# C:\Users\SAMUEL\AppData\Local\Temp\ksohtml13828\wps4.jpg

|  |
| --- |
|  |
|  |  |

**KABARAK UNIVERSITY**

**SCHOOL OF SCIENCE, ENGINEERING AND TECHNOLOGY**

**DEPARTMENT OF COMPUTER SCIENCE AND IT**

**COSF 422: RESEARCH PROJECT.**

**PROJECT TITLE.**

**ENHANCING ONLINE SECURITY WITH ADVANCE URL SCANNER**

**JOHN NGEGWA MWAURA**

**BSCSF/MG/3262/09/21**

Project Document Submitted in Partial Fulfillment for an award of Bachelor degree in Computer Security and Forensics at Kabarak University.

**SEPTEMBER-DECEMBER 2024.**

# **Declaration**

I hereby declare this project as my original work and presented to the best of my knowledge. I also declare that this project has never been submitted to any higher learning institution in any capacity.

STUDENT NAME: ……………..

REGISTARTION NO: ……………………..

# SIGNATURE: …………………..

# DATE: ………………………………………

# **Recommendation**

This work was done under my supervision as the university supervisor

Signature………………………………………….

Date……………………………………….

Chrispus Alukwe Akhonya- CEH

Department of computer science and IT

KABARAK UNIVERSITY

# **Acknowledgement**

I would like to thank the almighty God who has been a source of wisdom, inspiration, knowledge and understanding.

I also thank my family, friends and everyone who has been part of this journey to the completion of this project.

I am deeply grateful to Mr. Chrispus Alukwe Akhonya- CEH for his valuable guidance and support throughout this project.

The github link: <https://github.com/john-arch-3333/URL-site-scanner>

# **Abstract**

In the ever evolving digital world where the internet has become an inseparable part of our lives, the need for secure online navigation and access has become a requirement that cannot be ignored. As we navigate the internet we are exposed to a large ground of threat, malicious websites and phishing attempts to malware infected links and redirects to suspicious websites.

The employment and use of traditional URL scanning methods rely on manual analysis which can be time consuming and prone to human errors. The need for an automated solution for efficient and effective URL scanning system has become increasingly important. The proposed system utilizes the VirusTotal API to scan URL links for malware, phishing and other types of threats. The system consists of three primary modules: URL input module, VirusTotal API integration module and report generation module. The URL input module allows users to input URL for scanning. The VirusTotal API integration module sends the input URL to VirusTotal for analysis and retrieves the results. The report generation module presents the scan results by providing a comprehensive report on the scanned links.

The need for secure access of resources on the internet while maintaining the integrity of each user is crucial and critical matters than needs to be addressed. The proposed system aims at addressing this without the user breaking a sweat. The need for user-friendliness is addressed in the system without compromise to the overall performance of the user’s computer. A system that can be easily setup by every user is needed as many have no technical skills to implement the complex technical aspects of other systems. The system ensures simplicity while guaranteeing total security to the end user.

TABLE OF CONTENTS

[**Declaration** i](#_Toc182914367)

[**Recommendation** ii](#_Toc182914370)

[**Acknowledgement** iii](#_Toc182914377)

[**Abstract** iv](#_Toc182914378)

[**1.0 INTRODUCTION.** 1](#_Toc182914379)

[**1.1 BACKGROUND OF THE STUDY.** 1](#_Toc182914380)

[**1.2 PROBLEM STATEMENT.** 2](#_Toc182914381)

[**1.3 OBJECTIVES.** 2](#_Toc182914382)

[**1.3.1 General objective.** 2](#_Toc182914383)

[**1.3.2 Specific objective.** 2](#_Toc182914384)

[**1.4 RESEARCH QUESTIONS.** 2](#_Toc182914385)

[**1.5 SIGNIFICANCE OF THE STUDY.** 2](#_Toc182914386)

[**1.6 SCOPE AND LIMITATIONS OF THE STUDY.** 3](#_Toc182914387)

[**1.6.1 Limitations of the study.** 3](#_Toc182914388)

[**CHAPTER TWO: LITERATURE REVIEW** 4](#_Toc182914389)

[**2.0 Introduction.** 4](#_Toc182914390)

[**2.1 Provision of real time feedback on the safety of search queries.** 4](#_Toc182914391)

[**2.2 Avoidance of unnecessary alerts by minimizing false positives.** 5](#_Toc182914392)

[**2.3 Enhancement of user experience through seamless integration.** 5](#_Toc182914393)

[**2.4 Use of VirusTotal database to improve scanning accuracy.** 6](#_Toc182914394)

[**CHAPTER THREE: METHODOLOGY** 8](#_Toc182914395)

[**3.0 Introduction.** 8](#_Toc182914396)

[**3.1 Research methodology.** 8](#_Toc182914397)

[**3.3 Sample size** 9](#_Toc182914398)

[**3.4 Design diagrams.** 10](#_Toc182914399)

[3.4.1 Context diagram. 10](#_Toc182914400)

[**3.4.2 Use case diagram.** 11](#_Toc182914401)

[**3.5 Research ethics.** 11](#_Toc182914402)

[**4.1 INTRODUCTION** 13](#_Toc182914403)

[**4.2 SYSTEM ARCHITECTURE** 13](#_Toc182914404)

[**4.3 FRONTEND DEVELOPMENT** 15](#_Toc182914405)

[**4.3.1 HTML Structure** 15](#_Toc182914406)

[**4.3.2 CSS Styling** 16](#_Toc182914407)

[**4.4 USER INTERFACE DESIGN** 17](#_Toc182914408)

[**4.5 BACKEND DEVELOPMENT** 19](#_Toc182914409)

[**4.5.1 DATABASE DESIGN** 22](#_Toc182914413)

[**4.6 CODE TESTING** 22](#_Toc182914414)

[**4.7 DEPLOYMENT METHODS** 23](#_Toc182914415)

[**5.1 CONCLUSION** 24](#_Toc182914416)

