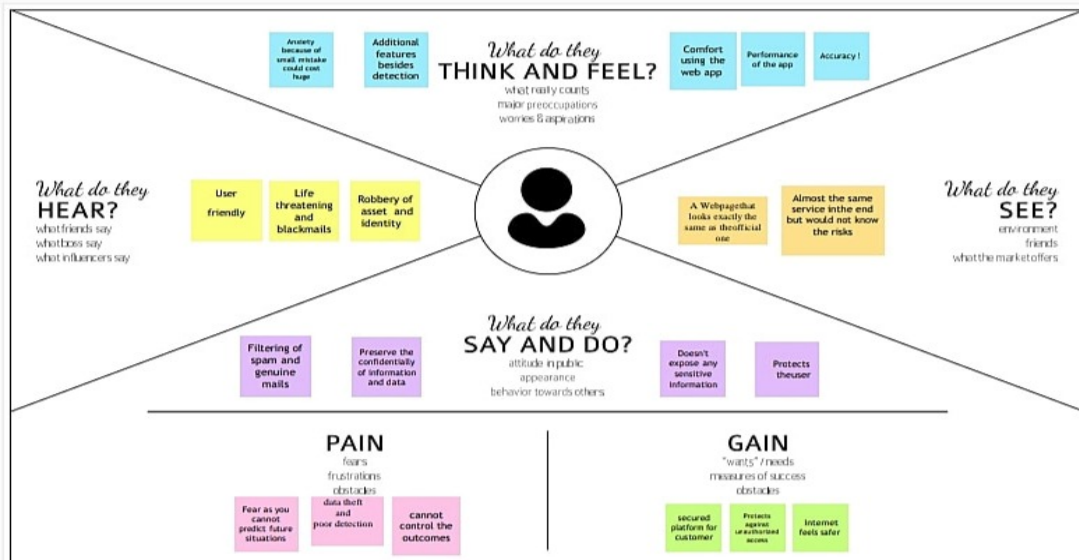
[9] T. N. Agnatic, N. A. Johnson, M. Jacobson, and F. Menczer, "Social phishing," *Communications of the ACM*, vol. 50, no. 10, pp. 94–100, 2007.

[10] C. H. Hsu, P. Wang, and S. Pu, "Identify fixed-path phishing attack by STC," in *Proceedings of the 8th Annual Collaboration, Electronic Messaging, Anti-Abuse and Spam Conference (CEAS '11)*, pp. 172–175, ACM, Perth, Australia, September 2011.

[11] N. A. G. Arachchilage and M. Cole, "Designing a mobile game for home computer users to protect against phishing attacks," <https://arxiv.org/abs/1602.03929>.

3.1 EMPATHY MAP CANVAS

Build empathy and keep your focus on the user by putting yourself in their shoes.



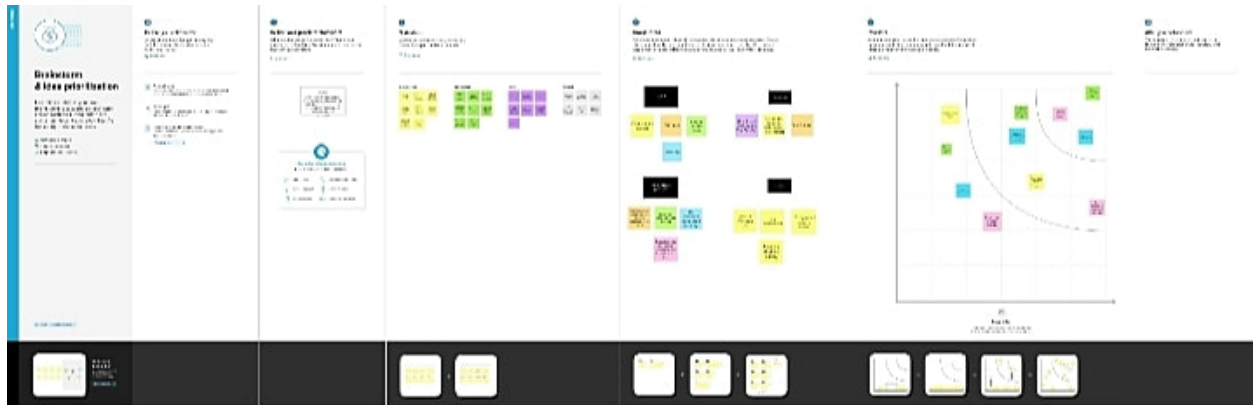
The empathy map represents a principal user and helps teams better understand their motivations, concerns, and user experience. Empathy mapping is a simple yet effective workshop that can be conducted with a variety of different users in mind, anywhere from stakeholders, individual use cases, or entire teams of people.

An empathy map canvas helps brands provide a better experience for users by helping teams understand the perspectives and mindset of their customers. Using a template to create an empathy map canvas reduces the preparation time and standardizes the process so you create empathy map canvases of similar quality. Empathy is important because it helps us understand how others are feeling so we can respond appropriately to the situation. It is typically associated with social behaviour and there is lots of research showing that greater empathy leads to more helpful behaviour.

3.2 IDEATION & BRAIN STORMING

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich number of creative solutions.

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.



3.3 Proposed solution











S.NO.	PARAMETER	DESCRIPTION
1.	Problem Statement (problem to be solved)	A particular challenge in this domain is that notorious hackers are constantly making new strategies to break into our defence measures. The drawback of the existing systems is detecting some minor false positive and false negative results. These disadvantages can be abolished by introducing a much-enhanced feature to feed to the machine learning algorithm that would result in much higher accuracy.
2.	Idea/ Solution Description	In order to detect and predict phishing website, we proposed an intelligent, flexible and effective system that is based on using classification Data mining algorithm. We implemented classification algorithm and techniques to extract the phishing data sets criteria (URL and Domain Identity, security and encryption) to classify their legitimacy. Then detect whether the website is a phishing site or not.
3.	Novelty/ Uniqueness	1. Using Machine Learning we developed the Web application as Web Phishing Detection 2. It checks the URL and no of users visited in that particular webpage or website and creates an alert to the user if the website is dangerous or not .
4.	Social Impact/ Customer Satisfaction	1. Since Data mining algorithm used in this system, it provides better performance as compared to other traditional classifications algorithms. 2. With the help of this system user can also purchase products online without any hesitation. The accuracy of phishing site identification is around 89% .
5.	Business Model (Financial Benefit)	1. This model helps Banking and Financial sectors from data loss and data attack which leads to zero financial loss externally. 2. In Business Organization they can use this tool to get rid from cyber attack and can implement how to improve the security when this attack occur next time.
6.	Scalability of Solution	1. Additional features such as reporting the phishing website can be made. 2. Also this web application can be made as a chrome extension so that users can securely surf through websites without any hesitation.

