Dissertation Report on

WAVD: WEB APPLICATION VULNERABILITY DETECTOR

Submitted in partial fulfillment of the requirements of the degree of

Bachelor of Engineering

by

ROHAN SHARMA (34)
SHUBHAM YADAV (35)
DHIRAJ MISHRA (36)

Supervisor:

PROF. AKSHAYA PRABHU

Computer Engineering Department

VIVA Institute of Technology

University of Mumbai

2022-2023

CERTIFICATE

This is to certify that the project entitled **"WAVD: WEB APPLICATION VULNERABILITY DETECTOR"** is a bonafide work of **"Rohan Sharma, Shubham Yadav, Dhiraj Mishra"** submitted to the University of Mumbai in partial fulfillment of the requirement for the award of the degree of **"Bachelor of Engineering in Computer Engineering"**

Prof. Akshaya Prabhu
Supervisor

Prof. Sunita Naik
In Charge HOD

Dr. Arun Kumar
Principal

Project Dissertation Approval for B.E.

This project report entitled **WAVD: WEB APPLICATION VULNERABILITY DETECTOR** by **Rohan Sharma, Shubham Yadav and Dhiraj Mishra** is approved for the degree of **Bachelor of Engineering in Computer Engineering.**

Examiners

1.----------

2.----------

Date:

Place:

Abstract

In today's world, Cyber security has become an important leap in the form of jobs, education. But the reality is that only a few are aware of the major web vulnerabilities. Some statistical studies show that small scale industries are directly and indirectly connected to the world of the internet, but they are not aware of the major web vulnerabilities of their web application. Since website hosting has become common nowadays, most of the web applications are prone to attacks and malicious attacks of web applications. Assessing and avoiding these vulnerabilities require deep knowledge of these vulnerabilities. There are numerous online scanners available on the Internet that provide only paid limited service. The tools are made in a way that it can only operate in command line interface or in any programming language. So, it is a difficult task for a normal person to operate the scanners without previous knowledge. The web applications are the most common interface for security-sensitive information and functionality available. As web applications are sources of sensitive data, they are prone to vast numbers of web-based attacks. The majority of these attacks happen because of vulnerabilities resulting from input validation problems. Although these vulnerabilities are easy to understand and mitigate, many web developers are unaware of these security aspects. Which results in more vulnerable web applications on the Internet. We implemented a system which will scan the web application for the most frequent vulnerabilities in an automated manner. Our system detects flaws in web applications and presents a comprehensive report

Table of Contents

Acknowledgment

List of Figures

List of Tables

Declaration

We declare that this written submission represents our ideas in our own words and where other's ideas or words have been included. We have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in our submission. We understand that any violation of the above will result in disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

ROHAN SHARMA

SHUBHAM YADAV

DHIRAJ MISHRA

Date

Chapter 1

Introduction

Web applications have become an integral part of everyday life, but many of these applications are associated with vulnerabilities. In this era, where website hosting has become cheap and easy, the security has failed to keep up. Such vulnerabilities can risk small scale to large scale industries. Exploitation of vulnerability by an unauthorized person demands for quick recovery of these flaws so that reputation of the organization can be recovered. Therefore, vulnerability scanners can be widely used to evaluate the known weakness and vulnerabilities in a website. Many applications are becoming online, but how secure these applications are a matter of concern. Thus, it becomes necessary to find vulnerabilities that may cause severe risk to user's security. Vulnerability assessment means detecting the vulnerabilities before they could be used by an attacker. It is not only performed on a particular application, but it can be run on any platform on which the application is run. This strategy only takes into consideration all the factors that can provide the correct answer for assessment of the vulnerability and security of the system. Therefore, vulnerability scanners are used to scan the network and software application.

We can infer that Security plays an important role in developing websites. Unfortunately, web developers are not aware of these security aspects resulting in more vulnerable websites. Some

of the most commonly occurring ones being SQLi, XSS, RFI, Open Redirection and Phishing. So, we developed a system that will find these vulnerabilities in given web applications and report them to the user of the system. We are developing a system that will accept the target URL from the user. The system will then crawl the target URL in an Automated way using AI techniques and collect all the connected URLs. Then it will scan all collected URLs and it will test different payloads to exploit the vulnerabilities. Finally, a report will be generated which will contain the detected vulnerabilities and payloads used.

Chapter 2

Literature Survey

The following chapter is a literature survey of the previous research papers and research which gives detailed information about the previous system or previous methods along with its advantages and disadvantages.

2.1 Survey of the existing system

A survey was done on the existing literature, products, and technologies to find out their shortcomings, to learn about the working flow of architecture, the impact of effective scanners, and research gaps in various case studies or in their systems.

Haibo Chen, et. al [1], proposed a web vulnerability scanner that is designed based on python programming language. They have also adopted Browser/server architecture for realization. The information collection module and the vulnerability detection module are deployed on the database created by them. The main modules presented are 1) Database and management module 2) Information collection module 3) Vulnerability detection module 4) Tasks and targets management module. The database is established based on SQLite in version 3.24.0. The scanner ran on a machine serving Windows 10 Home, and equipped with 16 GB RAM, i5 CPU in 2.3 GHz. There are 2 detection modules provided by the proposed scanner including

comprehensive detection and special detection and it is mainly focused on common web security vulnerabilities such as SQL injection, XSS, framework vulnerabilities etc. The special vulnerability module is developed based on the pocsuite3.

Mr. Kalyan D Bmane, et. al [2], the proposed system has used waterfall model for the project, Demand Gathering and analysis, System Design , Implementation , Testing , Reading system , Maintenance Which researchers on numerous internet vulnerabilities that comes under linguistics uniform resource locator, XSS, RFI, LFI, SQLi, CMDi . So, system mainly concentrates on detection and prevention of web application by using various techniques such as Dynamic Allocation, File Size Verification, Digital Signature and Sanitization of Input

Pranav Gadekar, et. al [3] The system will crawl the target URL in an Automated way using AI techniques and collect all the connected URLs Then it will scan all collected URLs and it will test different payloads to exploit the vulnerabilities. A report will be generated which will contain the detected vulnerabilities and payloads used. Various Scanners used in the system Acunetix Vulnerability Assessment Engine: It is a security testing solution used for both standalone and as a part of complex environments. Burp Suite Web Vulnerability Scanner: It hunts out an honest range of vulnerabilities Qualys Web Application Scanner: It also covers public cloud instances and provides you instant visibility of vulnerabilities like SQLi and XSS. Nessus Vulnerability Scanner: Nessus is the vulnerability assessment solution for security practitioners

Bin Wang, et. al [4] the proposed system uses a Web crawler module which can crawl from a URL to any URL associated with it. Core vulnerability detection module enhances the scalability of the system. Scanner interface and overall function integration, the interface of the system is implemented by pyqt4 technology. The interface component is written through the xml file, and the specific components in the relevant xml file are retrieved in the python core file. It also scans multiple target websites at the same time. In this system XSS vulnerability; SQL injection vulnerability and File upload vulnerability.

Binny George, et. al [5] proposed a system which scans SQL injection, cross site scripting and Broken Authentication If a situation arrives where the scanner cannot detect the vulnerability, then the attacker can easily crawl into the system and exploit the data and resources. So Nmap, Nessus, Acunetix, Nikto, Burp Suite are compared. Burp Suite scans SQL Injection, Improper Error, Cross Site, Scripting, Insecure, Cookies, Session Token, URL, Password Auto Enabled.

Vahid Shahrivari, et. al [6] proposed in this paper, a comparative evaluation of different machine learning methods provided on detecting the phishing websites. The machine learning methods studied are Logistic Regression, Decision Tree. Random Forest, AdaBoost, SVM, KNN, Artificial Neural Networks, Gradient Boosting, and XGBoost. They evaluated the accuracy, precision, recall, F1 score, training time, and testing time of these models and they used different methods of feature selection and hyperparameters tuning for getting the best results. XGBoost gives the best accuracy of 98.32% using the phish tank dataset.

Rishikesh Mahajan, el. at [7] in this paper deals with machine learning technology for detection of phishing URLs by extracting and analyzing various features of legitimate and phishing URLs. Decision Tree, random forest and Support vector machine algorithms are used to detect phishing websites. The Python program is implemented to extract features from URL like Address bar bashed, Domain Bar Based, HTML and JS based features. Here Random Forest gives the best accuracy of 97.14% by using alexa.com dataset. Scikit-learn tool has been used to import ML algorithms.

P.S.Sadaphule, et. al [8] proposed System to prevent web application from various malicious attacks of RFI and LFI by using PHP language and CSS, preventing them using Dynamic Allocation, File Size Verification, Digital Signature and Sanitization of Input prevention methods. Since RFI and LFI are very rare attacks but can harm the whole system so this project focuses on RFI and LFI vulnerabilities detection.

Smita Patil, et. al [9] proposed an approach that allows to automatically identify whether above vulnerabilities present in Web applications or not. System uses black box approach for the analysis of the targeted application. First to find out all the notions of injections. To avoid false negatives Author maintains a state while crawling. By applying clustering algorithm system detect vulnerabilities with assurance of increasing performance.

Table 2.1: Analysis Table

Title	Summary	Advantages	Tech Stack
An Automatic Vulnerability Scanner for Web Application. [1]	The proposed scanner includes comprehensive detection and special detection, and it is mainly focused on common web security vulnerabilities such as SQL injection, XSS, framework vulnerabilities etc. The special vulnerabilities etc. The special vulnerability module is developed based on the pocsuite3	The vulnerability scanning system can improve the efficiency of web application vulnerability	Python, JavaScript
Web Vulnerability Scanner. [2]	The project comes underneath linguistics uniform resource locator. System tends to be studied numerous vulnerabilities like Remote File Inclusion, Locate File Inclusion, SQLI, Cross-Site Scripting.	RFI (Remote File Inclusion) and LFI (Local File Inclusion) are truly vulnerable. Though these kinds of attacks are rare, unauthorized access can harm the whole system.	Python, JavaScript, Burp Suite
Automated Web Application Vulnerability Scanner. [3]	application is having any of these vulnerabilities: SQL Injection and Cross Site Scripting The report will be generated consisting of endpoint affected, payload used,	Supports automated and reliable crawling. Optimized use of the number of threads to control the load on the target application.	Python, JavaScript

	and generalized remediation.		
Research on Web Application Security Vulnerability Scanning Technology. [4]	In this paper, the characteristics of various types of vulnerabilities is analyzed, Such as XSS Detection, SQL Injection Detection and File upload Vulnerability A web application vulnerability detection method is proposed, combining the fuzzy test method to enable the scanning system to automatically detect vulnerabilities.	The automated vulnerability scanning system can improve the efficiency of web application vulnerability scanning compared to traditional manual detection vulnerabilities that are time-consuming, labor-intensive, and inefficient.	Python, JavaScript
Web Application Security Scanner for Prevention and Protection against Vulnerabilities. [5]	Proposed method is a vulnerability scanner which detects the vulnerabilities like SQL injection, cross site scripting, broken authentication, payload, email disclosure. It uses Burp Suite which gives best scanning results.	The above proposed scanner is best suited for beginners who are not aware of the complex steps of scanning Vulnerability scanning identifies the security. vulnerabilities in an organization. The advantage of using vulnerability scanners is that it identifies known	Burp Suite, fuzzy testing, Python.

		security exposures before attackers find them..	
Phishing Detection Using Machine Learning Techniques. [6]	In this paper, a comparative evaluation of different machine learning methods provided on detecting the phishing websites. They used different methods of feature selection for getting the best results.	he main advantage of XGBoost is its fast speed compared to other algorithms, such as ANN and SVM, and its regularization parameter that successfully reduces variance	Python , JavaScript
Detection Using Machine Learning Algorithms. [7]	Machine learning technology for detection of phishing URLs by extracting and analyzing various features of legitimate and phishing URLs. Decision Tree, random forest and Support vector machine algorithms are used to detect phishing websites.	XGBoost is its fast speed compared to other algorithms, such as ANN and SVM, and its regularization parameter that successfully reduces variance.	Forest classifier gives best accuracy with lowest false negative rate than other two classifiers
Prevention of Website Attack Based on Remote File Inclusion. [8]	RFI and LFI are very rare attacks but can harm the whole system so this project focuses on RFI and LFI vulnerability detection.	identifies Remote File Inclusion and Local File Inclusion which are rare vulnerabilities.	CSS and PHP.
Design of Efficient	Proposed clustering	Automatically	Python

Web Vulnerability Scanner. [9]	approach to efficiently detect the SQL Injection, Xpath Injection and Cross Site Scripting attacks. The objective is to improve and its evolution hist detection efficiency of vulnerability scanner maintaining low false positive and false negative rate.	identify whether above vulnerabilities is present in web applications of no. .	