[**5.2 RECOMMENDATION AND FUTURE WORK** 24](#_Toc182914417)

[**REFERENCES** 27](#_Toc182914418)

[**APPENDICES.** 28](#_Toc182914419)

[**Appendix I – Project schedule.** 28](#_Toc182914420)

[**Appendix II – Budget.** 28](#_Toc182914421)

List of Figures

[Figure 1 Concept Map 7](file:///E:\acadenic\4.1\PROJECT\computer%20project%20final%20continuation.docx#_Toc182914327)

[Figure 2 rapid Application Development Cycle 9](#_Toc182914328)

[Figure 3 Context Diagram 10](#_Toc182914329)

[Figure 4 Use Case Diagrams 11](#_Toc182914330)

[Figure 5.1 front end html 16](#_Toc182914331)

[Figure 6 css styling for the front end 17](#_Toc182914332)

[Figure 7scan result 18](#_Toc182914333)

[Figure 8results 19](#_Toc182914334)

[Figure 9 api intergration 20](#_Toc182914335)

[Figure 10resource management 21](#_Toc182914336)

[Figure 11 22](#_Toc182914337)

LIST OF ABBRIVIATIONS

API……………………………………………………Application Programming Interface

URL………………………………………………….Uniform Resource locator

HTML……………………………………………….Hypertext Markup Language

CSS…………………………………………………..Cascading Style Sheets

RAD………………………………………………….Rapid Application Development

DFD…………………………………………………..Data Flow Diagram

DOM………………………………………………….Document Object Model

UI…………………………………………………….User Interface

**CHAPTER ONE: INTRODUCTION.**

# **1.0 INTRODUCTION.**

As the internet keeps on growing day by day due to its vast applicability in almost every domain of our lives in today’s world, so does the number of malicious actors whose main aim is to steal from unsuspecting users in the internet space. As a result, various security measures need to be taken to safeguard the internet users and I believe my proposed system can be one of the effective solutions. In most cases when users are performing searches on the internet via their browsers, there is lack of real time threat assessment for the user entered search queries and this is what malicious actors take advantage of as they can craft malicious links which users can click/search without knowing they are harmful. The proposed system involves creation of an extension that will be capable of intercepting search bar inputs and send it to VirusTotal for scanning which will then display the results. Users will be able to get immediate feedback on the safety of their search queries and hence they will be able to avoid the malicious links that they often fall for.

# **1.1 BACKGROUND OF THE STUDY.**

The proposed system will be able to safeguard personal data of the internet users as a lot of cases where there is theft of the user’s private information comes as a result of them clicking on malicious links. The extension will strictly focus on the domain of threat detection and browser security. A lot of organizations nowadays deal with a lot of personal data that is collected from their clients and if a security incident arises whereby these data get to be exposed, they stand to face a lot of loses as a result. These organizations can make use of my proposed system by installing it on each and every machine that their workers use to minimize the risk of them getting compromised via a given endpoint as a result of clicking on a suspicious link. The extension will work by hooking into a browser’s search bar events whereby it will extract the input and invoke the VirusTotal API which will be responsible for performing a comprehensive scan on the inputs. It has been noted by various researchers that the use of systems such as the proposed one can significantly lower cases of severe cybersecurity incidents. For example, in the case of the Bangladesh Bank heist whereby the swift system was accessed by the hackers through one of the bank’s infected computer which they compromised by getting an employee to click on a malicious link, had there been an extension installed on the computer similar to the one that I am proposing, then maybe one of the biggest bank heist in history would have not taken place.

# **1.2 PROBLEM STATEMENT.**

Currently, a lot the URL scans are done manually which can be time consuming and error prone because sometimes the list of URLS that need to be scanned is endless and doing so manually can lead to missing out on scanning some of the URLS. There is also lack of real time threat assessment which leaves the internet users vulnerable. My proposed system aims to address this by automating the URL scanning process and providing round the clock threat assessment as the extension will automatically take any input entered on the search bar and subject it to a vulnerability scan with the help of virusTotal API.

# **1.3 OBJECTIVES.**

## **1.3.1 General objective.**

To develop a browser extension that will enhance security and seamlessly integrate with the user search behavior.

## **1.3.2 Specific objective.**

1. To provide real time feedback on the safety of search queries.
2. To avoid unnecessary alerts by minimizing false positives.
3. To enhance user experience through seamless integration.
4. Leveraging VirusTotal database to improve scanning accuracy.

# **1.4 RESEARCH QUESTIONS.**

1. How can the extension be optimized to minimize false positives?
2. What impact on user behavior does real time threat assessment have?
3. Which privacy considerations should be addressed?
4. How can efficient integration with different web browsers be ensured?

# **1.5 SIGNIFICANCE OF THE STUDY.**

The study will combine user experience, security and web development which will offer an interesting challenge in balancing between speed, user interaction and accuracy of the reports produced by the system. Also, with the increasing cyber threats, real time security tools such as the proposed extension are crucial. Organizations that embrace the proposed system will also benefit from improved security posture which will enhance their cyber resilience.

# **1.6 SCOPE AND LIMITATIONS OF THE STUDY.**

The proposed system aims to work with major browsers such as chrome and Firefox. Its main aim will be on URLs but it will also facilitate the scanning of other inputs such as uploaded files. Non browser applications will not be facilitated in this study as well as deep integration with browser internals as this will be beyond the scope of the study.

## **1.6.1 Limitations of the study.**

The extension will involve scrapping of the inputs on the search bar which may result to privacy concerns whereby some users may feel like their privacy is being infringed.

# **CHAPTER TWO: LITERATURE REVIEW**

## **2.0 Introduction.**

In this chapter, I will attempt to expound on what various researchers have already found with regards to the objectives that I have outlined and address the various gaps that have not been adequately addressed with regards to the same. The main focus areas are on how the extension will provide real time feedback on the safety of search queries, how false positives will be minimized by avoiding unnecessary alerts, how user experience will be enhanced through seamless integration of the extension with the browser and finally how the VirusTotal database will be leveraged to improve the scanning accuracy.