Proposed Solution Template:

Project Title: WEB PHISHING DETECTION

Project Design Phase-I -Solution Fit

Team ID: PNT2022TMID16684

Ready stage TR & EM	1. CUSTOMER SEGMENT(S)  Web users are mainly persons who purchase products through online payment or makeonline transactions.	6. CUSTOMER CONSTRAINTS  No breakdown of server connections and full permission to scan the transaction process.	5. AVAILABLE SOLUTIONS  Use multi-factor authentication to secure your accounts. Some accounts supply more security by needing two or more credentials to log in. Multi-factor authentication is one of the available solutions.
	2. JOBS-TO-BE-DONE/PROBLEMS  To keep the user's data and transactions protected from phishing sites and attackers.	9. PROBLEM ROOT CAUSE  Poor network authentication or use of traditional encryption techniques. Fooling customers by spoofing original websites.	7. BEHAVIOUR  Directly related: finds the user-friendly Web phishing detection application Indirectly related: permission to access the whole transaction process and server connectivity
	3. TRIGGERS  If web phishing detection is implemented successfully, it makes other users and shopping sites prefer our application for payments and transactions. 4. EMOTIONS: BEFORE / AFTER  Before: getting cheated by phishing websites. After: data confidentiality and secure transactions.	10. YOUR SOLUTION  1. Create a web application or web page to get the active URL as input. 2. Extract URL contents and test the model using a data mining algorithms and predict. If the website is hacked one sends an alert message and stores it in blacklisted URLs or else continues the transaction process. 3. Prediction is more accurate.	8. CHANNELS of BEHAVIOUR  Online: Inputs the active URL and extract the details for prediction. Offline: Stores the detected phishing sites to a Blacklisted URL.

3.4 PROBLEM SOLUTION FIT

SOLUTION ARCHITECTURE:

Solution architecture is a complex process – with many subprocesses – that bridges the gap between business problems and technological solutions. Its goals are to:

Find the best tech solution to solve existing business problems.

Describe the structure, characteristics, behaviour, and other aspects of the software to project stakeholders.

Define features, development phases, and solution requirements

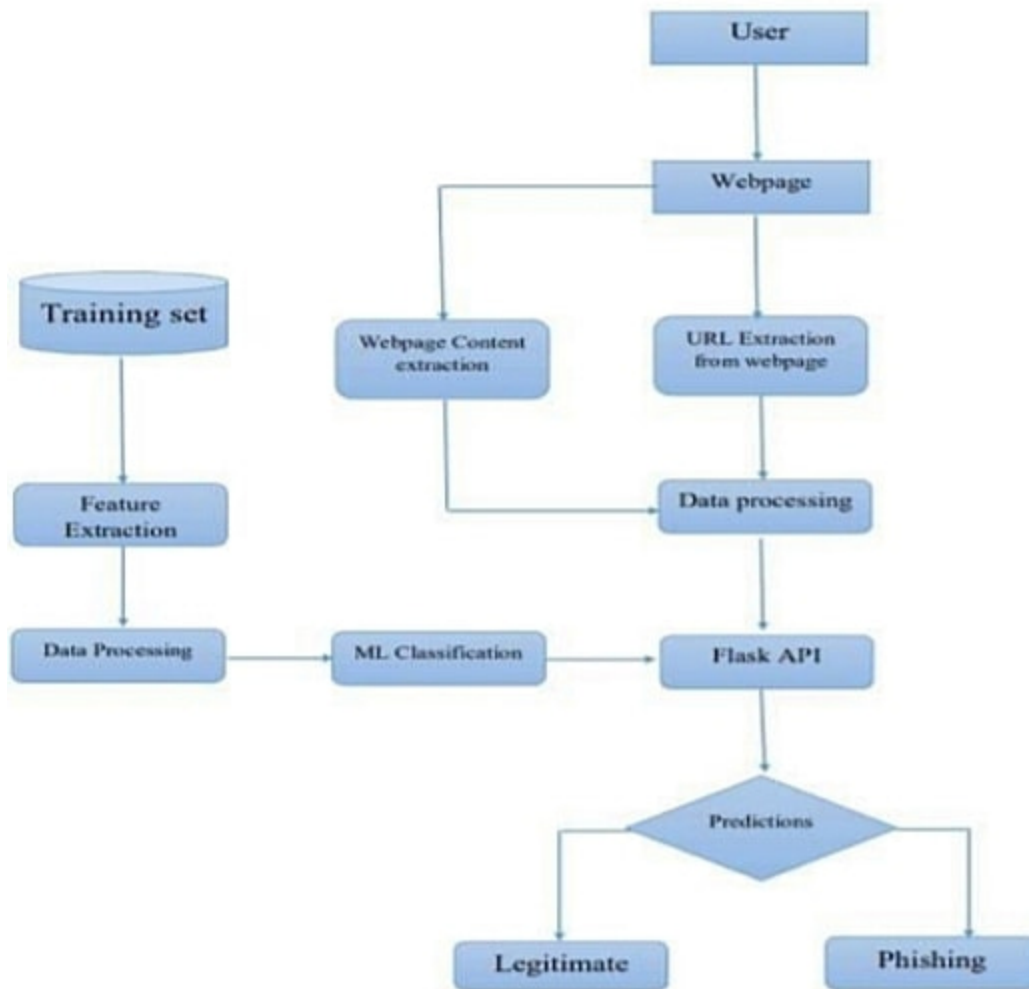
Provide specifications according to which the solution is defined, managed, and delivered.

Customer journey map:

STAGE	AWARENESS	CONSIDERATION	DECISION	SERVICE	LOYALTY
CUSTOMER ACTION	Watching some website-related awareness videos to get some awareness of using the website.	Comparison of secure and insecure websites.	The customer decides to avoid scam websites to prevent virus attacks on our computers and systems.	customer can contact customer care service and can provide proper documentation.	they can share their experience while using the websites
TOUCHPOINTS	Social media, Traditional media	Website Certifications	Website, Mobile app	Web Service	Review sites
CUSTOMER EXPERIENCE	Interested to get aware of phishing websites	Awareness of phishing websites	They plan to Detect Legal and Phishing websites to prevent the attacks made by the attackers	Provides Efficiency and Accuracy of the website ^s	Satisfied, Excited
KPIS	They check the No. of people getting aware of the web phishing attacks	They see the count of how many visits the new website has.	They check the Conversion rate of visiting the websites.	It provides Less time in producing the result for the website visitors.	Provides Customer satisfaction score.
BUSINESS GOALS	It provides an Increase in the awareness	It aims on detecting phishing websites with high accuracy.	It gives an Increase in the customer rate of visiting the websites.	It provides an increase in customer satisfaction.	It Generates some positive reviews from the customer
	of the phishing website attacks.				side of visiting the websites

DATA FLOW DIAGRAM:

The technique comprises of host based, page based and lexical feature extraction of collected websites. The primary step is the collection of phishing and begin websites.



Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIn
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Website Comparison	The model compares the websites using Blacklist and Whitelist approach
FR-4	Prediction	The model predicts the URL using a Machine Learning algorithms such as Logistic Regression, KNN
FR-5	Classifier	The model sends all output to the classifier and produces the final result.
FR-6	Events	This model needs the capability of retrieving and displaying accurate results for a website

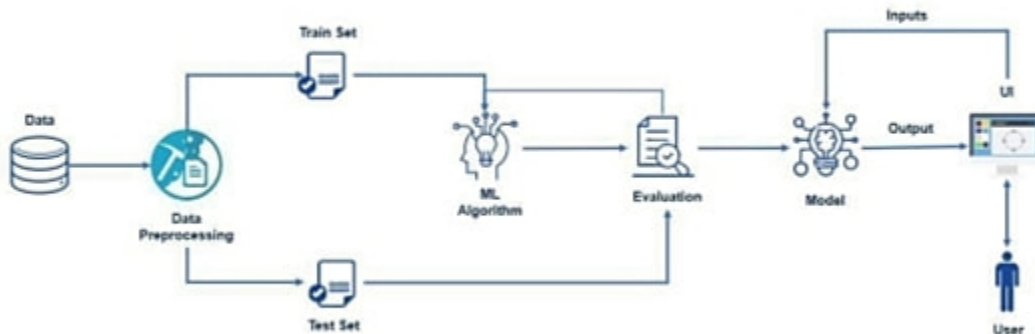
Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Users can have full access to the particular websites they using and must proceed some certain user-friendly websites so that it does not affect the data.
NFR-2	Security	To check whether the particular website is secure or not we can notify it by displaying an alert box while using the website.
NFR-3	Reliability	It must be a reliable source for the users while they using the websites.
NFR-4	Performance	The performance must be good while using the websites to which the users proceed the website.
NFR-5	Availability	The website availability must be valid for the users to access the resources.
NFR-6	Scalability	It must be able to handle an increase in users and loads without disrupting the end users.