2.2 Research gap

There are tools that detect vulnerabilities, but they are not open source and also expensive. 10 Most of the scanners detect few vulnerabilities, generally the most common SQL Injection and XSS injection. One of the most common attacks is phishing which is not detected in most Vulnerability systems.

2.3 Problem statement and Objective

There is an unavailability of automated scanners for detecting vulnerabilities in web applications. And some scanners which are present are highly expensive. This leads to defacement, hijacking, stealing data from the server. This creates a security problem for all businesses as well as government people. So, making an affordable scanner is important. Scanners first crawl the web pages of a particular domain and scan each URL using different payloads and find out if a URL is vulnerable or not. The aim here is to develop WAVD (Web App Vulnerability Detector), a tool to scan & test URLs for certain vulnerabilities & security issues by simply inspecting the corresponding client-side website. The overall system would include a virtual server with modules for detecting the different vulnerabilities, along with a proxy server, to direct requests from a browser to the virtual server first while visiting a website. The proxy could warn the user before redirecting to the website if some vulnerabilities are found during the scan done by our virtual server.

We intend to identify & assess the following classes of vulnerabilities that a website may possess:

• Absence of Valid TLS Certificates

• Cross-Site Scripting (XSS)

• Potential Phishing Attempts

• Open Redirection

2.4 Scope

We are going to scan rare vulnerabilities which affect the whole system. We have added phishing detection using machine learning with best accuracy 98.32% using XGradient Boost. XSS, File Upload Vulnerability and TLS Certificate Validation vulnerabilities will be added in the system.

Chapter 3

Proposed system and Implementation

This chapter gives an overview of the Proposed system.

3.1 Algorithm

The algorithm for the proposed system is as follows:

Step 1: Start

Step 2: Configure Browser

Step 3: Start Web Browser

Step 4: Start Intercept Proxy

Step 5: Enter URL

Step 6: Send URL to restful API component

Step 7: Restful API will call services

Step 8: Servers will perform security check result

Step 9: Restful API fetches security check result

Step 10: Proxy will fetch results

Step 11: Result will be formatted in HTML

Step 12: It will return the result on the web browser

Step 13: Show results

Step 14: END

3.2 Details of System

3.2.1 Software Requirements

1. Proxy Server: NodeJS
2. Virtual Server: Python & Fast API
3. Visual Studio Code.
4. WAVD Dashboard: HTML, CSS, JS & Bootstrap Framework
5. Brower: Google Chrome
6. Operation System: Window & Linux
7. Jupyter notebook

3.2.1 Hardware Requirements

1.Processor: intel i5 (7th gen).
2. Ram: 8GB+.
3. Graphic Card: 2GB.
4. Storage: 120GB

3.3 Design Details

In this section flow diagrams and block diagrams of the system are explained.

3.3.1 Flow Chart Diagram

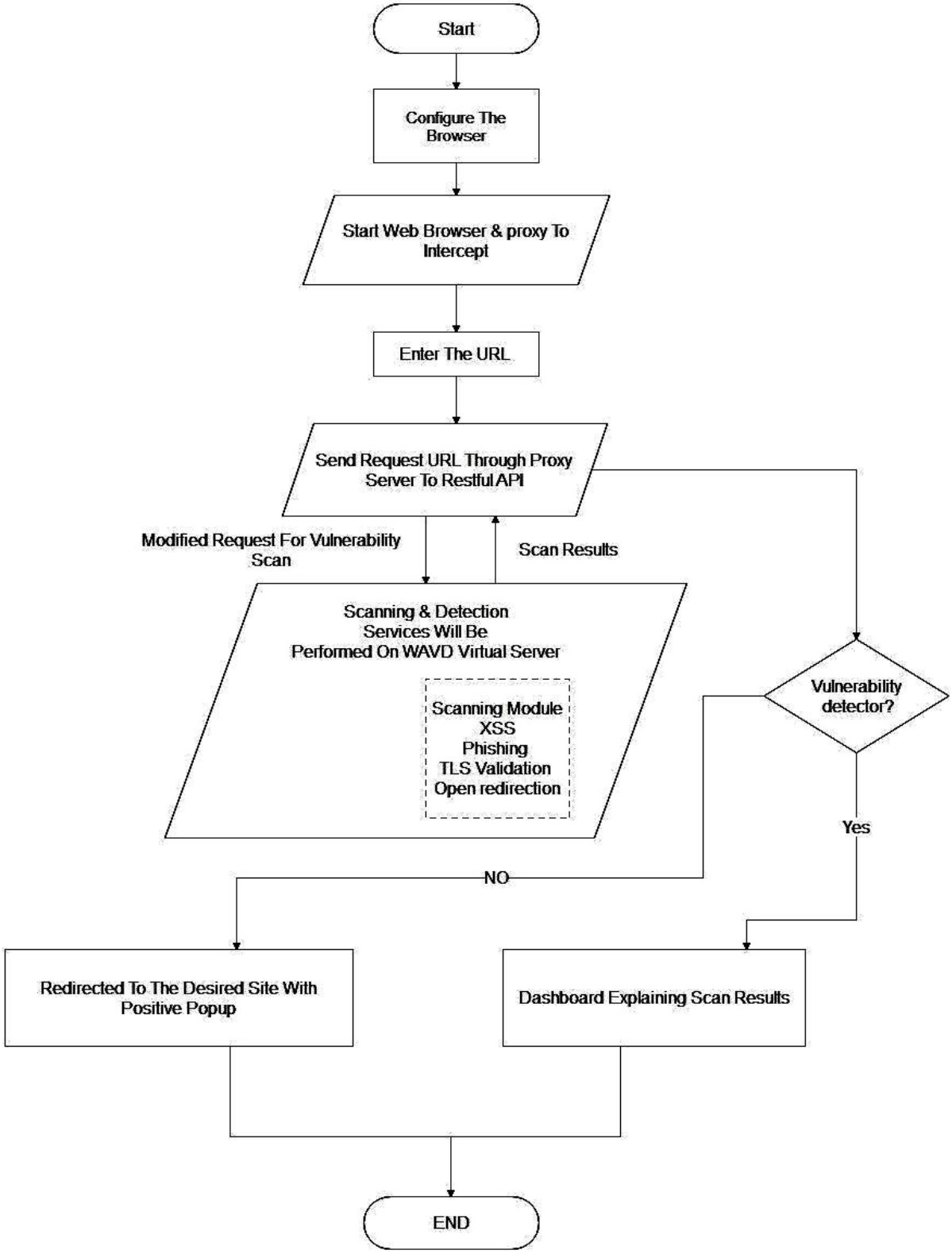


Figure 3.1 WAVD flow diagram

In Figure 3.1 represents a flow diagram for the proposed system. A flow diagram is a collective term for a diagram representing a flow or set of dynamic relationships in a system. Flow diagrams are used to structure and order a complex system, or to reveal the underlying structure of the elements and their interaction.

3.3.2 Block Diagram

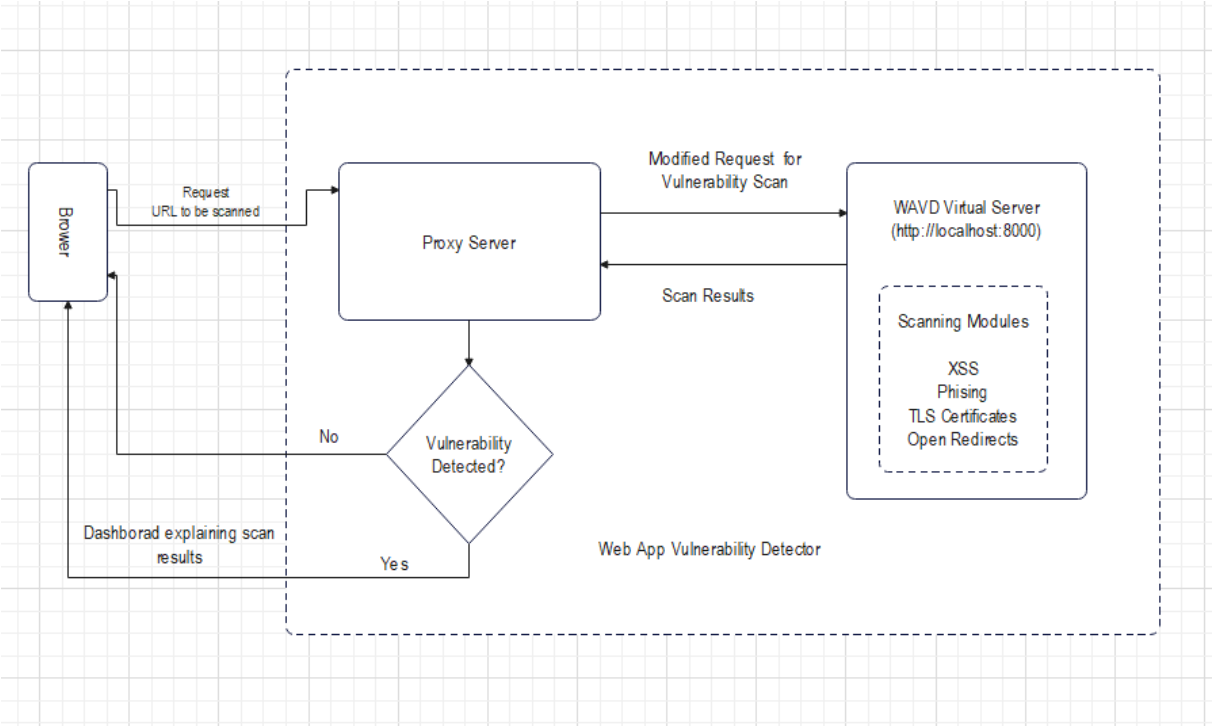


Figure 3.2 WAVD block diagram

In Figure 3.2 Block diagram for the complete system of WAVD is demonstrated. The application starts with a dashboard where the user can enter the URL then it is connected to Proxy Server. The proxy passes the URL to the virtual server where the models are shown each model is different for different vulnerabilities.

3.3 System Architecture

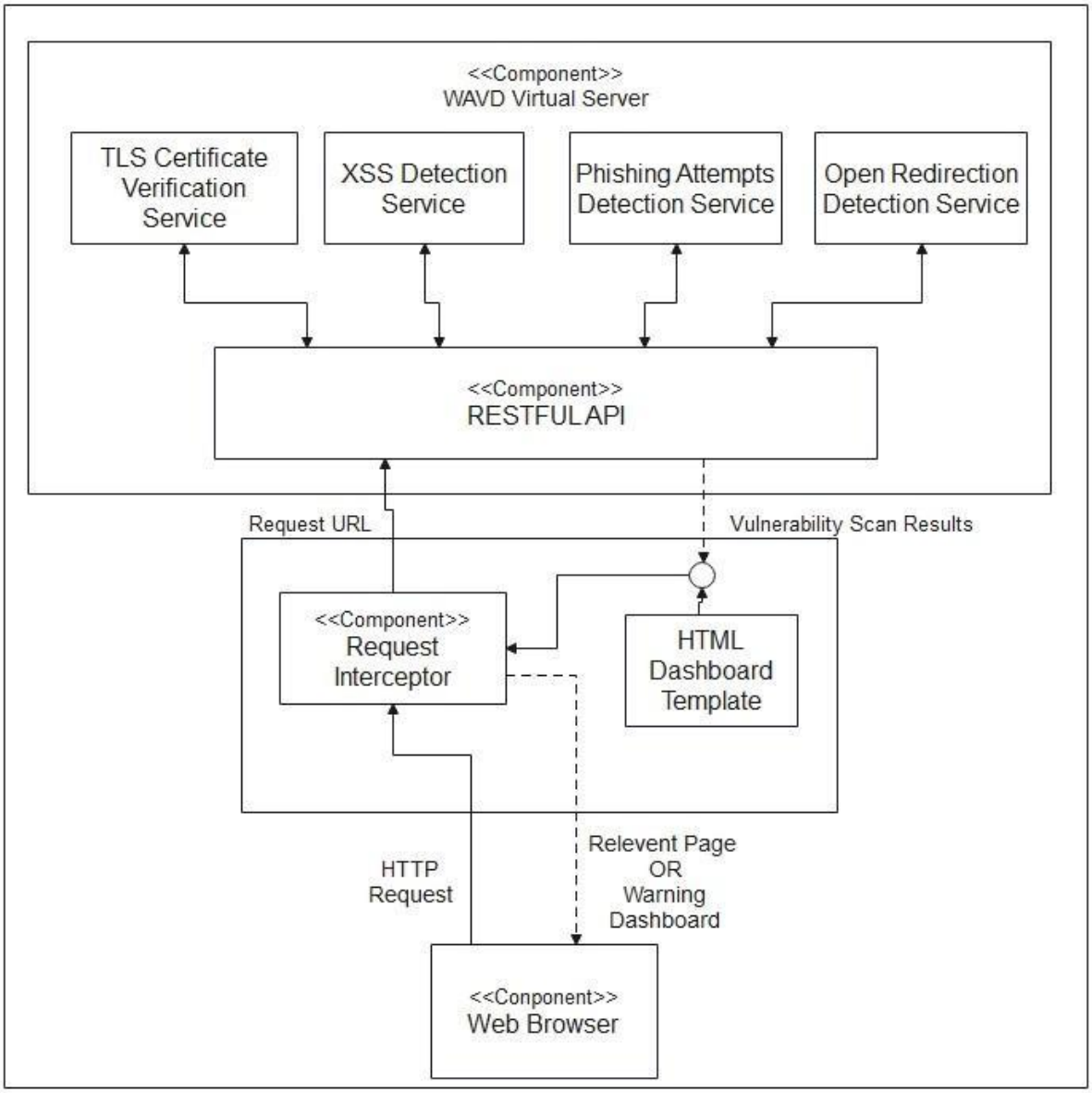


Figure 3.3 WAVD System Architecture

In Figure 3.3 System architecture for the complete system of WAVD is demonstrated. The application starts with a dashboard where the user can enter the URL then it is connected to Proxy Server. The proxy passes the URL to the virtual server where the models are shown each model is different for different vulnerabilities.