## **2.1 Provision of real time feedback on the safety of search queries.**

The design of the extension will be in such a way that whenever a user inputs any data be it a link, an attachment, a file etc on the search bar, this data VirusTotal will be intercepted immediately by the extension and the search queries will be sent to API for analysis. VirusTotal will conduct its analysis on the queries in a matter of milliseconds and send its findings to the user and based on the findings the user will be advised on whether it is safe to proceed with the search if nothing malicious was found or if the search should be aborted immediately if the query is found to be harmful. All of this will be done in a matter of seconds so as to ensure that the risk of getting breached is minimized as much as possible as time is of essence in this scenarios. Web of Trust emphasize that clicking of an unsafe link can lead to malware infections, personal data theft, stress and financial loses and this is what I hope to address with my proposed extension. According to keeper security, in order to stay safe, whenever you receive unsolicited link, always assume it’s malicious and approach it with caution. However, this may be difficult to implement for a user who deals with large volumes of links, attachments on a daily basis and may sometimes accidentally click on the links while in a rush. My extension aims to automate this whole process in such a way that even if a user clicks on a link accidentally, the link will still be scanned and a response given almost instantly to the user on the safety of the link and as a result a lot of companies will be saved from impending attacks of such kinds which will save them billions of dollars in terms of damages had the attack been successful.

## **2.2 Avoidance of unnecessary alerts by minimizing false positives.**

It has been noted that in most security systems, false positives tend to hamper the threat detection process and divert attention from genuine threats which increases the overall risk since responders will end up spending a lot of time in false positives instead of addressing the real threats. To help address this, my proposed web extension will be designed in such a way that it will have the capabilities to dynamically adjust various security thresholds as per the user context and behavior. Tag rules will also be implemented where necessary to help reduce false positives by filtering excessive noise. Various design models for the web extension will also be explored to help decide on the best model which will prioritize usability without sacrificing security since understanding user behavior is very crucial for reducing false positives and enhancing threat detection.

## **2.3 Enhancement of user experience through seamless integration.**

The web extension will be integrated with VirusTotal API which will allow users to smoothly analyze files, links and other items at the comfort of their browsing window. This will ensure that users will not have to leave their tabs and go to a different tab to carry out analysis of their links, files which can be a very frustrating and annoying process for the user. The proposed system will simplify and automate this process in such a way that when a user inputs any data on the search bar, it will be automatically scanned without the user having to go through the hustle of changing tabs to achieve the same which in turn will enhance the user experience and at the same time ensure that security is not neglected. To also ensure that user experience is not tampered with, the web extension will be created using HTML, JavaScript and CSS which are also the primary programming of most web browsers. The usage of this programming languages will help to ensure that when the extension is being integrated with the web browsers, no major complications will arise as a result of two different architectures being implemented. This will help to ensure that users will not experience sudden breakdowns of the extension or malfunctioning of the extension and render them exposed to various threats or cause the browser to not perform at their optimal which in turn will impact the user experience and cause them to lose faith in the purpose and aim of the proposed extension.

## **2.4 Use of VirusTotal database to improve scanning accuracy.**

VirusTotal database is a very valuable asset in enhancing scanning accuracy and ensuring that user’s security is well taken care of which it is of great benefit to both organizations and individuals. This database gathers data from a wide range of threat intelligence feeds, user submission and anti-virus engines which will make the extension to be capable of providing real time threat assessments for files and URLs. VirusTotal is also known for its prowess in identifying various indicators of compromise such as IPs, domains and URLs. The proposed extension aims to take advantage of this to improve the scanning accuracy by cross referencing the indicators of compromise that may be obtained from a user’s search bar against the VirusTotal database.

2.5 Concept map.

Indicators of compromise

File scanning.

URL scanning

Web extension.

Independent variable

Dependent variable

Figure 1 Concept Map

**CHAPTER THREE: METHODOLOGY.**

# **3.0 Introduction.**

In this chapter, I’ll be discussing about the various research methodologies that I will apply in my study. I will discuss in detail about the various approaches, techniques and methods that I will deploy to collect and analyze data applicable in my proposed project. This includes a comprehensive overview of the data collection methods, research design and the software development strategy that I deem best for the study. Finally, this chapter will outline the various applicable design diagrams that will attempt to give a diagrammatic representation of how the system will function as well as the research ethics that will be put into consideration during the development of the system.

# **3.1 Research methodology.**

RAD (Rapid Application Development) is a very popular agile project management strategy used in software development. One of the biggest pros of adopting RAD is its capability to fasten the project completion time, which makes it an attractive option for programmers who work on time constrained environments such as software development. It also allows stakeholders and project managers to be able to keep track of progress and communicate on evolving issues in real time by emphasizing on prototype iterations and reducing on planning time. This in turn results in faster development, effective communication and greater efficiency. To attempt to visualize this methodology, below is a figure that expounds on the RAD cycle and the steps it takes from planning to test.

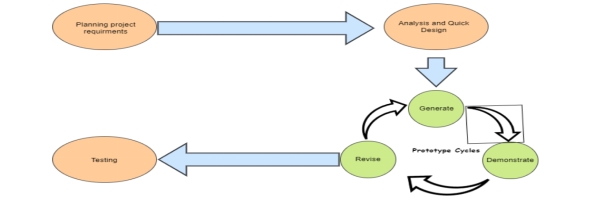


Figure 2 rapid Application Development Cycle

**3.2 Data collection methods.**

To ensure that maximum data is collected to facilitate the development of a well working system, I will be making use of two data collection methods namely primary data collection method and secondary data collection method. By browsing on the internet for various paper works that have been done by previous researchers, I will be able to find the necessary secondary data that will be applicable for my study. For the primary data, I plan to conduct interviews on various stakeholders to help me gather the data that I require.

**3.3 Sample size.**

The sample size used for my study is of 10 personnel whereby five of them are security researchers and the other five are programming experts who would be of great help in terms of giving me feedback on how to ensure that the extension works smoothly with various web browsers such as chrome, Mozilla and Firefox.