TECHNOLOGY ARTITECTURE:

Technical Architecture:



6.PROJECT PLANNING & SCHEDULING

In Scrum Projects, Estimation is done by the entire team during Sprint Planning Meeting. The objective of the Estimation would be to consider the User Stories for the Sprint by Priority and by the Ability of the team to deliver during the Time Box of the Sprint. During sprint planning, we break the stories down into tasks, estimate those tasks, and compare the task estimates against our capacity. It's that, not points, that keep us from overcommitting in this sprint. No need to change the estimate.

How to run a sprint planning

- Examine team availability.
- Establish velocity for your team.
- Plan your sprint planning meeting.
- Start with the big picture.
- Present new updates, feedback, and issue.
- Confirm team velocity and capacity.
- Go over backlog items.
- Determine task ownership

Milestone and Activity List:

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Home page	USN-1	I can look through the homepage's functional resources as a user.	10	Low	Elayabharathi, Harish
Sprint-1		USN-2	As a user, I can get knowledge of the various aspects of web phishing and become informed about scams.	5	High	Harish, Hariprasath
Sprint-2	Final page	USN-3	I can use the end page's resources to learn more about how it works as a user.	15	Low	Elayabharathi, Harish, Deepan
Sprint-3	Prediction	USN-4	As a user, I can quickly guess the URL to determine whether a website is trustworthy or not.	10	High	Elayabharathi, Harish, Hariprasath, Deepan
	Dashboard					
Sprint-4	Chat	USN-5	As a user, I can provide feedback or contact the administrator for assistance.	10	High	Elayabharathi, Hariprasath, Harish
Sprint-1	Homepage	USN-6	As administrators we can create user interfaces and maintain the functionality of the website.	5	High	Hariprasath, Deepan
Sprint-2	Final page	USN-7	To make a website more user-friendly, we as administrators can reduce its complexity.	5	Medium	Elayabharathi, Harish
Sprint-3	Prediction	USN-8	As an administrator, you can utilise a variety of ML classifier models to do precise research for URL detection.	10	High	Elayabharathi, Harish, Hariprasath, Deepan
	Dashboard					
Sprint-4		USN-9	We can respond to the user feedback for website enhancement as admins.	10	Medium	Deepan, Elayabharathi

Project Tracker, Velocity & Burndown:

Project Tracker, Velocity & Burndown Chart(4 Marks)

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed(as on Planned End Date)	Sprint Release Date(Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	12 Nov 2022

Velocity:

Imagine we have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day)

$$AV = \frac{\text{sprint duration}}{\text{velocity}} = \frac{20}{10} = 2$$

We have a 6-day sprint duration, and the velocity of the team is 20 (points per sprint). So our team's average velocity (AV) per iteration unit (story points per day)

$$AV = (\text{Sprint Duration} / \text{Velocity}) = 20/6 = 3.33$$

Burndown Chart:

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.



7. CODING & SOLUTIONING:

Coding Solutions is a highly competitive job accelerator and talent refinement program that recruits and transitions college graduates with past programming experience or technical degrees into professional careers with Alabama companies and organizations at no cost to the graduates.

7.1 FEATURE 1

```
from flask import Flask, request, render_template
```

```
import numpy as np
```

```
import RFC
```

```
import pandas as pd
```

```
from sklearn import metrics
```

```
import requests
```

```
import json
```

```
import warnings
```

```
import pickle
```

```
warnings.filterwarnings('ignore')
```

```

from feature import FeatureExtraction

API_KEY = "ITkxxANDG6N3roYajqDgCD4qZOdsELITcTYseAecmJKu"

token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey":
API_KEY, "grant_type": 'urn:IBM:params:oauth:grant-type:apikey'})

mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

file = open("pickle/model.pkl", "rb")

RFC = pickle.load(file)

file.close()

app = Flask(__name__)

@app.route("/", methods=["GET", "POST"])

def index():

    if request.method == "POST":

        url = request.form["url"]

        obj = FeatureExtraction(url)

        x = np.array(obj.getFeaturesList()).reshape(1,30)

        y_pred =rfc.predict(x)[0]

        #1 is safe

        #-1 is unsafe

```

```

y_pro_phishing = rfc.predict_proba(x)[0,0]

y_pro_non_phishing = rfc.predict_proba(x)[0,1]

# if(y_pred ==1 ):

pred = "It is {0:.2f} % safe to go ".format(y_pro_phishing*100)

return render_template('index.html',xx =round(y_pro_non_phishing,2),url=url )


return render_template("index.html", xx =-1)

payload_scoring = {"input_data": [{"field": ["PrefixSuffix-",

        "SubDomains",

        "HTTPS",

        "AnchorURL",

        "WebsiteTraffic"], "values": [1, -1]]}]

response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/f44a67e9-ebef-4ab8-abf0-
91b5e1307e64/predictions?version=2022-11-13', json=payload_scoring,

headers={'Authorization': 'Bearer ' + mltoken})

print("Scoring response")

print(response_scoring.json())


if __name__ == "__main__":

    app.run(debug=True)