3.4 Methodology

The proposed system is developed based on the concept of enhancing the existing systems for checking web application vulnerabilities. To develop this system, an analysis of the existing systems was done. The system comprises a Proxy Server and a Virtual Server. The Proxy Server intercepts the URL and redirects it to the Virtual Server. The Virtual Server houses six different modules that perform various vulnerability detection tasks.

To develop the Proxy Server, we needed to learn how to inject relevant information from JSON objects into DOM elements of an HTML template. We also familiarized ourselves with node-http-proxy for developing the proxy server. After intercepting the request, the URL is being sent to the virtual server where all the modules are present. Each module performs different tasks based on their detection service. The modules include TLS Certificate Verification Service, XSS Detection Service, Open Redirection Detection Service, and Phishing Detection Service.

To develop the Virtual Server, we familiarized ourselves with FastAPI & Heroku for developing & deploying the virtual server. We set up the server & integrated the various microservices such as TLS Certificate Verification Service where we gained background knowledge about TLS certificates, the security they offer & ways to detect their absence. We also familiarized ourselves with the working of available CLI python tools like check-tls-certs & used a similar approach to develop the service for checking the existence & validity of a website's TLS certificate. After this microservice, the system checks XSS, Open Redirection by implementing the payloads for particular detection services & inject them into DOM elements that are susceptible to the particular attacks.

After checking all these microservices, the system checks for phishing attempts by delving deeper into phishing attacks & studying the paper Intelligent rule-based phishing website rule-based phishing websites classification to understand the relevant features to identify phishing websites. We use kaggle phishing websites datasets to train & test some traditional ML classifiers (DT, SVM & RF) and identify the most important features indicating phishing & build functions to extract them.

The system generates a report specifying an overview of the detected vulnerabilities. It is useful for small-scale industries that cannot afford the expensive tools required for web security, which is crucial for any firm because security and privacy are the most important things for users as

well as for admin.

In conclusion, the proposed system provides an effective way to detect and report vulnerabilities in web applications. The system comprises a Proxy Server and a Virtual Server that work together to scan and detect various types of vulnerabilities. The Proxy Server intercepts the URL and redirects it to the Virtual Server, where different modules perform different vulnerability detection tasks. The system is developed using various tools and technologies, including FastAPI & Heroku, node-http-proxy, and CLI python tools like check-tls-certs. The system generates a detailed report of the detected vulnerabilities, which is useful for small-scale industries that cannot afford expensive web security tools. Overall, the proposed system is an effective solution to enhance web application security.

3.5 UML Diagram

3.5.1 Use Case Diagram

A use case diagram at its simplest is a representation of a user's interaction with the system that shows the relationship between the user and the different use cases in which the user is involved.

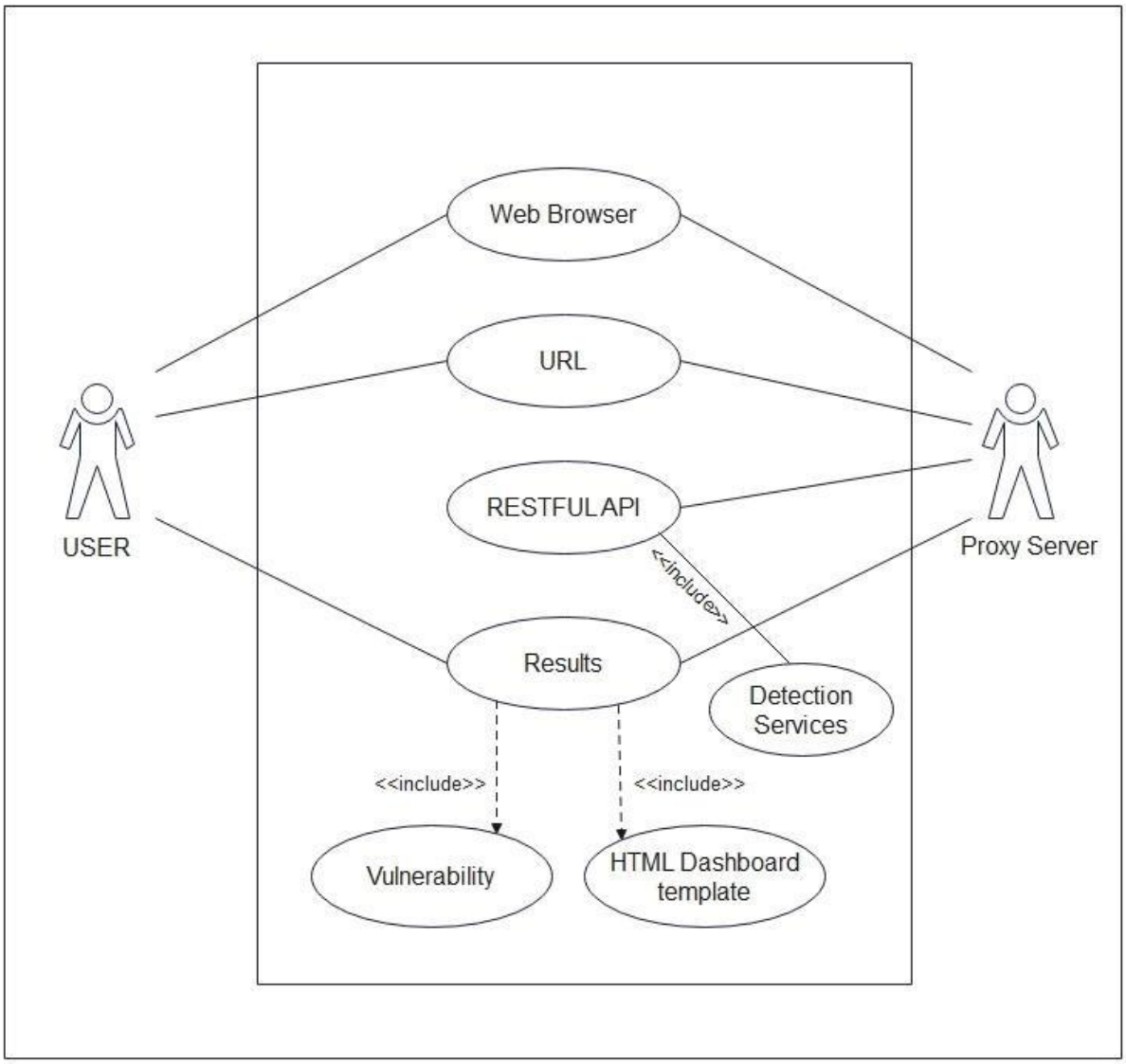


Figure 3.4 Use case diagram

Chapter 4

Results and analysis

4.1 Results

The results demonstrate the need for the web vulnerability scanner on daily basis where data is Stole on vulnerable sites and it also help small business to track their bugs and vulnerabilites which prevents them to not to buy expensive tools . This chapter covers the results and analysis of the system. The main focus is to create the Proxy Server & Virtual Server, as it will be the one to handle intercepting the URL and redirecting it to the microservices tasks for all. Integration of all the modules is another part. After all, this will test all the modules together and will fix identified bugs.

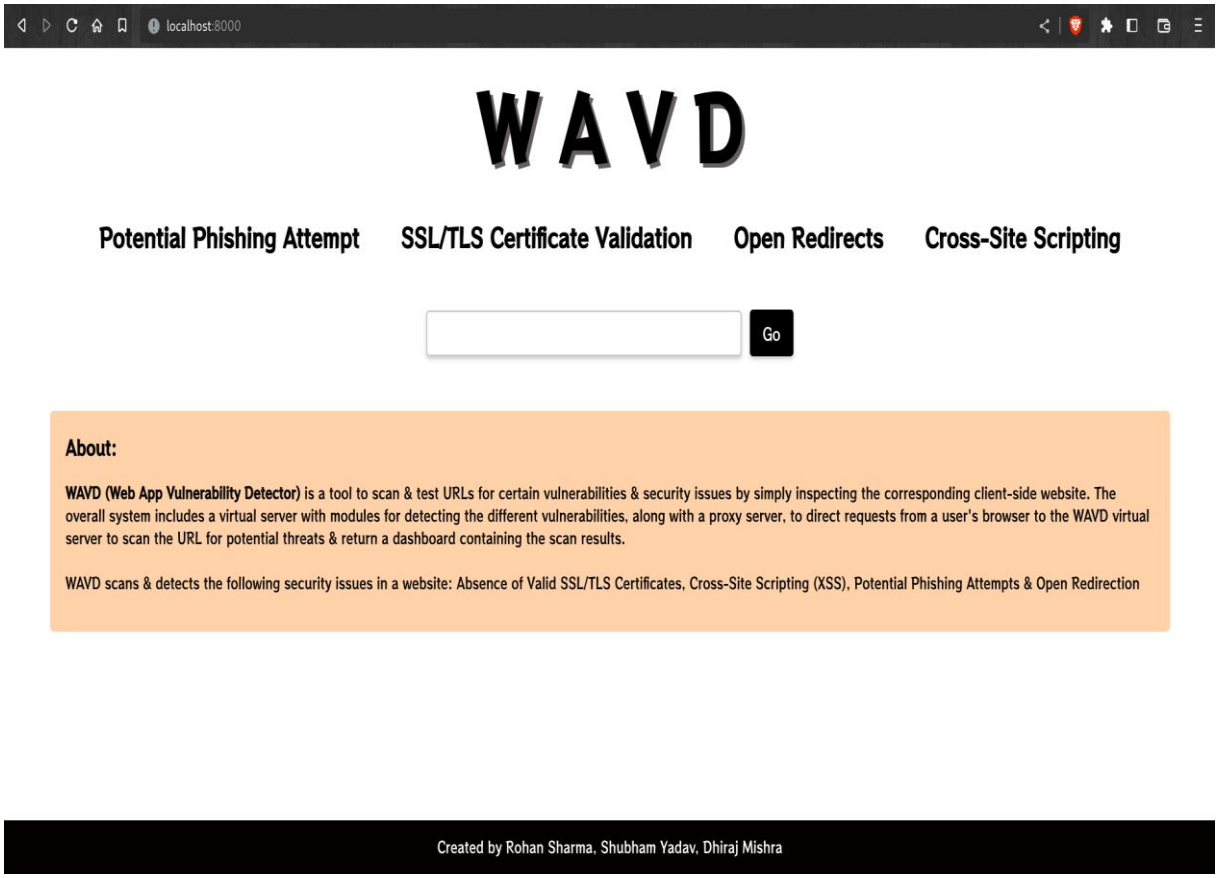
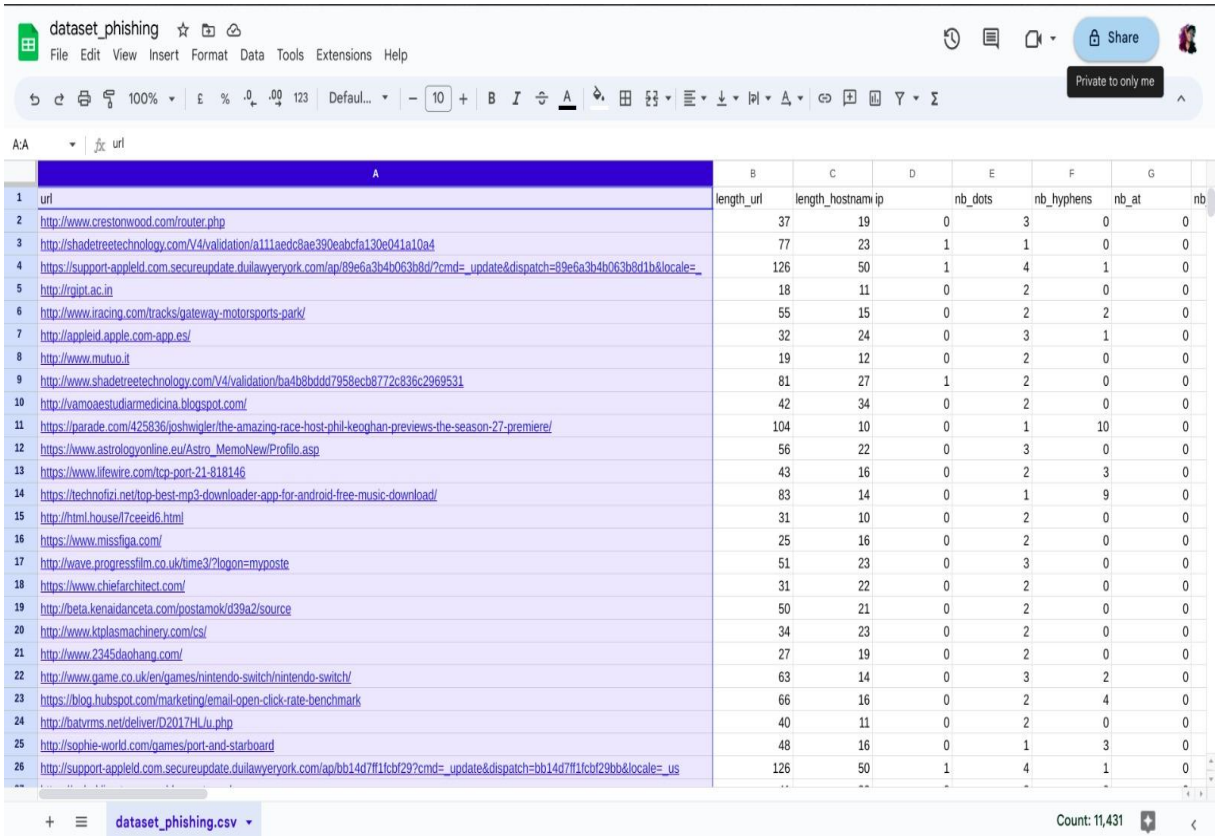