# **3.4 Design diagrams.**

## 3.4.1 Context diagram.

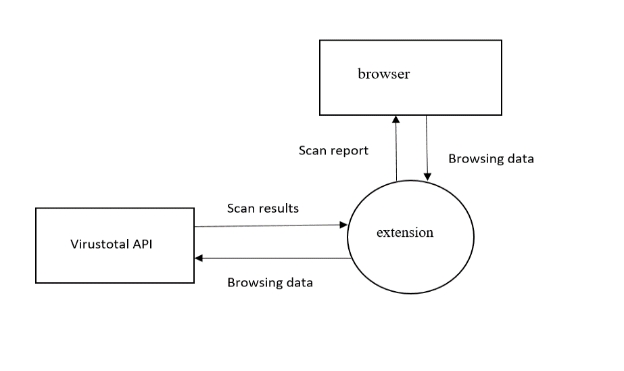


Figure 3 Context Diagram

**3.4.2 Level 1 DFD.**

A level 1 data flow diagram is not applicable in this study as no data will be stored on a database since no authentication mechanism will be put in place and APIs will be used to facilitate the scanning of various browsing data.

## **3.4.2 Use case diagram.**

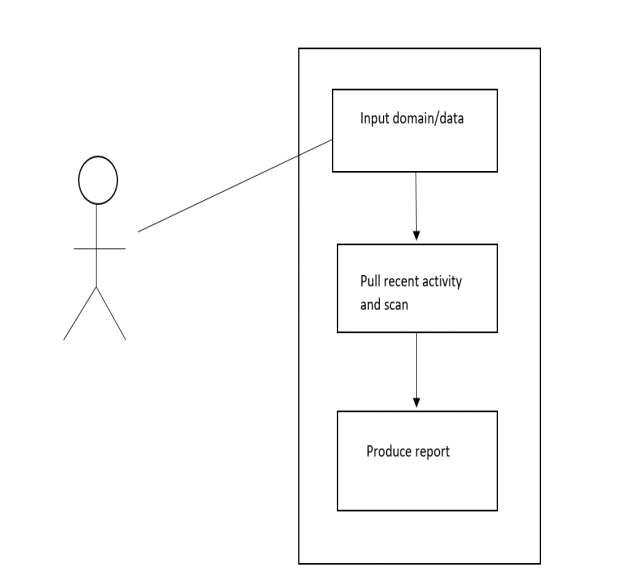


Figure 4 Use Case Diagrams

# **3.5 Research ethics.**

During the course of the study, various research ethics would be taken into consideration and they include:

* Privacy – this will be enforced by ensuring that no personal identifiable information is collected about the user and also limited to no data at all will be collected by the extension.
* Informed consent – before any data is collected from the user, informed consent would have to be obtained from them. It would explain in detail what data will be collected by the extension, how it will be used and any potential risks associated with how it will be used.
* Data security – in case any data will be collected by the extension, various encryption will be applied to them during transit and storage where applicable.
* Transparency – the users will be made fully aware about the purpose of the extension, any third parties that may be involved and the data collection practices.

**CHAPTER FOUR:SYSTEM IMPLEMENTATION AND DEPLOYMENT**

# **4.1 INTRODUCTION**

In this chapter, we will thoroughly explore the URL Site Scanner, providing insight into its functionality and significance for users. By including relevant code snippets and detailed explanations, we aim to clarify how the system operates and the benefits it offers.

We will begin by discussing the system architecture, outlining the various components that work together to create a seamless user experience. Next, we will examine the front-end and back-end development processes, highlighting the technologies and frameworks employed to build the application.

Additionally, we will delve into the database design models, explaining how data is structured and managed within the system to ensure efficiency and reliability. We will also cover code testing practices, illustrating how we maintain high quality and performance through rigorous testing methodologies.

Finally, we will outline the deployment methods used to launch the application, ensuring it is accessible and scalable for users. To conclude, we will discuss potential future work and enhancements that could further improve the system, making it an even more valuable tool for URL scanning.

# **4.2 SYSTEM ARCHITECTURE**

The architecture of the URL Site Scanner is designed to provide an efficient and user-friendly experience while ensuring robust functionality for scanning web pages. This system operates as a web extension, allowing it to analyze the content and scripts loaded on a page with the necessary permissions granted for the entire loaded page.

**Core Components**

**Web Extension Framework:** The URL Site Scanner is built as a browser extension, which enables it to interact directly with web pages. This architecture allows the extension to access the Document Object Model (DOM) of the loaded page, facilitating the analysis of content and scripts. The extension is compatible with the chrome browser as the primary supported browser but further improvemet can be made to support other major flavours of browsers.

**User Interface (UI):** The front-end of the extension is developed using HTML, CSS, and JavaScript. HTML structures the user interface, CSS styles it for an appealing look, and JavaScript handles user interactions and dynamic content updates. The UI is designed to be intuitive, allowing users to view scan results of the loaded web pages and scripts on a tap of a button on the extensions icon on the top-right side of the screen easily and seamlessly. This kind of result display is implemented in order to reduce the number of popup on the screen which will be annoying to the user as each scanned resource with result to a popup. Scan results can be viewed on on need basis.

**API Integration:** A crucial component of the URL Site Scanner is its integration with the VirusTotal API. When a user requests a scan, the extension collects relevant data from the loaded page, including URLs and scripts. This information is then sent to the VirusTotal API, which performs comprehensive checks against its extensive database of known threats. The use of the VIrusTotal API reduces the occurance of false positives for the database of the known threats is extensive,accurate and trusted.

**Popup for Scan Results:** After the API call is made, the extension retrieves the scan results from VirusTotal. A popup interface is used to display these results to the user. This popup provides a clear and concise overview of any detected threats or issues, including details such as the status of the URL, the actual URL scanned , and any relevant comments. The design of the popup ensures that users can quickly understand the scan results without navigating away from their current page.