```

7.2 FEATURE 2

```
import ipaddress

import re

import socket

import time

import urllib.request

from datetime import date, datetime

from urllib.parse import urlparse

import domain

import requests

import response

import whois


from bs4 import BeautifulSoup

from dateutil.parser import parse as date_parse

from googlesearch import search

class FeatureExtraction:

    features = []

    def __init__(self,url):

        self.features = []

        self.url = url

        self.domain = ""

        self.whois_response = ""

        self.urlparse = ""

        self.response = ""
```

```
self.soup = ""  
  
url=""  
  
try:  
  
self.response = requests.get(url)  
  
self.soup = BeautifulSoup(response.text, 'html.parser')  
  
except:  
  
pass
```

```
  
try:  
  
self.urlparse = urlparse(url)  
  
self.domain = self.urlparse.netloc  
  
except:  
  
pass  
  
try:  
  
self.whois_response = whois.whois(self.domain)  
  
except:  
  
pass
```

```
  
  
self.features.append(self.UsingIp())  
  
self.features.append(self.longUrl())  
  
self.features.append(self.shortUrl())  
  
self.features.append(self.symbol())  
  
self.features.append(self.redirecting())  
  
self.features.append(self.prefixSuffix())  
  
self.features.append(self.SubDomains())
```

```
self.features.append(self.Hppts())

self.features.append(self.DomainRegLen())

self.features.append(self.Favicon())

self.features.append(self.NonStdPort())

self.features.append(self.HTTPSDomainURL())

self.features.append(self.RequestURL())

self.features.append(self.AnchorURL())

self.features.append(self.LinksInScriptTags())

self.features.append(self.ServerFormHandler())

self.features.append(self.InfoEmail())

self.features.append(self.AbnormalURL())

self.features.append(self.WebsiteForwarding())

self.features.append(self.StatusBarCust())

self.features.append(self.DisableRightClick())

self.features.append(self.UsingPopupWindow())

self.features.append(self.IframeRedirection())

self.features.append(self.AgeofDomain())

self.features.append(self.DNSRecording())

self.features.append(self.WebsiteTraffic())

self.features.append(self.PageRank())

self.features.append(self.GoogleIndex())

self.features.append(self.LinksPointingToPage())


self.features.append(self.StatsReport())
```



```
# 1.UsingIp
```

```
def UsingIp(self):
```

```
try:
```

```
ipaddress.ip_address(self.url)
```

```
return -1
```

```
except:
```

```
return 1
```

```
# 2.longUrl
```

```
def longUrl(self):
```

```
if len(self.url) < 54:
```

```
return 1
```

```
if len(self.url) >= 54 and len(self.url) <= 75:
```

```
return 0
```

```
return -1
```

```
# 3.shortUrl
```

```
def shortUrl(self):
```

```
match=re.search('bit\..ly|goo\..gl|shorte\..st|go2l\..ink|x\..co|ow\..ly|t\..co|tinyurl|tr\..im|is\..gd|cli\..gs|'
```

```
'yfrog\..com|migre\..me|ff\..im|tiny\..cc|url4\..eu|twit\..ac|su\..pr|twurl\..nl|snipurl\..com|'
```

```
'short\..to|BudURL\..com|ping\..fm|post\..ly|Just\..as|bkite\..com|snipr\..com|fic\..kr|loopt\..us|'
```

```
'doiop\..com|short\..ie|kl\..am|wp\..me|rubyurl\..com|om\..ly|to\..ly|bit\..do|t\..co|lnkd\..in|'
```

```
'db\..tt|qr\..ae|adf\..ly|goo\..gl|bitly\..com|cur\..lv|tinyurl\..com|ow\..ly|bit\..ly|ity\..im|'
```

```
'q\..gs|is\..gd|po\..st|bc\..vc|twitthis\..com|u\..to|j\..mp|buzurl\..com|cutt\..us|u\..bb|yourls\..org|'
```

```
'x\..co|prettylinkpro\..com|scrnch\..me|filoops\..info|vzturl\..com|qr\..net|1url\..com|tweez\..me|v\..gd|tr  
\..im|link\..zip\..net', self.url)
```

```
if match:
```

```
    return -1
```

```
    return 1
```

1. 4.Symb

```
    ol@ def
```

```
        symbol
```

```
        (self):
```

```
if re.findall("@",self.url):
```

```
    return -1
```

```
    return 1
```

2. 5.Redirecting//

```
def redirecting(self):
```

```
if self.url.rfind('/')>6:
```

```
    return -1
```

```
    return 1
```

6.prefixSuffix

```
def prefixSuffix(self):
```

```
try:
```

```
    match = re.findall('\-', self.domain)
```

```
if match:
```

```
    return -1
```

```
return 1
```

```
except:
```

```
return -1
```

```
# 7.SubDomains
```

```
def SubDomains(self):
```

```
dot_count = len(re.findall("\.", self.url))
```

```
if dot_count == 1:
```

```
return 1
```

```
elif dot_count == 2:
```

```
return 0
```

```
return -1
```

```
# 8.HTTPS
```

```
def Hppts(self):
```

```
try:
```

```
https = self.urlparse.scheme
```

```
if 'https' in https:
```

```
return 1
```

```
return -1
```

```
except:
```

```
return 1
```

```
# 9.DomainRegLen
```

```

def DomainRegLen(self):

    try:

        expiration_date = self.whois_response.expiration_date
        creation_date = self.whois_response.creation_date
        try:
            if(len(expiration_date)):
                expiration_date = expiration_date[0]

        except:

            pass

        try:
            if(len(creation_date)):
                creation_date = creation_date[0]

        except:

            pass


    age = (expiration_date.year-creation_date.year)*12+ (expiration_date.month-creation_date.month)

    if age >=12:

        return 1

    return -1

    except:

    return -1

```

1. 10.

Favicon

def

```

        Favicon
        (self):
        try:
for head in self.soup.find_all('head'):

for head.link in self.soup.find_all('link', href=True):

dots = [x.start(0) for x in re.finditer('\.', head.link['href'])]

if self.url in head.link['href'] or len(dots) == 1 or domain in head.link['href']:
return 1

return -1
except:
return -1

```

2. 11. NonStdPort

```

def NonStdPort(self):
try:
port = self.domain.split(":")
if len(port)>1:
return -1
return 1
except:
return -1

```

1. 12.

```

HTTPSDomainU
RL def
HTTPSDomainUR

```

```

        L(self): try:
if 'https' in self.domain:
return -1

return 1
except:
return -1

```

2. 13. RequestURL

```

def RequestURL(self):

try:

for img in self.soup.find_all('img', src=True):

dots = [x.start(0) for x in re.finditer('\.', img['src'])]

if self.url in img['src'] or self.domain in img['src'] or len(dots) == 1:

success = success + 1

i = i+1


for audio in self.soup.find_all('audio', src=True):

dots = [x.start(0) for x in re.finditer('\.', audio['src'])]

if self.url in audio['src'] or self.domain in audio['src'] or len(dots) == 1:

success = success + 1

i = i+1


for embed in self.soup.find_all('embed', src=True):

dots = [x.start(0) for x in re.finditer('\.', embed['src'])]

if self.url in embed['src'] or self.domain in embed['src'] or len(dots) == 1:

```

```
success = success + 1
```

```
i = i+1
```

```
for iframe in self.soup.find_all('iframe', src=True):
```

```
    dots = [x.start(0) for x in re.finditer('\.', iframe['src'])]
```

```
    if self.url in iframe['src'] or self.domain in iframe['src'] or len(dots) == 1:
```

```
        success = success + 1
```

```
    i = i+1
```

```
try:
```

```
    percentage = success/float(i) * 100
```

```
    if percentage < 22.0:
```

```
        return 1
```

```
    elif((percentage >= 22.0) and (percentage < 61.0)):
```

```
        return 0
```

```
    else:
```

```
        return -1
```

```
except:
```

```
    return 0
```

```
except:
```

```
    return -1
```

1. 14.

AnchorURL

```

def
    AnchorURL
    (self): try:
i,unsafe = 0,0

for a in self.soup.find_all('a', href=True):

if "#" in a['href'] or "javascript" in a['href'].lower() or "mailto" in a['href'].lower() or not (
    urllib.request.urlopen in a['href'] or self.domain in a['href']):

    unsafe = unsafe + 1

i = i + 1

try:

percentage = unsafe / float(i) * 100

if percentage < 31.0:

    return 1

elif ((percentage >= 31.0) and (percentage < 67.0)):

    return 0

else:

    return -1

except:

    return -1

except:

    return 1

```



```

        LinksInScriptTag
        gs def
        LinksInScriptTag
        s(self): try:
i,success = 0,0

for link in self.soup.find_all('link', href=True):

dots = [x.start(0) for x in re.finditer('\.', link['href'])]

if self.url in link['href'] or self.domain in link['href'] or len(dots) == 1:

success = success + 1

i = i+1

for script in self.soup.find_all('script', src=True):

dots = [x.start(0) for x in re.finditer('\.', script['src'])]

if self.url in script['src'] or self.domain in script['src'] or len(dots) == 1:

success = success + 1

i = i+1

try:

percentage = success / float(i) * 100

if percentage < 17.0:

return 1

elif((percentage >= 17.0) and (percentage < 81.0)):

return 0

else:

return -1