Figure 4.1 Dashboard

Figure 4.1 depicts homepage dashboard mentioned here is designed to provide an easy-to-use interface for users to initiate a website vulnerability scan. The central input field serves as the primary means for the user to input the URL of the website they want to scan. This input field may be accompanied by additional fields or options that allow the user to customize the scan parameters, such as the depth of the scan, the type of vulnerabilities to scan for, or the scanning frequency.

Once the user submits the URL, the dashboard initiates a scan of the website. Depending on the complexity of the website and the parameters selected by the user, the scan may take some time to complete. During the scan, the dashboard may display a progress bar or other visual indicators to show the user the status of the scan and how much time is remaining.



dataset_phishing

File Edit View Insert Format Data Tools Extensions Help

Share

Private to only me

A:A url

	A	B	C	D	E	F	G
1	url	length_url	length_hostnam	ip	nb_dots	nb_hyphens	nb_at	nb
2	http://www.crestonwood.com/router.php	37	19	0	3	0	0	
3	http://shadetreetechnology.com/V4/validation/a111aedc8ae390eabcfa130e041a10a4	77	23	1	1	0	0	
4	https://support-appleId.com.secureupdate.duilawyeryork.com/ap/89e6a3b4b063b8d/?cmd=_update&dispatch=89e6a3b4b063b8d1b&locale=_	126	50	1	4	1	0	
5	http://rgipt.ac.in	18	11	0	2	0	0	
6	http://www.iracing.com/tracks/gateway-motorsports-park/	55	15	0	2	2	0	
7	http://appleid.apple.com-app.es/	32	24	0	3	1	0	
8	http://www.mutuo.it	19	12	0	2	0	0	
9	http://www.shadetreetechnology.com/V4/validation/ba4b8bddd7958ecb8772c836c2969531	81	27	1	2	0	0	
10	http://vamoaestudiarmedicina.blogspot.com/	42	34	0	2	0	0	
11	https://parade.com/425836/joshwigler/the-amazing-race-host-phil-keoghan-previews-the-season-27-premiere/	104	10	0	1	10	0	
12	https://www.astrologyonline.eu/Astro_MemoNew/Profilo.asp	56	22	0	3	0	0	
13	https://www.lifewire.com/tcp-port-21-818146	43	16	0	2	3	0	
14	https://technofizi.net/top-best-mp3-downloader-app-for-android-free-music-download/	83	14	0	1	9	0	
15	http://html.house/l7ceeid6.html	31	10	0	2	0	0	
16	https://www.missfiga.com/	25	16	0	2	0	0	
17	http://wave.progressfilm.co.uk/time3/?logon=myposte	51	23	0	3	0	0	
18	https://www.chiefarchitect.com/	31	22	0	2	0	0	
19	http://beta.kenaidanceta.com/postamok/d39a2/source	50	21	0	2	0	0	
20	http://www.ktplasmachinery.com/cs/	34	23	0	2	0	0	
21	http://www.2345daohang.com/	27	19	0	2	0	0	
22	http://www.game.co.uk/en/games/nintendo-switch/nintendo-switch/	63	14	0	3	2	0	
23	https://blog.hubspot.com/marketing/email-open-click-rate-benchmark	66	16	0	2	4	0	
24	http://batvrms.net/deliver/D2017HL/u.php	40	11	0	2	0	0	
25	http://sophie-world.com/games/port-and-starboard	48	16	0	1	3	0	
26	http://support-appleId.com.secureupdate.duilawyeryork.com/ap/bb14d7ff1fcbf29?cmd=_update&dispatch=bb14d7ff1fcbf29bb&locale=_us	126	50	1	4	1	0	

dataset_phishing.csv

Count: 11,431

Figure 4.2 Phishing Dataset

Figure 4.2 illustrate dataset of phishing and legitimate URLs is a collection of web addresses used for training and testing phishing detection models. The dataset contains examples of both phishing and legitimate URLs, which are usually labeled or categorized accordingly. During the training phase, the machine learning algorithm is fed a large number of examples from the dataset to learn to recognize the patterns and features that distinguish phishing URLs from legitimate ones. Once trained, the algorithm can be tested on a separate set of examples from the same dataset to evaluate its accuracy and effectiveness. A diverse and representative dataset is crucial for developing accurate phishing detection modules, and it should be periodically updated and expanded to keep up with evolving cybercriminal tactics.

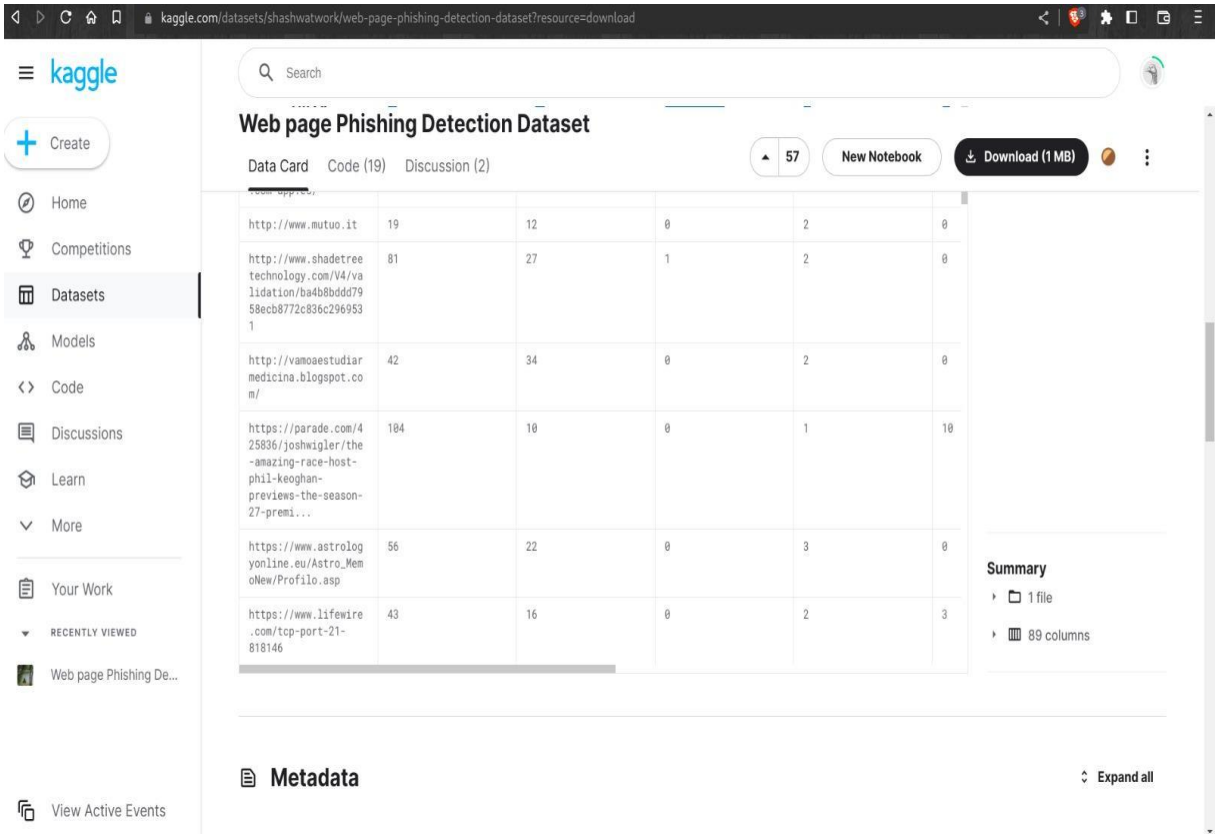


Figure 4.3 Phishing Dataset and its Attributes

Figure 4.3 depicts the Kaggle dataset of phishing URLs is a popular resource used for scanning and detecting phishing attacks by phishing detection modules. This dataset contains a vast collection of URLs that have been identified as part of previous phishing attacks. The dataset includes both labeled phishing URLs and legitimate URLs, making it an excellent resource for training and testing machine learning algorithms. The labeled URLs in the dataset enable the machine learning algorithm to learn and recognize the patterns and characteristics that distinguish phishing URLs from legitimate ones, making it easier to detect phishing attacks in real-time.