**Permissions Management:** The URL Site Scanner operates with specific permissions that allow it to access the content of the loaded page. This capability is essential for performing thorough scans, as it enables the extension to analyze all scripts and content that may pose a risk. Users are informed about the permissions required, ensuring transparency and trust.the web extension integrates permissions such as javascript,popups and redirect so that it can properly and seamlessly function. Javascript is the core for the whole project for it relies entirely on javascript for it to perform the API calls and display the results. Popups and redirect is used when user needs to view the scan results of the visited sites. The user is given the ability to manage the permissions the scanner can user based on their own preferences.

**Workflow Overview**

The workflow of the URL Site Scanner can be summarized as follows:

**User Interaction**: The user activates the extension while visiting a web page. This can be done through a simple click on the extension icon in the browser toolbar. The user interface is designed to be straightforward, prompting users to initiate a scan easily. When a user input a search query in the chrome browser search bar the extension is activated in order to gather the search results that is given back by the browser in order to sent it for scanning.

**Data Collection**: Once activated, the extension collects relevant URLs and script data from the page. This step involves parsing the DOM to gather necessary information, such as links and embedded scripts that could potentially be harmful. The extension not only scans the clicked URL but the entire scan results displayed after a search prompt

**API Call**: The collected data is sent to the VirusTotal API for scanning. This process involves formatting the data into a suitable request and handling any necessary authentication with the API. The extension ensures that the request is made securely and efficiently.

**Result Retrieval**: After the API processes the request, the extension receives the scan results from VirusTotal. This step includes error handling to manage any potential issues that may arise during the API interaction, ensuring that the user is informed of the scan's success or failure. A waiting time is given in order for the VirusTotal API may perfom the scan and provide the scan results to the uses.

**Display Results:** A popup is triggered to show the user the scan results, highlighting any potential threats. The popup is designed for clarity, displaying key information such as the scan status, the actual scanned results and the threat level. This allows users to quickly assess the safety of the web page they are visiting.

The architecture of the URL Site Scanner combines modern web technologies and powerful API integration to deliver an effective solution for users concerned about online security. By leveraging the capabilities of the VirusTotal API and a user-friendly interface, the extension provides a valuable tool for scanning web pages and enhancing user safety while browsing the internet. Future enhancements may include additional features such as real-time monitoring, user preferences for scan settings, and expanded reporting capabilities.

# **4.3 FRONTEND DEVELOPMENT**

The front-end development of the URL Site Scanner is centered around creating a user-friendly interface that allows users to easily view scan results. A key feature of this interface is the minimalistic popup that displays the results of the scan, providing essential information in a clear and concise manner. This section will describe the use of HTML and CSS in crafting this popup, ensuring it is both functional and visually appealing.

## **4.3.1 HTML Structure**

The HTML markup for the popup is designed to be simple yet effective. It consists of a main container that displays the scan status and a section for listing the resources loaded on the scanned page. Below is the provided HTML code snippet that outlines this structure:



Figure 5.1 front end html

In this structure, the **<div id="result">** serves as the primary container for displaying the scan status. Initially, a loading message is shown to inform users that the scan is in progress. Once the scan is complete, this section will be dynamically updated with the results. Below this, the **<div id="resource-results">** section is dedicated to listing the resources loaded on the scanned page, providing users with additional context about the content being analyzed.

## **4.3.2 CSS Styling**

The CSS included in the **<style>** tag enhances the visual presentation of the popup, focusing on a clean and minimalistic design. The styles are crafted to ensure that the information is easy to read and visually distinct based on the scan results. Here’s a breakdown of the CSS styles used:



Figure 6 css styling for the front end

The **.safe**, **.warning**, and **.danger** classes are used to visually categorize the scan results. Each class has a distinct background color and text color to indicate the level of threat detected:

**.safe**: Green background for safe sites, indicating no threats.

**.warning**: Yellow background for sites that may have potential issues, alerting users to proceed with caution.

**.danger**: Red background for dangerous sites, clearly signaling that the site poses a significant risk.

The **.loading** class centers the loading message, making it prominent while the scan is in progress.

# **4.4 USER INTERFACE DESIGN**

The user interface (UI) of the URL Site Scanner is designed with simplicity and functionality in mind, ensuring that users can easily navigate and interpret the scan results. The interface is presented as a minimalistic popup that appears when the user activates the extension while browsing a website. This design choice not only maintains focus on the current page but also allows for quick access to critical security information without overwhelming the user.

Upon activation, the popup initially displays a loading message, indicating that the site is being scanned. This loading state is visually centered, providing clear feedback to the user that their request is being processed. Once the scan is complete, the interface dynamically updates to present the results, which are categorized into three distinct status types: safe, warning, and danger. Each status is represented by a specific color scheme—green for safe sites, yellow for those with potential issues, and red for dangerous sites—allowing users to quickly assess the security level at a glance.

The layout is structured to include a primary section for the scan status and a dedicated area for listing the resources loaded on the scanned page. This organization helps users understand not only the overall safety of the site but also the specific components that were analyzed.

Overall, the user interface of the URL Site Scanner prioritizes clarity and ease of use, making it accessible for users of all technical backgrounds. By providing essential information in a straightforward manner, the UI empowers users to make informed decisions about their online safety.