```

```
except:
```

```
return 0
```

```
except:
```

```
return -1
```

1. 16.

```
ServerFormHandl
```

```
er def
```

```
ServerFormHandle
```

```
r(self): try:
```

```
if len(self.soup.find_all('form', action=True))==0:
```

```
return 1
```

```
else :
```

```
for form in self.soup.find_all('form', action=True):
```

```
if form['action'] == "" or form['action'] == "about:blank":
```

```
return -1
```

```
elif self.url not in form['action'] and self.domain not in form['action']:
```

```
return 0
```

```
else:
```

```
return 1
```

```
except:
```

```
return -1
```

1. 17.

```
InfoEmail
```

```

def
    InfoEmail
    (self): try:
if re.findall(r"[mail\\(|mailto:?}", self.soap):
return -1

else:
return 1
except:
return -1

```

2. 18. AbnormalURL

```

def AbnormalURL(self):

try:

if self.response.text == self.whois_response:

return 1

else:

return -1

except:

return -1

```

1. 19.

```

WebsiteForwarding
def
    WebsiteForwarding
    (self): try:

if len(self.response.history) <= 1:
return 1

```

```
elif len(self.response.history) <= 4:
```

```
    return 0
```

```
else:
```

```
    return -1
```

```
except:
```

```
    return -1
```

1. 20.

```
    StatusBarCu
```

```
    st def
```

```
    StatusBarCus
```

```
    t(self): try:
```

```
if re.findall("<script>.+onmouseover.+</script>", self.response.text):
```

```
    return 1
```

```
else:
```

```
    return -1
```

```
except:
```

```
    return -1
```

2. 21. DisableRightClick

```
def DisableRightClick(self):
```

```
    try:
```

```
if re.findall(r"event.button ?== ?2", self.response.text):
```

```
    return 1
```

```
else:
```

```
    return -1
```

```
except:
```

```
return -1
```

```
1. 22.
```

```
UsingPopupWind
```

```
ow def
```

```
UsingPopupWindo
```

```
w(self): try:
```

```
if re.findall(r"alert\\(", self.response.text):
```

```
return 1
```

```
else:
```

```
return -1
```

```
except:
```

```
return -1
```

```
1. 23.
```

```
IframeRedirection
```

```
def
```

```
IframeRedirection
```

```
(self): try:
```

```
if re.findall(r"<iframe>|<frameBorder>", self.response.text):
```

```
return 1
```

```
else:
```

```
return -1
```

```
except:
```

```
return -1
```

2. 24. AgeofDomain

```
def AgeofDomain(self):  
  
    try:  
  
        creation_date = self.whois_response.creation_date  
  
        try:  
  
            if(len(creation_date)):   
  
                creation_date = creation_date[0]  
  
        except:  
  
            pass  
  
        today = date.today()  
  
        age = (today.year-creation_date.year)*12+(today.month-creation_date.month)  
  
        if age >=6:  
  
  
  
  
  
  
  
  
  
        return 1  
  
        return -1  
  
    except:  
  
        return -1
```

1. 25.

```
    DNSRecording  
    def  
    DNSRecording  
    (self): try:  
  
creation_date = self.whois_response.creation_date
```

```

try:
if(len(creation_date)):

creation_date = creation_date[0]
except:
pass

today = date.today()

age = (today.year-creation_date.year)*12+(today.month-creation_date.month)

if age >=6:

return 1

return -1

except:

return -1

```

1. 26.

```

WebsiteTraffic
def
WebsiteTraffic
(self): try:

rank = BeautifulSoup(urllib.request.urlopen("http://data.alexa.com/data?cli=10&dat=s&url=" +
urllib.request.urlopen().read(), "xml").find("REACH")['RANK']

if (int(rank) < 100000):

return 1

return 0

except :

return -1

```

1. 27.

```
PageRank
def
PageRank
(self): try:

prank_checker_response = requests.post("https://www.checkpagerank.net/index.php", {"name":
self.domain})

global_rank = int(re.findall(r"Global Rank: ([0-9]+)", prank_checker_response.text)[0])

if global_rank > 0 and global_rank < 100000:

return 1

return -1

except:

return -1
```

1. 28.

```
GoogleIndex
def
GoogleIndex
(self): try:

site = search(self.url, 5)

if site:

return 1

else:

return -1

except:

return 1
```


29. LinksPointingToPage

```
def LinksPointingToPage(self):  
  
    try:  
  
        number_of_links = len(re.findall(r"<a href=", self.response.text))  
  
        if number_of_links == 0:  
  
            return 1  
  
        elif number_of_links <= 2:  
  
            return 0  
  
        else:  
  
            return -1  
  
    except:  
  
        return -1
```

1. 30.

```
    StatsReport  
    def  
    StatsReport  
    (self): try:  
  
url_match = re.search(  
  
    'at\.ua|usa\.cc|baltazarpresentes\.com\.br|pe\.hu|esy\.es|hol\.es|sweddy\.com|myjino\.ru|96\.lt|ow  
    \.ly', urllib.request.urlopen)  
  
    ip_address = socket.gethostbyname(self.domain)  
  
    ip_match=re.search('146\.112\.61\.108|213\.174\.157\.151|121\.50\.168\.88|192\.185\.217\.116|78\  
    .46\.211\.158|181\.174\.165\.13|46\.242\.145\.103|121\.50\.168\.40|83\.125\.22\.219|46\.242\.145
```

```
\.98|'
```

```
'107\.151\.148\.44|107\.151\.148\.107|64\.70\.19\.203|199\.184\.144\.27|107\.151\.148\.108|107\.151\.148\.109|119\.28\.52\.61|54\.83\.43\.69|52\.69\.166\.231|216\.58\.192\.225|'
```

```
'118\.184\.25\.86|67\.208\.74\.71|23\.253\.126\.58|104\.239\.157\.210|175\.126\.123\.219|141\.8\.224\.221|10\.10\.10\.10|43\.229\.108\.32|103\.232\.215\.140|69\.172\.201\.153|'
```

```
'216\.218\.185\.162|54\.225\.104\.146|103\.243\.24\.98|199\.59\.243\.120|31\.170\.160\.61|213\.19\.128\.77|62\.113\.226\.131|208\.100\.26\.234|195\.16\.127\.102|195\.16\.127\.157|'
```

```
'34\.196\.13\.28|103\.224\.212\.222|172\.217\.4\.225|54\.72\.9\.51|192\.64\.147\.141|198\.200\.56\.183|23\.253\.164\.103|52\.48\.191\.26|52\.214\.197\.72|87\.98\.255\.18|209\.99\.17\.27|'
```

```
'216\.38\.62\.18|104\.130\.124\.96|47\.89\.58\.141|78\.46\.211\.158|54\.86\.225\.156|54\.82\.156\.19|37\.157\.192\.102|204\.11\.56\.48|110\.34\.231\.42', ip_address)
```

```
if url_match:
```

```
    return -1
```

```
elif ip_match:
```

```
    return -1
```

```
    return 1
```

```
except:
```

```
    return 1
```

```
def getFeaturesList(self):
```

```
    return self.features
```

8.TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product it is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

8.2 USER ACCEPTANCE TESTING

8.2.1 UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

8.2.2 INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfactory, as shown by successful unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

8.2.3 VALIDATION TESTING

An engineering validation test (EVT) is performed on first engineering prototypes, to ensure that the basic unit performs to design goals and specifications. It is important in

Identifying design problems and solving them as early in the design cycle as possible, is the key to keeping projects on time and within budget. Too often, product design and performance problems are not detected until late in the product development cycle when the product is ready to be shipped. The old adage holds true: It costs a penny to make a change in engineering, a dime in production and a dollar after a product is in the field. Verification is a Quality control process that is used to evaluate whether or not a product, service, or system complies with regulations, specifications, or conditions imposed at the start of a development phase. Verification can be in development, scale up, or production. This is often an internal process. Validation is a Quality assurance process of establishing evidence that provides a high degree of assurance that a product, service, or system accomplishes its intended requirements.