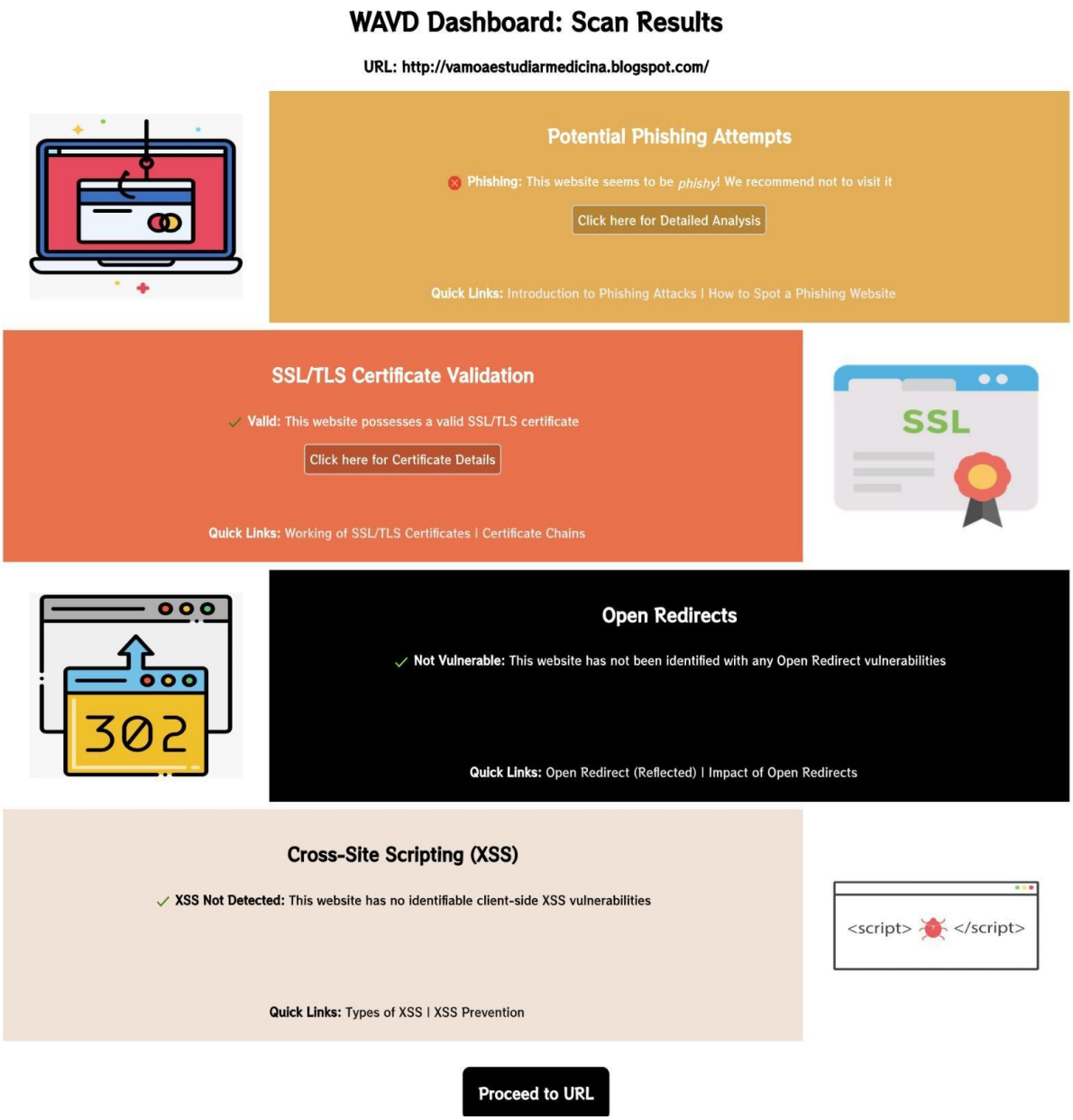
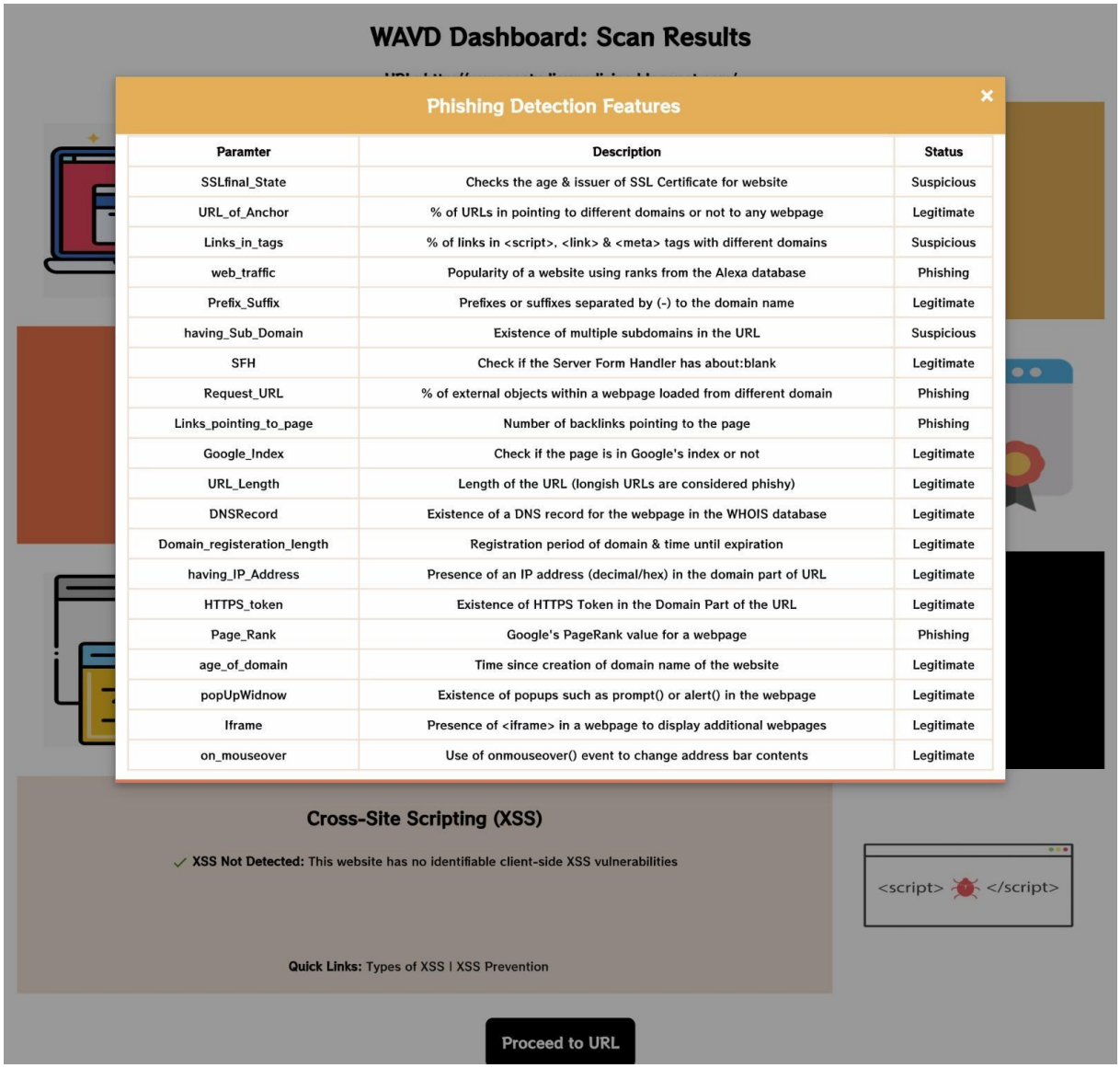


Figure 4.4 Potential Phishing Detected

Figure 4.4 shows an overview of the scan results is a useful tool for identifying potential phishing URLs. The dashboard provides a summary of the scan results, including the number of vulnerabilities detected, their severity level, and a summary of the affected pages and components of the phishing URL. This information is essential in determining the severity of the potential threat and taking the appropriate measures to mitigate the risk.



Paramter	Description	Status
SSLfinal_State	Checks the age & issuer of SSL Certificate for website	Suspicious
URL_of_Anchor	% of URLs in pointing to different domains or not to any webpage	Legitimate
Links_in_tags	% of links in <script>, <link> & <meta> tags with different domains	Suspicious
web_traffic	Popularity of a website using ranks from the Alexa database	Phishing
Prefix_Suffix	Prefixes or suffixes separated by (-) to the domain name	Legitimate
having_Sub_Domain	Existence of multiple subdomains in the URL	Suspicious
SFH	Check if the Server Form Handler has about:blank	Legitimate
Request_URL	% of external objects within a webpage loaded from different domain	Phishing
Links_pointing_to_page	Number of backlinks pointing to the page	Phishing
Google_Index	Check if the page is in Google's index or not	Legitimate
URL_Length	Length of the URL (longish URLs are considered phishy)	Legitimate
DNSRecord	Existence of a DNS record for the webpage in the WHOIS database	Legitimate
Domain_registeration_length	Registration period of domain & time until expiration	Legitimate
having_IP_Address	Presence of an IP address (decimal/hex) in the domain part of URL	Legitimate
HTTPS_token	Existence of HTTPS Token in the Domain Part of the URL	Legitimate
Page_Rank	Google's PageRank value for a webpage	Phishing
age_of_domain	Time since creation of domain name of the website	Legitimate
popUpWidnow	Existence of popups such as prompt() or alert() in the webpage	Legitimate
Iframe	Presence of <iframe> in a webpage to display additional webpages	Legitimate
on_mouseover	Use of onmouseover() event to change address bar contents	Legitimate

Figure 4.5 Phishing Detection Features

Figure 4.5 shows a detailed report of phishing vulnerability, including its parameters, descriptions, and statuses. The report displays which parameters of the URL are suspicious and which parameters are legitimate. This information is essential in identifying potential threats and taking the necessary measures to mitigate the risk.

The detailed report provides a comprehensive overview of the phishing vulnerability, including the specific parameters that are affected. It highlights which parameters are suspicious, providing a clear indication of the potential threat. Additionally, the report includes a description of the vulnerability and its status, which can help security teams better understand the potential impact of the threat.

.

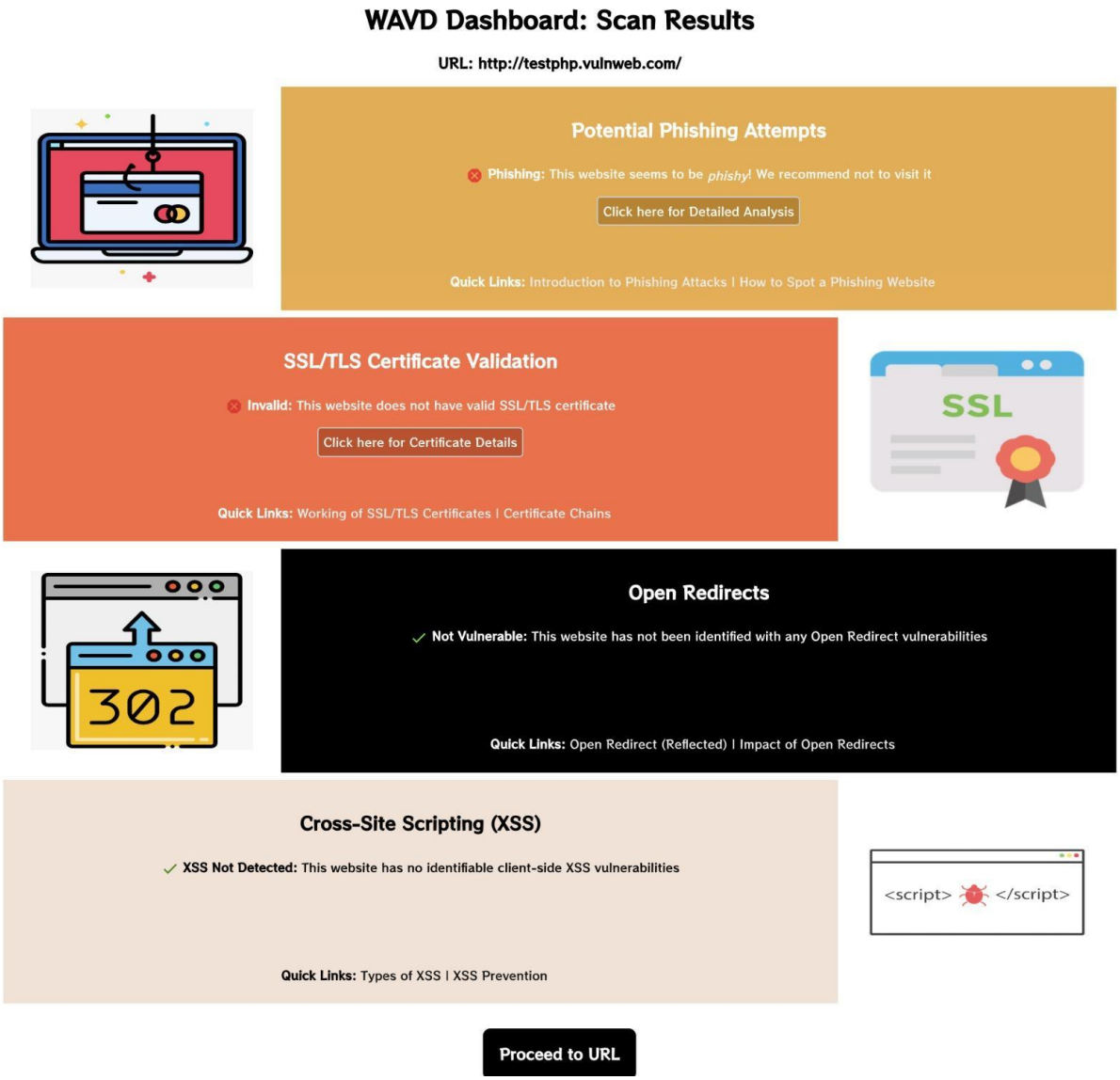


Figure 4.6 SSL/TLS Certificate Scanning

Figure 4.6 shows the detailed overview of the scan results, which includes various types of vulnerabilities such as phishing, cross-site scripting, SSL/TLS certification validation, and open-redirect vulnerabilities. The scan results help security teams to prioritize potential threats and take the necessary measures to prevent them from causing harm.

In this particular case, the figure shows that the potential vulnerable site is susceptible to potential phishing attempts and lacks SSL/TLS validation. This is a significant concern since SSL/TLS validation is essential in ensuring the confidentiality and integrity of data transmitted between the website and its users. If an attacker can intercept this data, it can lead to the theft of sensitive information such as login credentials, personal information, or financial data.