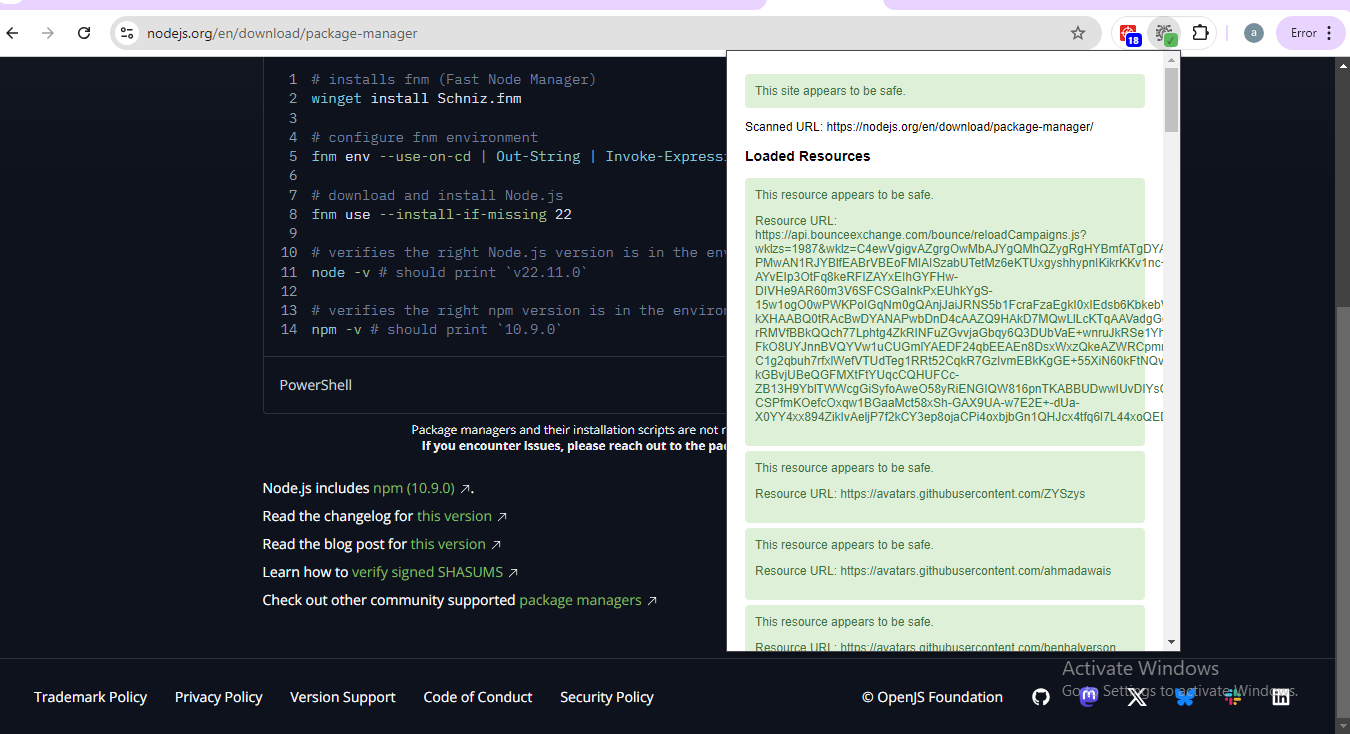


Figure 7scan result

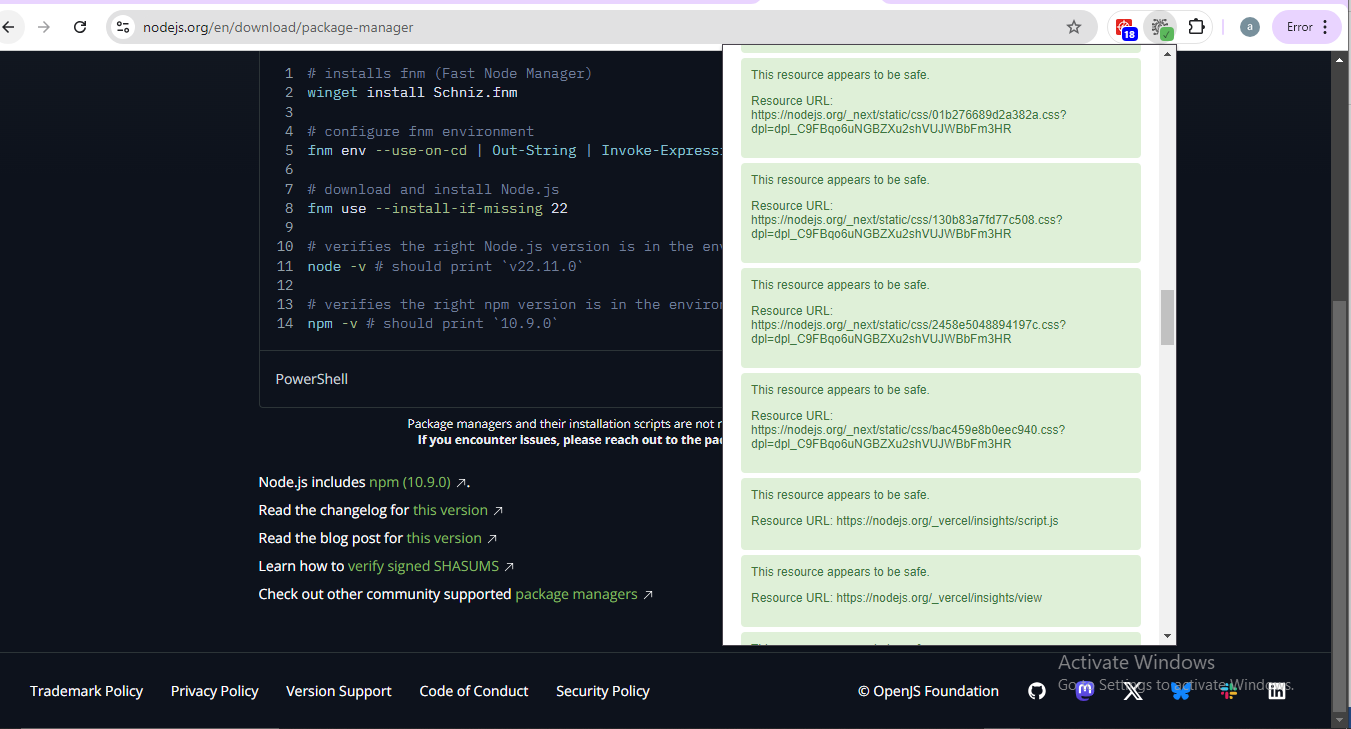


Figure 8results

# **4.5 BACKEND DEVELOPMENT**

# The URL Site Scanner is a Chrome extension designed to enhance user security by scanning URLs and loaded resources for potential threats using the VirusTotal API. The extension operates primarily through background scripts that monitor web requests and manage API interactions efficiently while adhering to rate limits and caching mechanisms**.**

**Key Components**

**API Integration:** The extension utilizes the VirusTotal API to submit URLs for scanning and retrieve reports on their safety. An API key is required for authentication, and the extension maintains a rate limit of 15 seconds between API calls to prevent exceeding usage quotas.