8.2.4 SYSTEM TESTING

System testing of software or hardware is testing conducted on a complete, integrated system to evaluate the system's compliance with its specified requirements. System testing falls within the scope of black box testing, and as such, should require no knowledge of the inner design of the code or logic. As a rule, system testing takes, as its input, all of the "integrated" software components that have successfully passed integration testing and also the software system itself integrated with any applicable hardware system(s). System testing is a more limited type of testing; it seeks to detect defects both within the "inter-assemblages" and also within the system as a whole. System testing is performed on the entire system in the context of a Functional Requirement Specification(s) (FRS) and/or a System Requirement Specification (SRS). System testing tests not only the design, but also the behavior and even

the believed expectations of the customer. It is also intended to test up to and beyond the bounds defined in the software/hardware requirements specification(s).

10. ADVANTAGES & DISADVANTAGES

ADVANTAGES

1. This system can be used by many E-commerce or other websites in order to have good customer relationship.
2. User can make online payment securely.
3. Data mining algorithm used in this system provides better performance as compared to other traditional classifications algorithms.
4. With the help of this system user can also purchase products online without any hesitation.
5. A mailbox-level anti-phishing solution offers an additional layer of protection by analyzing account information and understanding users' communication habits.
6. This delivers an enhanced level of phishing protection to detect attacks faster, alert users and remediate threats as quickly as possible.
7. Alert users and remediate threats as quickly as possible at detection
8. private information such as usernames, passwords, and credit card details, by way of impersonating a legitimate entity.
9. It will lead to information disclosure and property damage.

DISADVANTAGES

1. If Internet connection fails, this system won't work.
2. All websites related data will be stored in one place.

The problem with phishing is that attackers constantly look for new and creative ways to fool users into believing their actions involve a legitimate website or email. Phishers have become more skilled at forging websites to appear identical to the expected location, even including logos and graphics in the phishing.

11.CONCLUSION

This paper aims to enhance detection method to detect phishing websites using machine learning technology. We achieved 97.14% detection accuracy using random forest algorithm with lowest false positive rate. Also result shows that classifiers give better performance when we used more data as training data. In future hybrid technology will be implemented to detect phishing websites more accurately, for which

random forest algorithm of machine learning technology and blacklist method will be used. we get very good performance in classifiers namely, Random Forest computation duration and accuracy. The main idea behind ensemble algorithms is to combine several weak learners into a stronger one, this is perhaps the primary reason why machine learning is used in practice for most of the classification problems

12.FUTURE SCOPE

As we have already a real time implementation in our project the scope for future work for our project would be creating a GUI or web extension which would help our user if he accesses any phishing websites by any chance. The efficient of our product can be increased drastically provided your given access to the current fishing website data collection. As cyber-crime is a very prominent in our generation. The scope of future work for this project is perennial.

13.APPENDIX

13.1 Source code

Style.css

```
*,
*::after,
*::before {
    margin: 0;
    padding: 0;
    box-sizing: inherit;
    font-size: 62,5%;
}
body {
    padding: 10% 5%;
    background: rgb(25, 169, 185);
```

```
justify-content: center;

align-items: center;

height: 100vh;

color: rgb(81, 12, 12);
}

form__label {

font-family: 'Roboto', sans-serif;

font-size: 1.2rem;

margin-left: 2rem;

margin-top: 0.7rem;

display: block;

transition: all 0.3s;

transform: translateY(0rem);
}

.form__input {

top: -24px;

font-family: 'Roboto', sans-serif;

color: #333;

font-size: 1.2rem;

padding: 1.5rem 2rem;

border-radius: 0.2rem;
```

```
background-color: rgb(123, 140, 164);  
  
border: none;  
  
width: 75%;  
  
display: block;  
  
border-bottom: 0.3rem solid transparent;  
  
transition: all 0.3s;  
  
}
```

```
.form__input:placeholder-shown + .form__label {  
  opacity: 0;  
  visibility: hidden;  
  -webkit-transform: translateY(+4rem);  
  transform: translateY(+4rem);  
}
```

```
.button {  
  appearance: button;  
  background-color: transparent;  
  background-image: linear-gradient(to bottom, rgb(255, 255, 255), #e4dbf8);  
  border: 0 solid #e6eaeb;
```

border-radius: .5rem;

box-sizing: border-box;

color: #482307;

column-gap: 1rem;

cursor: pointer;

display: flex;

font-family: ui-sans-serif,system-ui,-apple-system,system-ui,"Segoe UI",Roboto,"Helvetica Neue",Arial,"Noto Sans",sans-serif,"Apple Color Emoji","Segoe UI Emoji","Segoe UI Symbol","Noto Color Emoji";

font-size: 100%;

font-weight: 700;

line-height: 24px;

margin: 0;

outline: 2px solid transparent;

padding: 1rem 1.5rem;

text-align: center;

text-transform: none;

transition: all .1s cubic-bezier(.4, 0, .2, 1);

user-select: none;

-webkit-user-select: none;

touch-action: manipulation;

box-shadow: -6px 8px 10px rgba(20, 170, 25, 0.1),0px 2px 2px rgba(9, 187, 33,


```
0.2);  
}
```

```
.button:active {  
    background-color: #27aca9;  
  
    box-shadow: -1px 2px 5px rgba(81,41,10,0.15),0px 1px 1px rgba(81,41,10,0.15);  
  
    transform: translateY(0.125rem);  
}
```

```
.button:focus {  
  
    box-shadow: rgba(72, 35, 7, .46) 0 0 0 4px, -6px 8px 10px rgba(81,41,10,0.1), 0px  
    2px 2px rgba(81,41,10,0.2);  
}
```

```
.main-body{  
  
    display: flex;  
  
    flex-direction: row;  
  
    width: 75%;  
  
    justify-content:space-around;  
}
```

```
.button1{

  appearance: button;

  background-color: transparent;

  background-image: linear-gradient(to bottom, rgb(160, 245, 174), #37ee65);

  border: 0 solid #e5e7eb;

  border-radius: .5rem;

  box-sizing: border-box;

  color: #482307;

  column-gap: 1rem;

  cursor: pointer;

  display: flex;

  font-family: ui-sans-serif,system-ui,-apple-system,system-ui,"Segoe UI",Roboto,"Helvetica Neue",Arial,"Noto Sans",sans-serif,"Apple Color Emoji","Segoe UI Emoji","Segoe UI Symbol","Noto Color Emoji";

  font-size: 100%;

  font-weight: 700;

  line-height: 24px;

  margin: 0;

  outline: 2px solid transparent;

  padding: 1rem 1.5rem;

  text-align: center;

  text-transform: none;
```

```
transition: all .1s cubic-bezier(.4, 0, .2, 1);

user-select: none;

-webkit-user-select: none;

touch-action: manipulation;

box-shadow: -6px 8px 10px rgba(81,41,10,0.1),0px 2px 2px rgba(81,41,10,0.2);

display: none;

}
```

```
.button2{

  appearance: button;

  background-color: transparent;

  background-image: linear-gradient(to bottom, rgb(252, 162, 162), #f51707);

  border: 0 solid #e5e7eb;

  border-radius: .5rem;

  box-sizing: border-box;

  color: #482307;

  column-gap: 1rem;

  cursor: pointer;

  display: flex;

  font-family: ui-sans-serif,system-ui,-apple-system,system-ui,"Segoe
UI",Roboto,"Helvetica Neue",Arial,"Noto Sans",sans-serif,"Apple Color
```

```
Emoji","Segoe UI Emoji","Segoe UI Symbol","Noto Color Emoji";  
  
font-size: 100%;  
  
font-weight: 700;  
  
line-height: 24px;  
  
margin: 0;  
  
outline: 2px solid transparent;  
  
padding: 1rem 1.5rem;  
  
text-align: center;  
  
text-transform: none;  
  
transition: all .1s cubic-bezier(.4, 0, .2, 1);  
  
user-select: none;  
  
-webkit-user-select: none;  
  
touch-action: manipulation;  
  
box-shadow: -6px 8px 10px rgba(81,41,10,0.1),0px 2px 2px rgba(81,41,10,0.2);  
  
display: none;  
  
}
```

```
.right {  
  
    right: 0px;  
  
    width: 300px;  
  
}
```

```
@media (max-width: 576px) {
```

```
  .form {
```

```
    width: 100%;
```

```
  }
```

```
}
```

```
.abc{
```

```
  width: 50%;
```

```
}
```