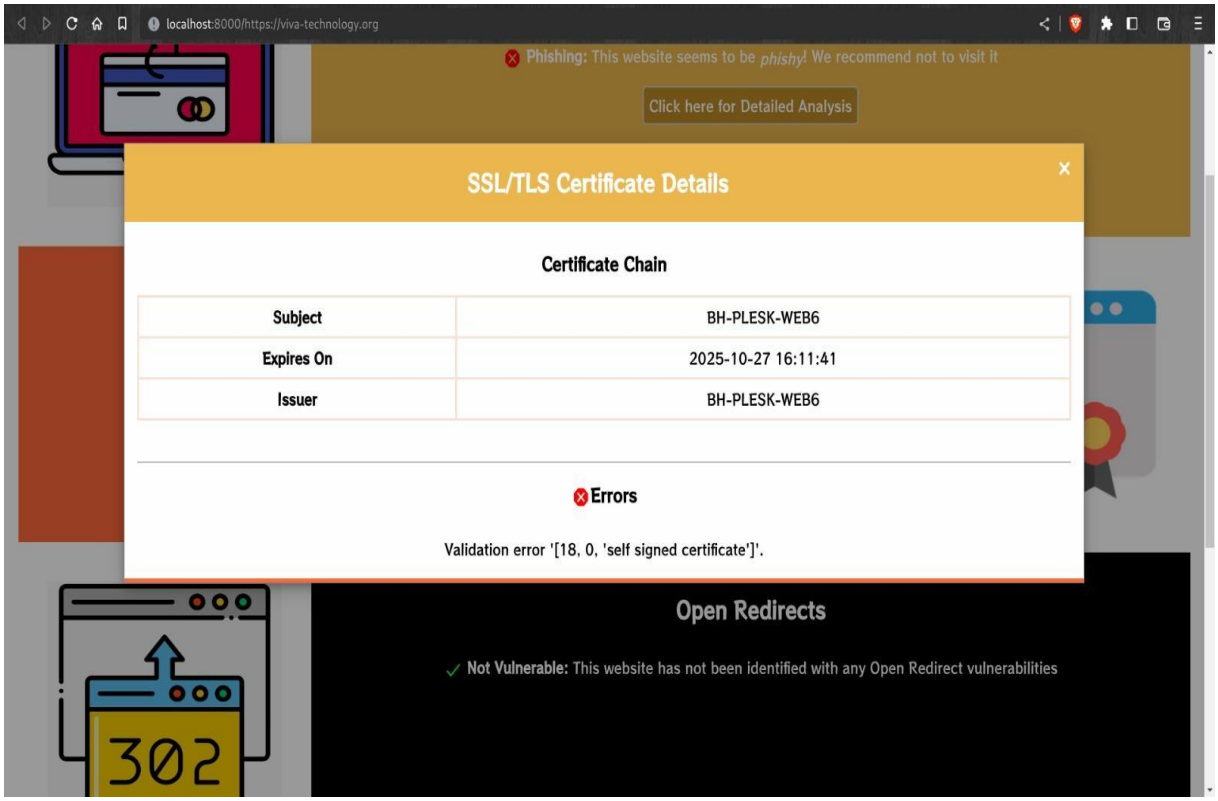


Figure 4.7 SSL/TLS Certificate Details

Figure 4.7 shows the detailed description of the SSL/TLS certificate validation vulnerability found in a potential vulnerable web application. It outlines the exact details of why the website is vulnerable, including information about the certificate itself, such as its expiration date, issuer, and whether it has been revoked. This information allows users to understand the specific reasons why the SSL/TLS certificate validation is inadequate, and take the necessary actions to protect themselves. For example, they may decide not to share sensitive information on the website until the certificate issue is resolved, or they may choose to use a different website with a more secure SSL/TLS certificate. By providing a detailed description of the vulnerability, the figure helps users make informed decisions about their online security and privacy.

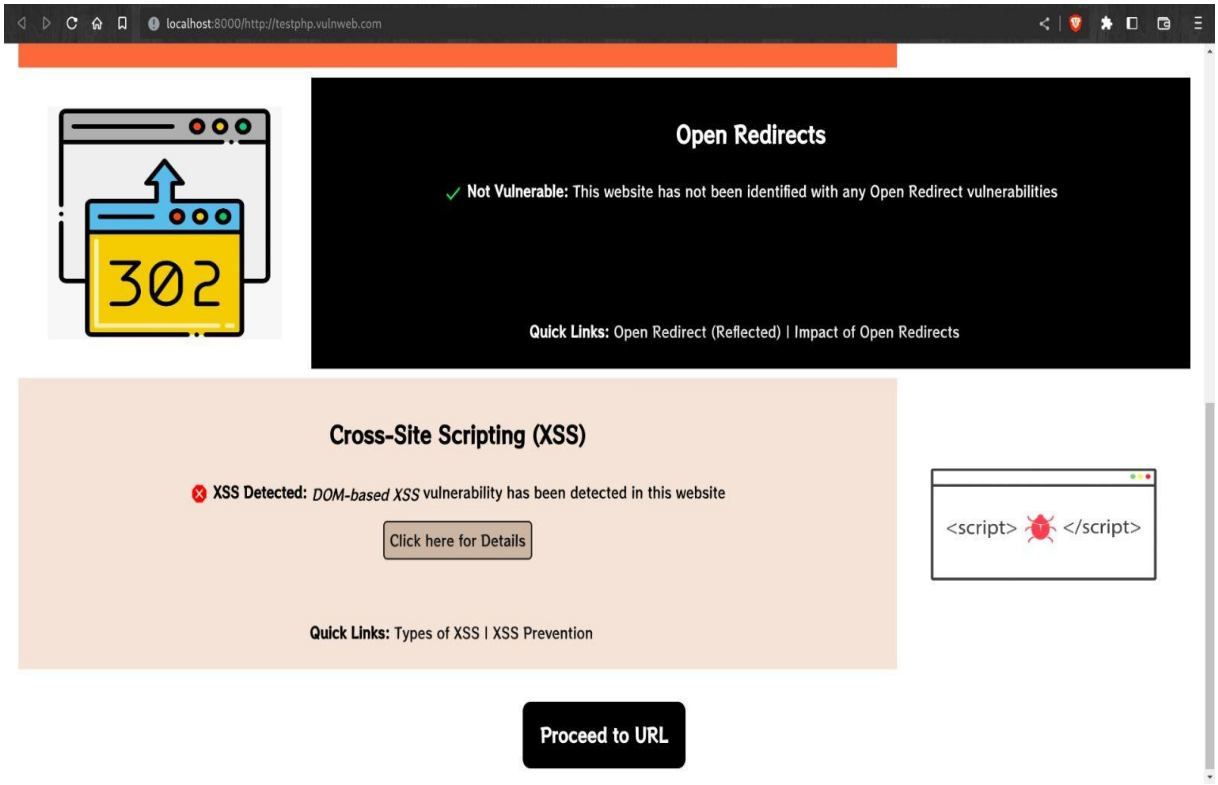


Figure 4.8 Cross-Site-Scripting Scanning

Figure 4.8 illustrates the detection of a cross-site scripting (XSS) vulnerability in a potential vulnerable web application. The scan results indicate that the website is susceptible to XSS attacks, which can allow an attacker to inject malicious scripts into the website and execute them in the context of the victim's browser. This type of attack can be used to steal sensitive information, such as login credentials or personal data, or to perform other malicious activities, such as redirecting the user to a phishing page or spreading malware. By identifying the XSS vulnerability, the figure highlights the potential risks associated with using the website and enables security teams to take appropriate measures to mitigate the vulnerability. This may include patching the website's code to prevent XSS attacks or implementing other security measures to protect against this type of vulnerability.

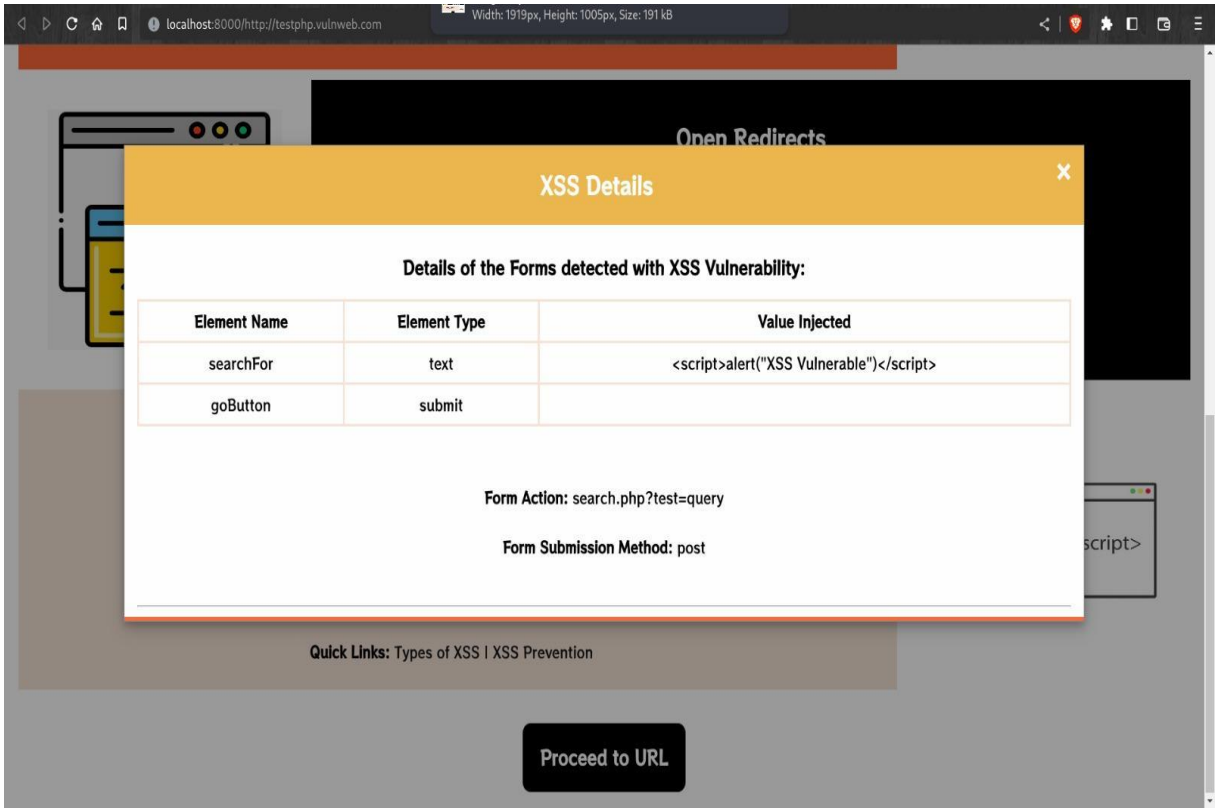


Figure 4.9 Cross-Site-Scripting Details

Figure 4.9 shows the detailed report of a cross-site scripting (XSS) vulnerability in a potential web application. The scan results indicate that the vulnerability is located in the search box section of the website, where a malicious payload can be injected and executed using the post method. This can allow an attacker to steal user cookies, which can contain sensitive information such as login credentials or session tokens. By exploiting this vulnerability, an attacker can gain unauthorized access to the user's account and perform malicious activities such as posting fake content, stealing personal information, or spreading malware.