# 

Figure 9 api intergration

**Resource Monitoring**: The extension listens for completed web requests using the **chrome.webRequest.onCompleted listener**. It specifically targets various resource types—such as scripts, stylesheets, images, and sub-frames—by checking their types against a predefined list. When a resource is loaded, it queues a scan for that resource.

**Caching Mechanism**: To optimize performance and reduce redundant API calls, the extension implements a caching system using JavaScript Maps. The **SCAN\_CACHE** stores the results of URL scans, while the **RESOURCE\_CACHE** holds the results of resource scans. Each cache entry is timestamped, allowing the extension to determine whether to reuse cached data or initiate a new API call based on a one-hour expiration policy.

**Resource Scanning**: When a resource is queued for scanning, the extension first checks the cache. If the resource is not cached or has expired, it submits the URL to VirusTotal for scanning. After a brief wait to allow the scan to complete, it retrieves the report and caches the results. If any threats are detected (i.e., if the number of positive detections exceeds zero), the extension notifies the user through a Chrome notification.

**Active Tab Scanning**: The extension also scans the currently active tab when it is activated or updated. It retrieves the URL of the active tab and checks it against the cache. If the cached result is outdated or absent, it makes an API call to fetch the latest scan results. The extension updates the browser action badge to visually indicate the safety status of the scanned URL.

**User Notifications**: For enhanced user awareness, the extension employs Chrome notifications to alert users about malicious resources. The notification includes details such as the resource type, URL, and the number of detections, ensuring users are informed of potential threats.

# Figure 10resource management



Figure 11

The URL Site Scanner combines efficient resource monitoring, API integration, and user notifications to provide a robust tool for assessing web safety. Its design prioritizes performance through caching and rate limiting while delivering critical security information to users in an accessible manner.

# **4.5.1 DATABASE DESIGN**

Databases design and storage of scan results is not within the scope of this project. To enhance the privacy of the use the scan results are not stored by the URL scanner. In latter versions of the project scan result storage can be implemented in order to allow user monitor their scan result later as they can be accessed on their history.

# **4.6 CODE TESTING**

We conducted testing for the URL Site Scanner project by running the system on a laptop to ensure that all established objectives were met. Given the limited number of units available for the system, this approach allowed us to effectively assess the correlation between various components and the overall functionality of the application.

After checking for errors and bugs within the system, we proceeded with tests to confirm that the program operates as intended. We identified several errors during this process, which we were able to address by enlisting the help of friends to review our code. Their fresh perspectives proved invaluable, as they pointed out issues that we had overlooked. Through thorough and repeated code reviews, we successfully eliminated these errors, resulting in a program that now runs smoothly and allows users to access the system without any challenges.

In conclusion, back-end development plays a critical role in the URL Site Scanner project. It is responsible for implementing the core functionalities of the system, managing data storage, facilitating communication with the front end, and ensuring the overall reliability and security of the application.

# **4.7 DEPLOYMENT METHODS**

The deployment of the URL Site Scanner involves a structured process to ensure that the application is delivered smoothly and operates effectively for end users. The deployment method encompasses several key phases, including preparation, testing, and final rollout.

**1. Preparation**

Before deployment, the URL Site Scanner undergoes rigorous development and testing phases. This includes establishing a robust back-end architecture that integrates with external APIs, such as VirusTotal and Bitdefender, to provide accurate and timely security assessments of URLs. The codebase is thoroughly reviewed to identify and rectify any bugs or errors, with peer reviews from team members enhancing code quality and reliability.

**2. Testing**

Testing is a critical component of the deployment process. The application is run on various platforms, including Windows, macOS, and Linux, to ensure compatibility and consistent performance across different environments. Actual scans are conducted on a range of URLs, including safe, malicious, and phishing sites, to validate the accuracy of the scanning results. During this phase, any identified issues are addressed promptly, and multiple iterations of testing are performed to ensure the system operates as intended.

**3. Final Rollout**

Once testing confirms that the URL Site Scanner is functioning correctly, the final rollout is executed. This involves packaging the application as a Chrome extension, which can be easily distributed through the Chrome Web Store. Users can install the extension directly from the store, ensuring a seamless installation process. Additionally, ongoing monitoring and maintenance are established to address any post-deployment issues and to implement updates as needed.

The deployment method for the URL Site Scanner is designed to ensure a reliable and secure user experience. By following a comprehensive approach that includes preparation, thorough testing, and careful rollout, the application is positioned to effectively serve users in assessing the safety of URLs. Continuous feedback and updates will further enhance the system’s functionality and security features.

**CHAPTER 5: RECOMMENDATION AND CONCLUSION**

# **5.1 CONCLUSION**

The URL Site Scanner project has successfully achieved its primary objectives, demonstrating its capability to assess the safety of URLs through integration with leading security services such as VirusTotal and Bitdefender. Throughout the development process, we emphasized rigorous testing and quality assurance to ensure that the application operates reliably across multiple platforms, including Windows, macOS, and Linux.

The testing phase revealed several initial errors; however, through collaborative code reviews and iterative testing, we were able to identify and rectify these issues. Feedback from peers proved invaluable, allowing us to enhance the code quality and functionality of the system. As a result, the URL Site Scanner now runs smoothly, providing accurate assessments of various websites, including safe, malicious, and phishing sites.

In addition to ensuring functionality, the project highlighted the critical role of back-end development in managing data storage, facilitating front-end communication, and maintaining system security. The deployment process was carefully structured, culminating in a user-friendly installation via the Chrome Web Store, which allows users to easily access the extension.