INDEX.HTML

```
<!DOCTYPE html>
```

```
<html lang="en">
```

```
<head>
```

```
  <meta charset="UTF-8">
```

```
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
```

```
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
```

```
  <meta name="description" content="This website is develop for identify the  
safety of url.">
```

```
  <meta name="keywords" content="phishing url,phishing,cyber  
security,machine learning,classifer,python">
```

```
  <meta name="author" content="Narma12">
```

```
<!-- BootStrap -->
```

```
<link rel="stylesheet"
```

```
href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css"
"
```

```
integrity="sha384-
```

```
9alt2nRpC12Uk9gS9baDI411NQApFmC26EwAOH8WgZl5MYYxFfc+NcPb1dKGj7Sk"
```

```
crossorigin="anonymous">
```

```
<link href="static/style.css" rel="stylesheet">
```

```
<title>Web Phishing detection</title>
```

```
</head>
```

```
<body>
```

```
<div class=" container">
```

```
<div class="row">
```

```
<div class="form col-md" id="form1">
```

```
<h2>WEB PHISHING DETECTION</h2>
```

```
<br>
```

```
<form action="/" method ="post">
```

```
<input type="text" class="form__input" name ='url' id="url"
placeholder="Enter URL" required="" />
```

```
<label for="url" class="form__label">URL</label>
```

```
<button class="button" role="button" >Check here</button>
```

```
</form>
```

</div>

<div class="col-md" id="form2">

<h6 class = "right ">{{ url }}</h6>

<h3 id="prediction"></h3>

<button class="button2" id="button2" role="button"
onclick="window.open('{{url}})' target="_blank" >still want to Continue</button>

<button class="button1" id="button1" role="button"
onclick="window.open('{{url}})' target="_blank">Continue</button>

</div>

</div>

<p>Narma12</p>

</div>

<!-- JavaScript -->

<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js"

integrity="sha384-
DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj"

crossorigin="anonymous"></script>

<script

```
src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"
```

```
integrity="sha384-
```

```
Q6E9RHvblyZFJoft+2mJbHaEWWlDl9l9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo"
```

```
crossorigin="anonymous"></script>
```

```
<script
```

```
src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js"
```

```
integrity="sha384-
```

```
OgVRvuATP1z7JjHLkuOU7Xw704+h835Lr+6QL9UvYjZE3lpu6Tp75j7Bh/kR0JkI"
```

```
crossorigin="anonymous"></script>
```

```
<script>
```

```
let x = '{{xx}}';
```

```
let num = x*100;
```

```
if (0<=x && x<0.50){
```

```
    num = 100-num;
```

```
}
```

```
let txtx = num.toString();
```

```
if(x<=1 && x>=0.50){
```

```
    var label = "Website is "+txtx +"% safe to use...";
```

```
    document.getElementById("prediction").innerHTML = label;
```

```
    document.getElementById("button1").style.display="block";
```

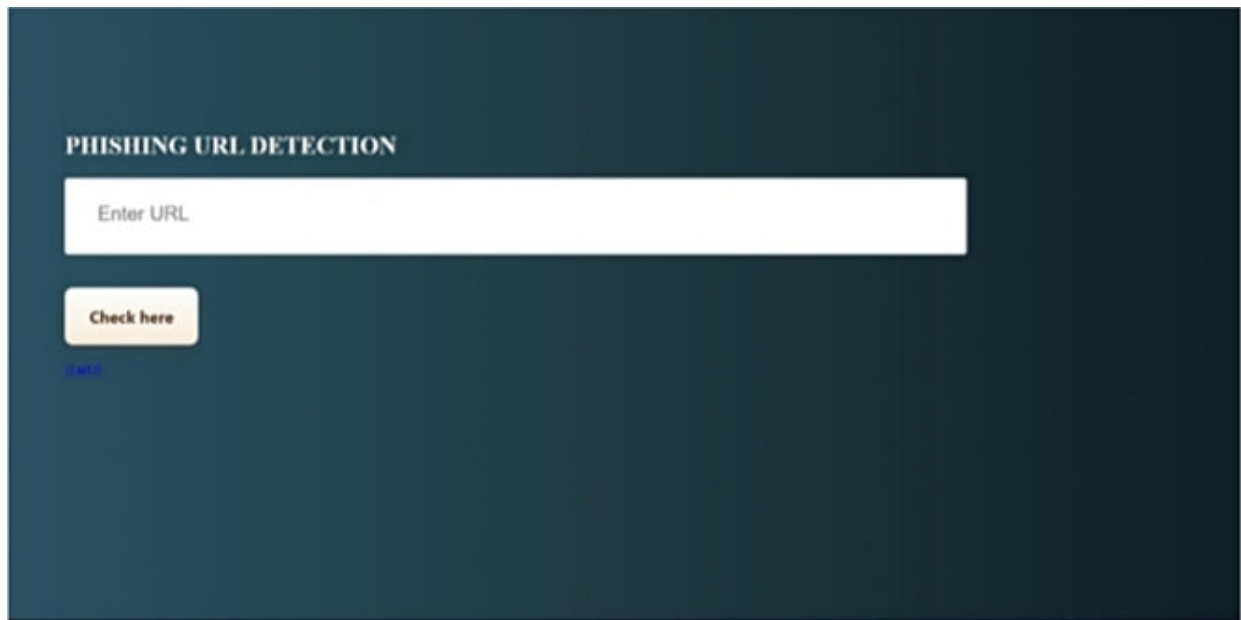
```
}
```



```
else if (0<=x && x<0.50) {  
  
    var label = "Website is "+txtx +"% unsafe to use..."  
    document.getElementById("prediction").innerHTML = label ;  
    document.getElementById("button2").style.display="block";  
}  
  
</script>  
  
</body>  
  
</html>
```