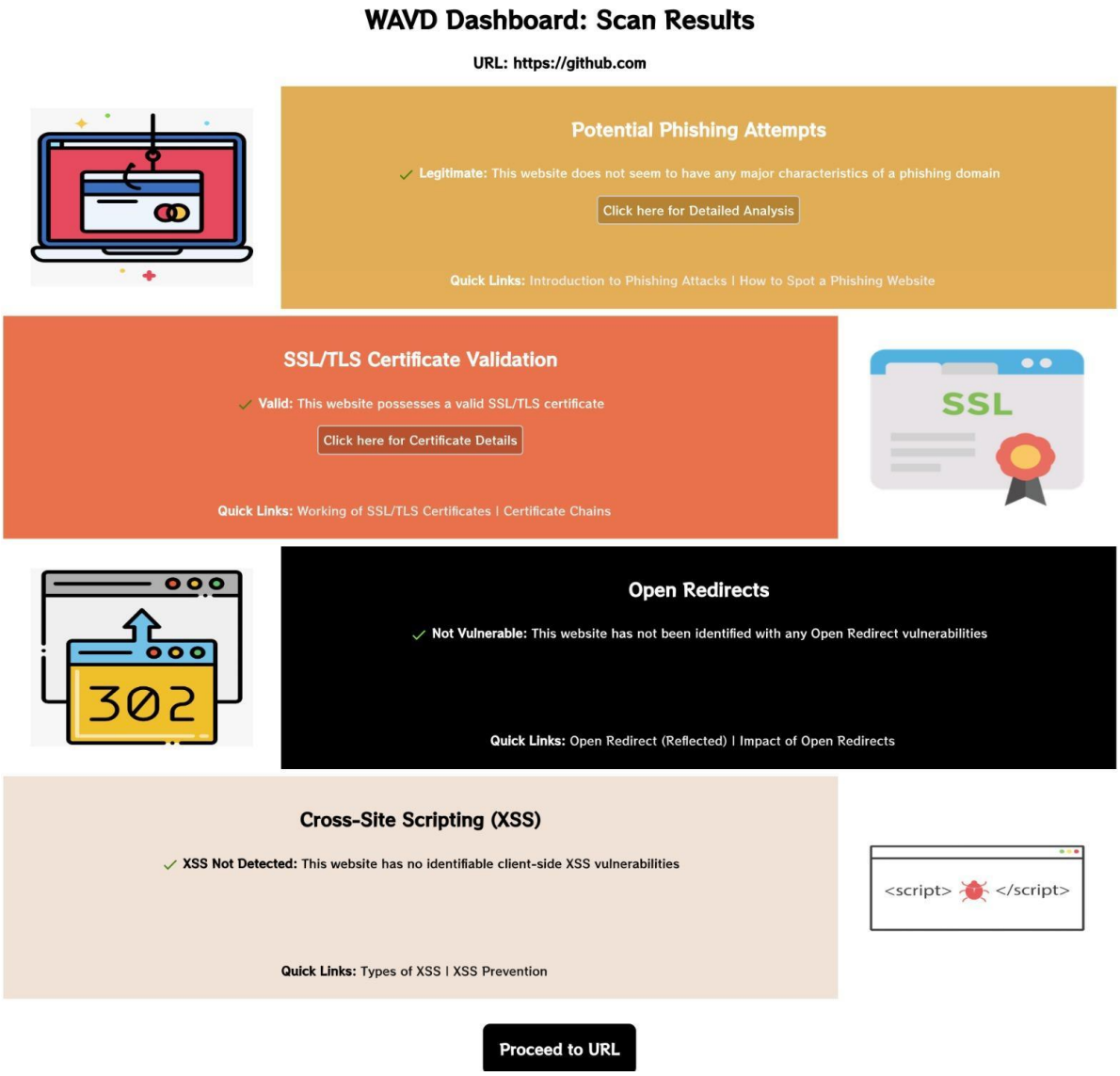


Figure 4.10 Results with no vulnerabilities

Figure 4.10 shows an overview of a scan result for the website github.com, indicating that no vulnerabilities have been detected. The panel confirms that the website is legitimate, valid, and not vulnerable to any known security issues or cross-site scripting (XSS) attacks. Based on this information, users can proceed to the website by clicking on the "Proceed to URL" button with a high level of confidence that it is safe to use.

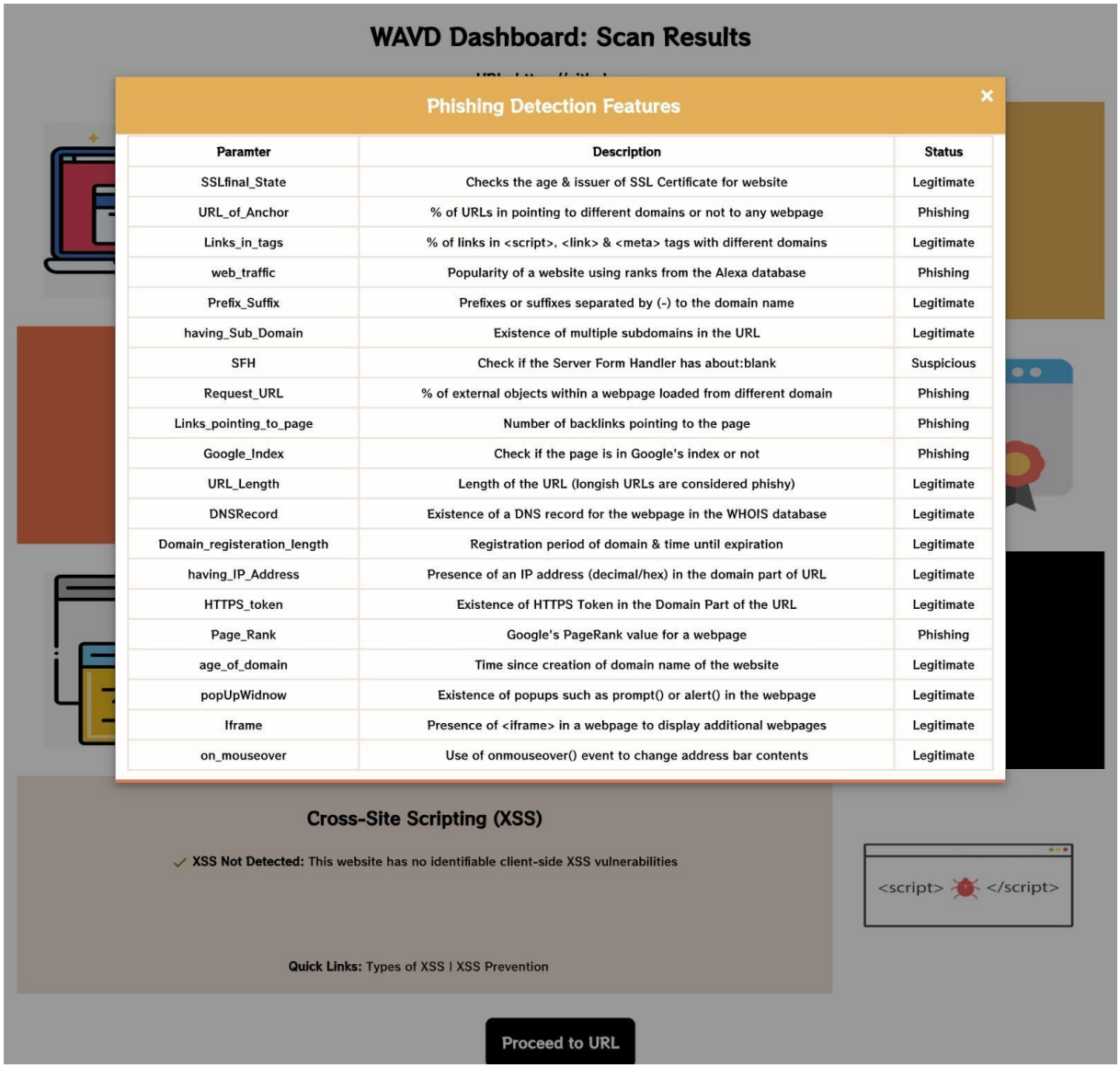


Figure 4.11 Phishing Detection Details

Figure 4.11 shows detailed information on phishing detection and warns users about potential phishing attacks. It shows the scan result of a web application, indicating that it is vulnerable to phishing attempts. The panel provides a summary of the vulnerabilities detected, the affected pages and components of the website, and their severity level.

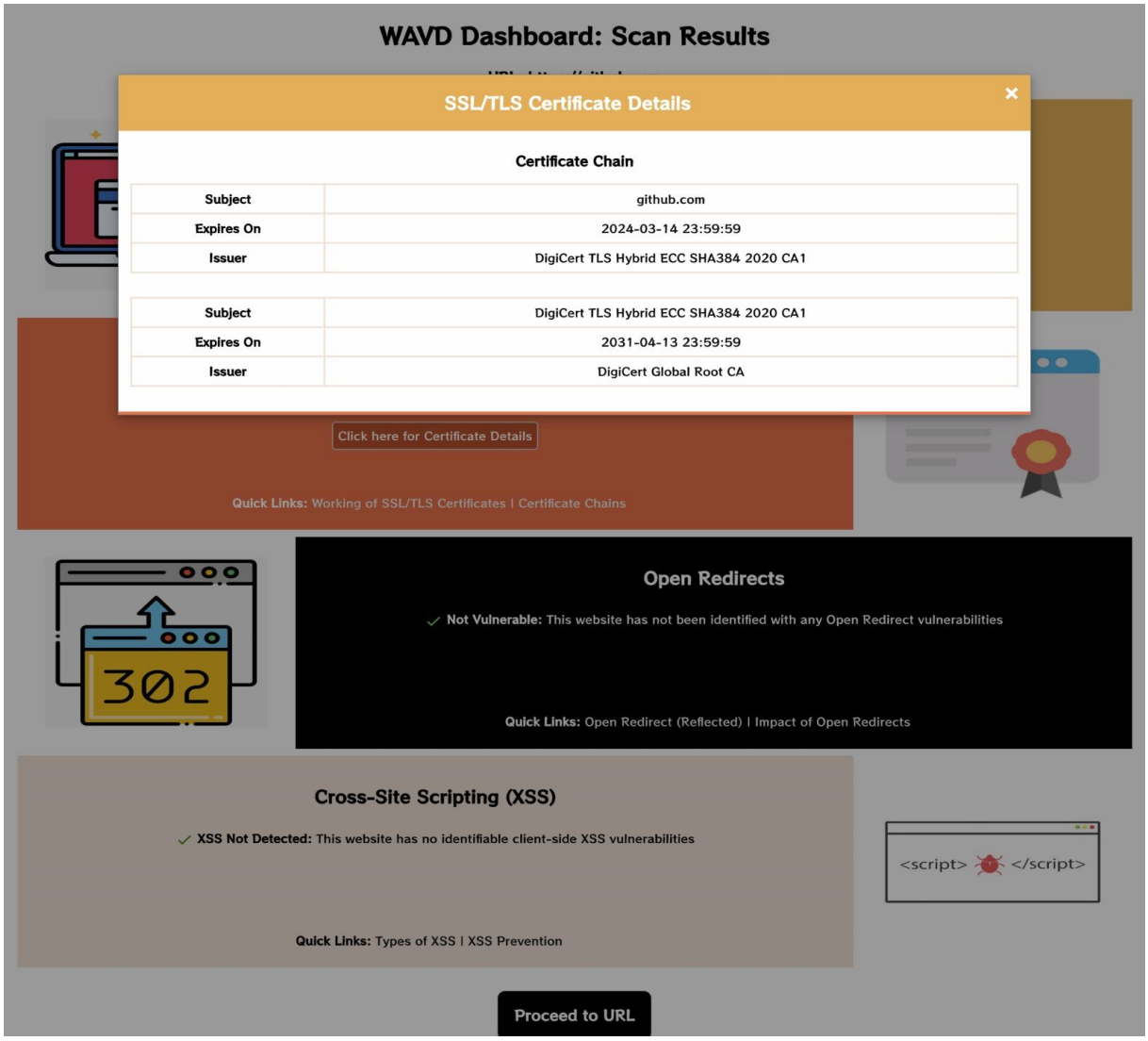


Figure 4.12 SSL/TLS Certificate Details

Figure 4.12 displays the scan result of a website, indicating that it does not have an SSL/TLS certificate. This means that the website is not secure, and any data transmitted through it is at risk of being intercepted and compromised by attackers. Without an SSL/TLS certificate, any information exchanged between the user's browser and the website is vulnerable to eavesdropping and tampering.