Overall, the URL Site Scanner project not only fulfills its intended purpose of enhancing online safety but also sets a foundation for future enhancements. Continuous monitoring and updates will ensure that the application remains effective against evolving security threats, thereby reinforcing its value to users seeking to navigate the web safely.

# **5.2 RECOMMENDATION AND FUTURE WORK**

As the URL Site Scanner project progresses beyond its initial deployment, several recommendations and potential improvements can be implemented to enhance its functionality, user experience, and overall effectiveness. The following outlines key areas for future development:

**1. Enhanced User Interface (UI) and User Experience (UX)**

While the current interface is functional, improving the UI/UX can significantly enhance user engagement and satisfaction. Recommendations include:

* **Intuitive Design:** Simplifying the layout and navigation can make it easier for users to access features. Implementing a dashboard that summarizes scan results, historical data, and user activity would improve usability.
* **Visual Feedback:** Incorporating visual indicators (e.g., color-coded alerts) for different levels of threat can help users quickly assess the safety of URLs. For instance, green for safe sites, yellow for caution, and red for malicious sites can provide immediate clarity.
* **Accessibility Features:** Ensuring that the application is accessible to users with disabilities is crucial. Implementing screen reader compatibility and keyboard navigation options can broaden the user base.

**2. Expanded Functionality**

To increase the utility of the URL Site Scanner, several features can be added:

* **Batch Scanning:** Allowing users to input multiple URLs for simultaneous scanning would save time and enhance efficiency, especially for businesses that need to assess numerous links at once.
* **Browser Integration:** Beyond a Chrome extension, developing versions for other popular browsers such as Firefox, Safari, and Edge would widen the user base and ensure accessibility across different platforms.
* **Detailed Reporting:** Providing users with comprehensive reports that include detailed analysis of the scanning results, historical trends, and recommendations for safe browsing practices can add significant value.

**3. Improved Security Measures**

Given the evolving nature of online threats, continuous improvement of security features is paramount:

* **Regular Updates:** Establishing a schedule for regular updates to the scanning algorithms will ensure that the system remains effective against the latest threats. Collaborating with cybersecurity experts to stay informed about emerging vulnerabilities can enhance the system's resilience.
* **User Education:** Implementing a section within the application that educates users on common online threats and safe browsing practices can empower them to make informed decisions. This could include tips on recognizing phishing attempts and understanding URL structures.

**4. Community and Feedback Mechanism**

Creating a community around the URL Site Scanner can foster user engagement and continuous improvement:

* **User Feedback Loop:** Implementing a feedback mechanism within the application allows users to report issues, suggest features, and share their experiences. This feedback can be invaluable for future updates and enhancements.
* **Community Forums:** Establishing forums or discussion boards where users can share tips, best practices, and experiences can build a supportive community and encourage user loyalty.

**5. Performance Optimization**

To ensure the application runs efficiently, especially as user demand grows, performance optimization is essential:

* **Load Testing:** Conducting load testing to identify performance bottlenecks will help ensure that the application can handle increased traffic without degradation in performance.
* **Resource Management:** Optimizing the application’s resource usage, such as memory and CPU consumption, will enhance its performance, particularly on lower-end devices.

By implementing these recommendations and focusing on continuous improvement, the URL Site Scanner can evolve into a more robust, user-friendly, and effective tool for assessing URL safety. These enhancements will not only improve user satisfaction but also position the application as a leading solution in the cybersecurity landscape, providing users with the confidence they need to navigate the web safely.

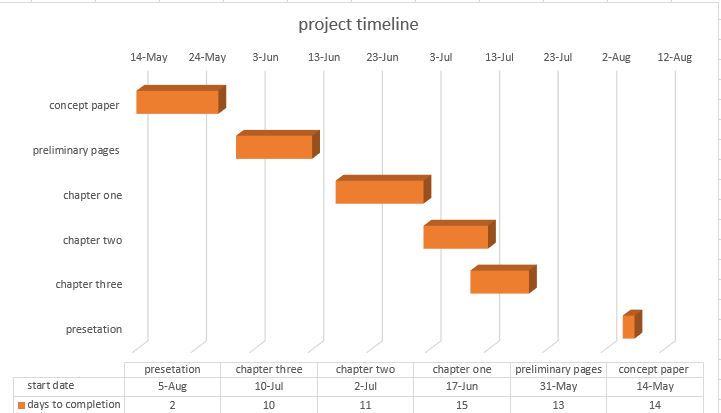
# **REFERENCES**

1. Aranza Trevino (2023, February 09) How to check if a link is safe. Retrieved from <https://www.keepersecurity.com/blog/2023/02/09/how-to-check-if-a-link-is-safe/#:~:text=Clicking%20on%20malicious%20links%20can%20lead%20to%20compromised,it%20is%20or%20by%20using%20a%20URL%20checker>.
2. Hope, P., Walther, B.(2008). Web security testing cookbook. O’Reilly Media
3. Web of trust.(n.d) how to know if a link is safe? Retrieved from <https://www.mywot.com/blog/safe-link>
4. Sharma , A. (2019). Web application security: a beginner’s guide. Packt Publishing. Doi:10.23912/9781789135974
5. Stuttard, D., &Pinto, M.(2011). The web application hacker’s handbook: finding and exploiting security flaws. Wiley
6. W3C. (n.d). the anatomy of a URL: understanding the power of web addresses. Syntax components. Retrieved from

<https://www.w3.org/Addressing/URL/url-spec#syntax-components>

# **APPENDICES.**

## **Appendix I – Project schedule.**



## **Appendix II – Budget.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Number | Item | Quantity | Cost per unit | Total cost |
| 1 | Laptop | 1 | 50,000 | 50,000 |
| 2 | Flash disk (32gb) | 1 | 1,000 | 1,000 |
| 3 | API keys | 1 | 3,000 | 6,000 |
| 4 | Operating system | 1 | 2,500 | 2,500 |
|  |  |  | TOTAL | 59,500 |

.