SCREEN SHOTS

Step1: Home Page of web Phishing Detection



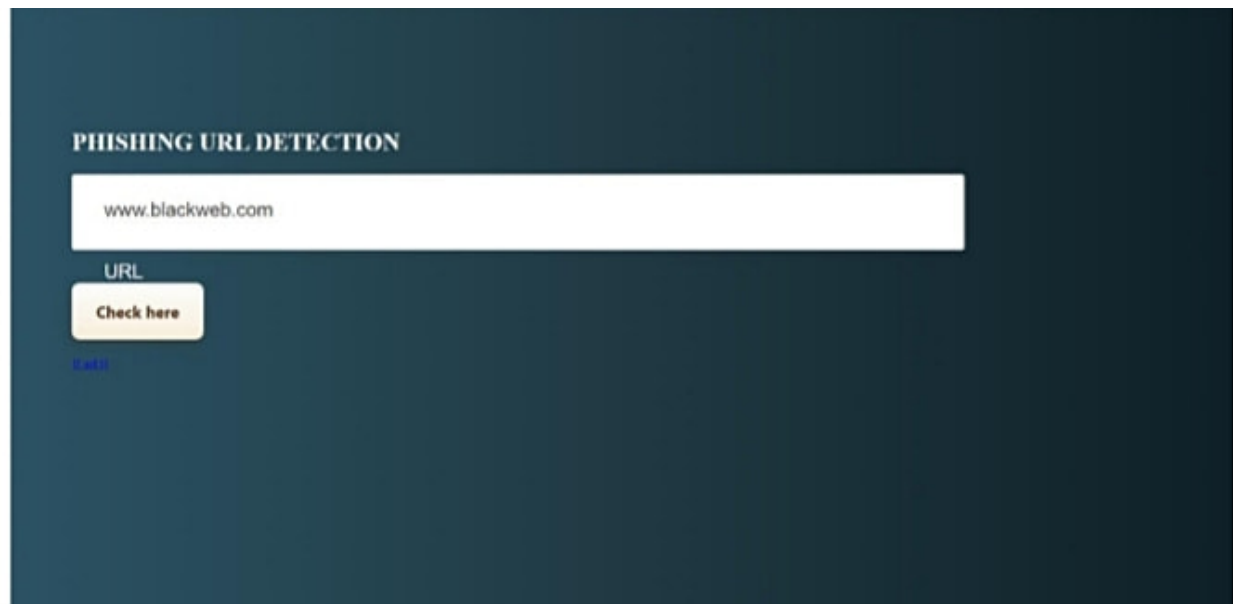
PHISHING URL DETECTION

Enter URL

Check here

[Back](#)

Step 2: Enter the Website URL for Detecting the Website is Phishing or not



PHISHING URL DETECTION

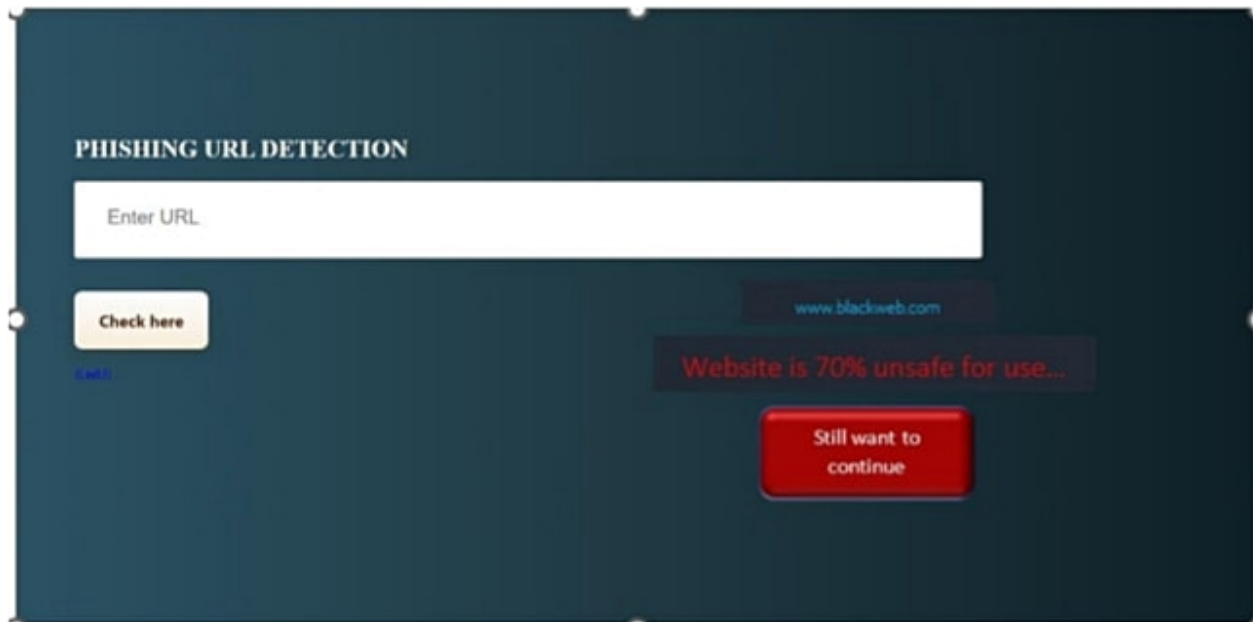
www.blackweb.com

URL

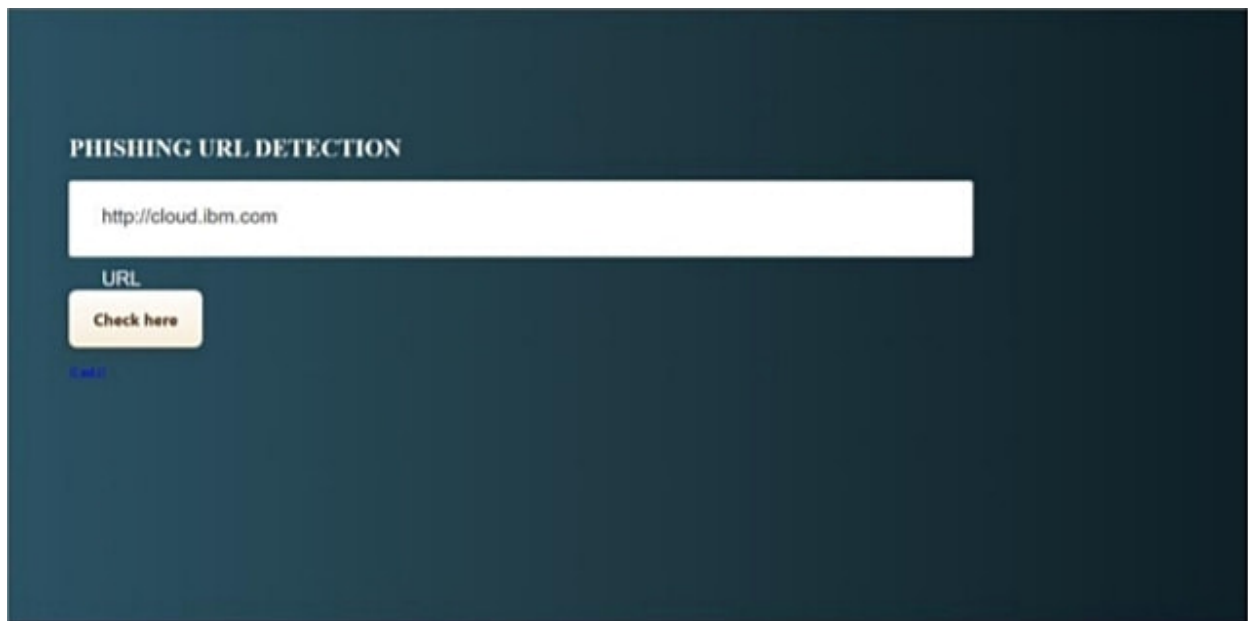
Check here

[Back](#)

Step 3: The Website is phishing it shows the warning Message like Website is unsafe to use



Step 4: Enter the Website URL for Detecting the Website is Phishing or not



Step 5: The Website is Legitimate it shows the Message safe to use

PHISHING URL DETECTION

Enter URL

Check here

[\[Add\]](#)

<http://cloud.ibm.com>

Website is 70% safe to use...

continue