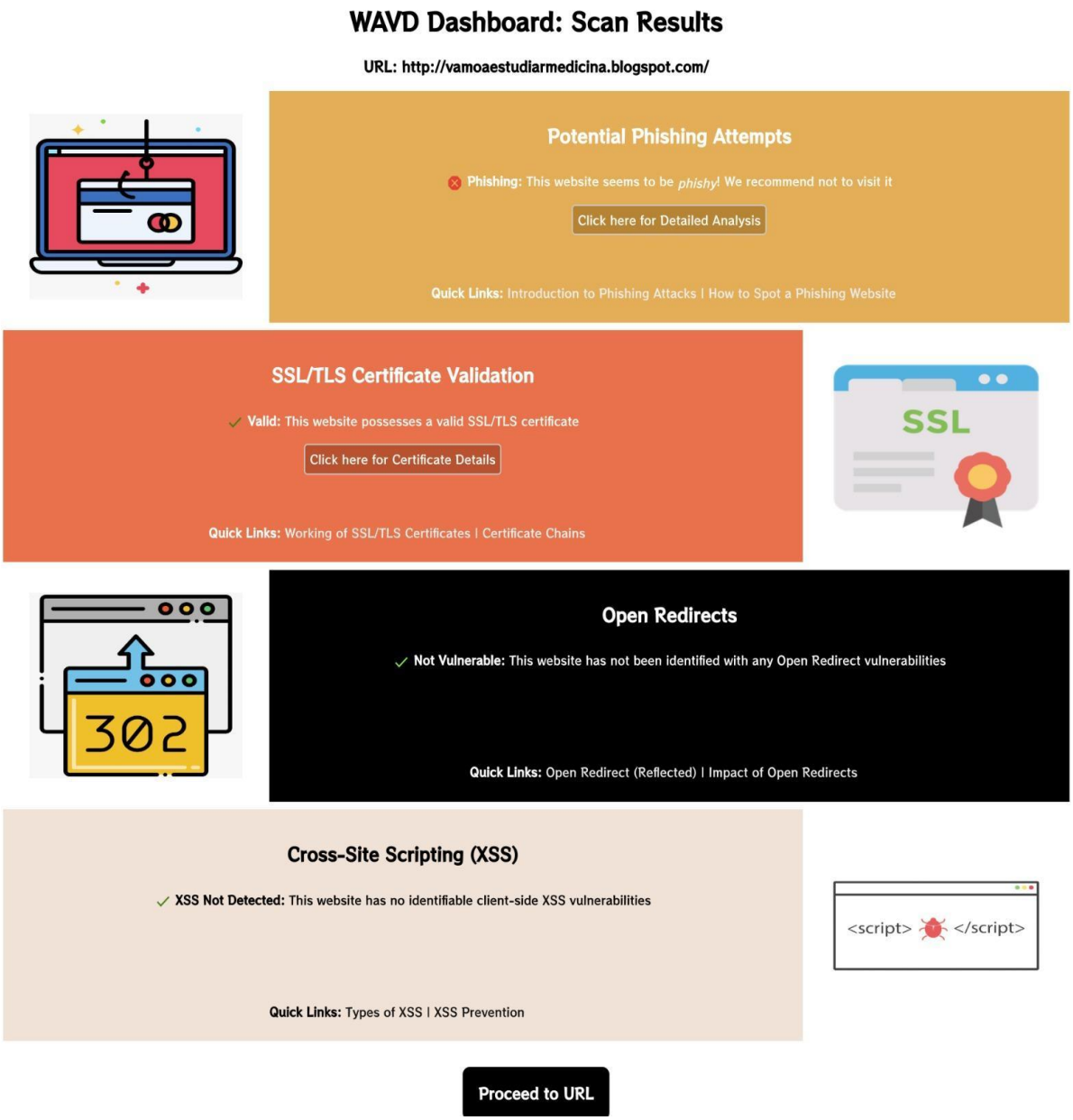


Figure 4.13 Possible phishing identified

Figure 4.13 shows an overview of the scan results is a useful tool for identifying potential phishing URLs. The dashboard provides a summary of the scan results, including the number of vulnerabilities detected, their severity level, and a summary of the affected pages and components of the phishing URL. This information is essential in determining the severity of the potential threat and taking the appropriate measures to mitigate the risk.

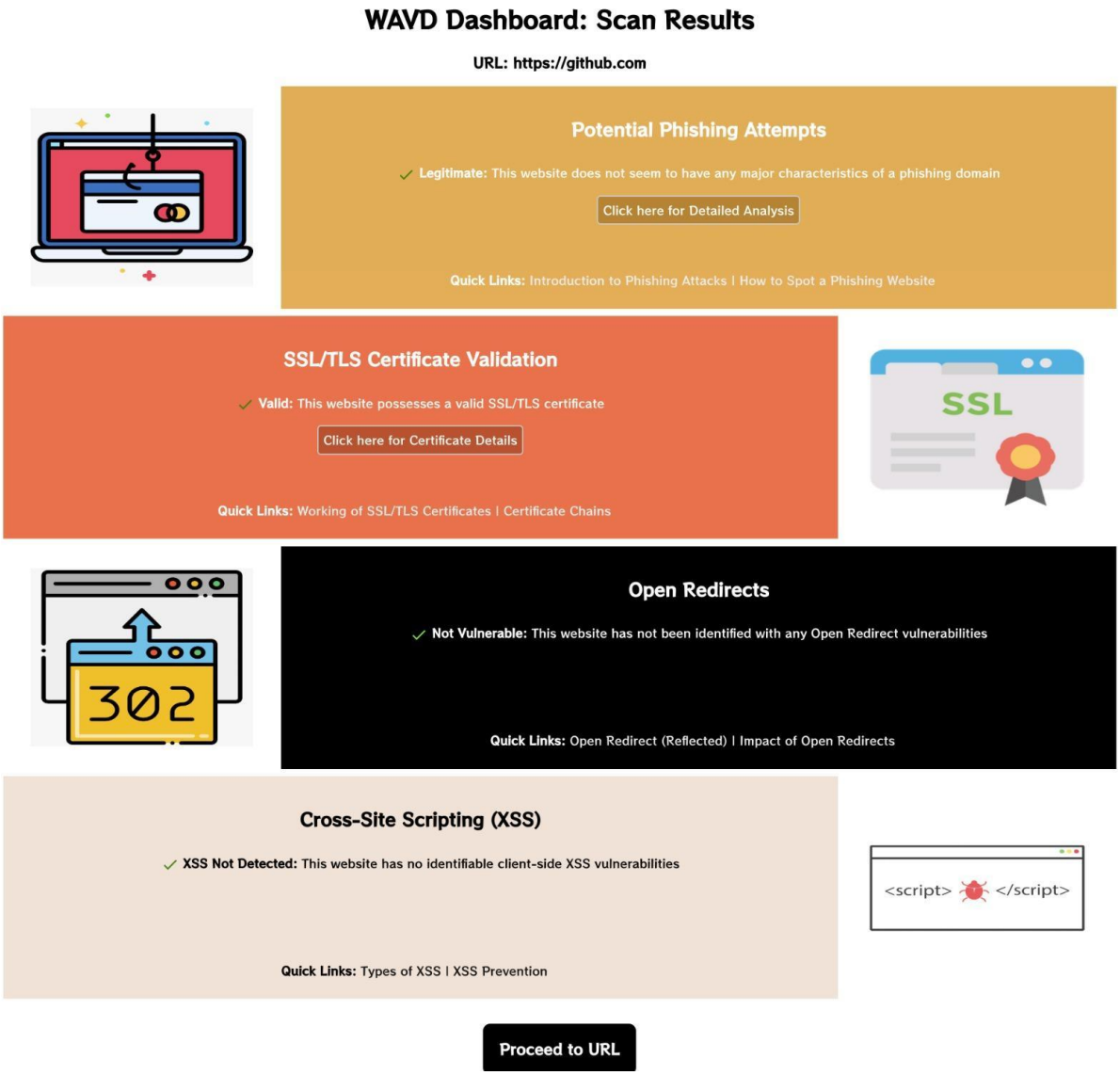


Figure 4.14 No vulnerabilites detected

Figure 4.14 shows an overview of a scan result for the website github.com, indicating that no vulnerabilities have been detected. The panel confirms that the website is legitimate, valid, and not vulnerable to any known security issues or cross-site scripting (XSS) attacks. Based on this information, users can proceed to the website by clicking on the "Proceed to URL" button with a high level of confidence that it is safe to use.

4.2 Analysis

Table 4.1 provides a comparative analysis of four different models used for phishing detection and their corresponding accuracy scores. The models analyzed in the table include XGBoost, Random Forest, SVM, and KNN. The accuracy scores obtained from each model are presented, allowing for a comparison of the effectiveness of each model in detecting phishing URLs. The highest accuracy score is obtained from the XGBoost model, with a score of 98.32%. This indicates that the XGBoost model performs the best among the four models tested.

Based on the comparison, it is determined that XGBoost is the best model for phishing detection. Therefore, the phishing URL dataset can be trained using the XGBoost model to achieve the best results. This information can be beneficial for those developing phishing detection systems, as they can use this table to choose the best model for their specific needs. Additionally, the table highlights the importance of evaluating multiple models to identify the most effective one, as the accuracy scores can vary significantly between different models. Overall, the table serves as a useful tool for those working on phishing detection and can aid in the development of more accurate and effective phishing detection systems.

4.1 Model Comparison Table

Model	Attributes	Accuracy
XGBoost	20	98.32%
Random Forest	17	97.14%
SVM	20	94.62%
KNN	18	86.94%

Chapter 5

Conclusion

In today's digital age, web application security is crucial to protect user data and prevent potential security breaches. The presented system offers a reliable and efficient solution to scan and detect various vulnerabilities in web applications. Its ability to generate detailed reports and provide recommendations for vulnerability mitigation can save valuable time and effort for small-scale industries that may not have the resources to invest in expensive web security tools. By utilizing a proxy server, the system ensures that user privacy and security are maintained during the scanning process. The system's ability to detect multiple types of vulnerabilities, including phishing, cross-site scripting, SSL/TLS certification validation, and open-redirect vulnerabilities, makes it a comprehensive and valuable tool for web application security.

In conclusion, the presented system provides a reliable and efficient solution for web application security that can help prevent potential security breaches and protect user data. Its user-friendly interface, comprehensive vulnerability detection, and recommendations for vulnerability mitigation make it a valuable tool for small-scale industries that want to ensure a secure and reliable web application environment.

References:

[1] Haibo Chen, Junzuo Chen, Jinfu Chen, Shang Yin Yiming Wu, Japing Xu,"An Automatic Vulnerability Scanner for Web Applications" IEEE, 2020.

[2] R. Brenn Mr.Kalyan D Bamane, Vaibhav Gaikwad, Nikhil Ahire, Kunal Sambhe, Chetan Jagtap, "Web Vulnerability Scanner" International Research Journal of Engineering and Technology (IRJET)Vol 7, Issue,03 March 2020.

[3] Pranav Gadekar,Samruddhi Kulkarni, Shalaka Kulkarni "Shruti More, "Automated Web Application Vulnerability Scanner"Vol 8, Issue 8, August 2021.

[4] Bin Wang, Lu Liu, Feng Li Jianye Zhang, Tao Chen Zhenwan Zou, "Research on Web Application Security Vulnerability Scanning Technology" IEEE,2019.

[5] Binny George', Jenu Maria Scaria, Jobin B', Praseetha VM, "Web Application Security Scanner for Prevention and Protection against Vulnerabilities" International Research Journal of Engineering and Technology (IRJET) ,Vol 8, Issue, 05 March 2020

[6] Mohammad Izadi, Vahid Shahrivar, Mohammad Mahdi Darabi,"Phishing Website Detection using Machine Learning Algorithms" International Journal of Computer Applications,20 September 2020.

[7] Rishikesh Mahajan, Irfan Siddavatam, "Phishing Website Detection using Machine Learning Algorithms" International Journal of Computer Applications Volume 181-25, October 2018.

[8] P.S.Sadaphule, Priyanka Kamble,Sanika Mehre, Utkarsha Dhande, Rashmi Savant" Prevention of Website Attack Based on Remote File Inclusion-A survey" International Journal of Advance Engineering and Research Development", 2017.

[9] Smita Patil, Nilesh Marathe, Puja Padiya" Design of Eficient Web Vulnerability Scanner"IEEE,2020.

[10] C. Anley, "Advanced SQL Injection in SQL Server Applications", White paper, Next Generation Security Software Ltd., 2020.

[11] W. G. Halfond, J. Viegas and A. Orso, “A Classification of SQL Injection Attacks and Countermeasures,” College of Computing Georgia Institute of Technology IEEE, 2020.

[12] “https://www.kaggle.com/datasets/shashwatwork/web-page-phishing-detection-dataset”, last accessed on;2/10/2022.

[13] “https://portswigger.net/web-security/cross-site-scripting/cheat-sheet”, last accessed on 29/12/2022.

Publications and Achievements

Publications:

1. Rohan Sharma; Shubham Yadav; Dhiraj Mishra; Akshaya Prabhu “A review on Web Application Vulnerability Scanner”, National Conference on Role of Engineers in Nation Building 2023 (NCRENB - 2023) 3rd – 4th Mar. 2023, VIVA Institute of Technology, Virar, Maharashtra, India, proceedings of NCRENB 2023.

Achievements:

1. Participated in PROJECTATHON 2023 – National level project competition hosted by Atharva College of Engineering, Mumbai, 3rd Apr 2023
2. Participated in IMPERIA – Intercollegiate project competition held at the Computer Engineering Department, VIVA Institute of Technology, Shirgaon, Virar (E), 6th Apr 2023.

Acknowledgment

We would like to express a deep sense of gratitude towards our guide Prof. Akshaya Prabhu, Computer Engineering Department for her constant encouragement and valuable suggestions. The work that we are able to present is possible because of her timely guidance.

We would like to pay gratitude to the panel of examiners for the time, and effort they put into evaluating our work and their valuable suggestions from time to time.

We would like to thank the Project Head of the Computer Engineering Department, Prof. Janhavi Sangoi for her support and coordination.

We would like to thank Head of the Computer Engineering Department, Prof. Ashwini Save and In charge HOD Prof. Sunita Naik, for her support and coordination.

We are also grateful to the teaching and non-teaching staff of the Computer Engineering Department who lend their helping hands in providing continuous support.

Rohan Sharma

Shubham Yadav

Dhiraj Mishra