AN INDUSTRY ORIENTED MINI PROJECT REPORT ON

SAFESURF-Chrome Extension

in the partial fulfillment of the requirements for the award of the degree of

BACHELOR OF TECHNOLOGY

in

CSE (Cyber Security)

Submitted by

E SRIMANI TEJA 22B81A6251

T VIVEKANANDA 22B81A6263

K VIGNESHWAR 22B81A6260

Under the guidance of
G SAHITHI
Assistant Professor

DEPARTMENT OF CSE(Cyber Security)

CVR COLLEGE OF ENGINEERING

(*An Autonomous institution, NAAC Accredited and Affiliated to JNTUH, Hyderabad*)
Vastunagar, Mangalpalli (V), Ibrahimpatnam (M),
Rangareddy (D), Telangana- 501 510

CVR COLLEGE OF ENGINEERING

(*An Autonomous institution, NAAC Accredited and Affiliated to JNTUH, Hyderabad*)
Vastunagar, Mangalpalli (V), Ibrahimpatnam (M),
Rangareddy (D), Telangana- 501 510

DEPARTMENT OF CSE (Cyber Security)

CERTIFICATE

This is to certify that the Industry Oriented Mini Project report entitled **"SAFESURF-Chrome Extension"** Bonafide record of work carried out by **E SRIMANI TEJA(22B81A6251), T VIVEKANANDA(22B81A6263)** and **K VIGNESHWAR(22B81A6260)**submitted to **Dr. C. Raghavendra, Associate Professor** for the requirement of the award of **Bachelor of Technology** in **CSE (Cyber Security)** to the CVR College of Engineering, affiliated to Jawaharlal Nehru Technological University, Hyderabad during the year 2024-2025.

Project Guide

G. Sahithi
Assistant Professor
Department of CSE(CS)

Project Coordinator

Dr. C. Ragahavendra
Associate Professor
Department of CSE(CS)

Head of the Department

External Examiner

CVR COLLEGE OF ENGINEERING

(UGC Autonomous Institution)
Affiliated to JNTU Hyderabad

Vastunagar, Mangalpalli (V), Ibrahimpatnam (M),

Ranga Reddy (Dist.), Hyderabad – 501510, Telangana State

DECLARATION

We hereby declare that the Industry Oriented Mini Project report entitled **"SAFESURF-Chrome Extension"** is an original work done and submitted to CSE (Cyber Security) Department, CVR College of Engineering, affiliated to Jawaharlal Nehru Technological University Hyderabad in partial fulfilment for the requirement of the award of Bachelor of Technology in CSE (Data Science) and it is a record of bonafide project work carried out by us under the guidance of **G. SAHITHI,** Assistant Professor, Department of CSE (Cyber Security).

We further declare that the work reported in this project has not been submitted, either in part or in full, for the award of any other degree in this Institute or any other Institute or University.

Signature of the Student

E SRIMANI TEJA

Signature of the Student

T VIVEKANANDA

Signature of the Student

K VIGNESHWAR

Date:

Place:

ACKNOWLEDGEMENT

We are thankful for and fortunate enough to get constant encouragement, support, and guidance from all **Teaching staff of CSE (Cyber Security) Department** which helped us in successfully completing this Industry Oriented Mini Project.

We are extremely thankful to our internal guide **G. Sahithi,** Assistant Professor, Department of CSE (Cyber Security) for providing support and guidance, which made us complete the Industry Oriented Mini Projecton time.

We thank **Dr. C. Raghavendra**, Project Coordinator and **Mrs. G. Sahithi, Mrs. A. Geetha,** Project Review Committee members for their valuable guidance and support which helped us to complete the Industry Oriented Mini ProjectWork successfully.

We thank Professor and Head of the Department **Dr**. **M. Sunitha,** for giving us all the support and guidance, which made us to complete the Industry Oriented Mini Project duly.

We would like to express heartful thanks to **Dr. H. N. Lakshmi,** Associate Dean, ET Department, for providing us an opportunity and extending support and guidance.

We thank our Vice-Principal **Prof. L. C. Siva Reddy** for providing excellent computing facilities and a disciplined atmosphere for doing our work.

We wish a deep sense of gratitude and heartfelt thanks to our Principal **Dr. K. Rama Mohan Reddy** and the **Management** for providing excellent lab facilities and tools.

TABLE OF CONTENTS

List of Figures

Abstract

In an era where cyber threats are becoming increasingly sophisticated, ensuring a safe browsing experience is paramount. **SAFESURF** is a lightweight Chrome extension developed using **JS** and the **VIRUS-TOTAL API** to enhance online security by providing real-time monitoring and scanning of URLs and downloaded files. Unlike traditional security tools that focus solely on either URL analysis or file scanning, SAFESURF integrates both functionalities to offer **comprehensive protection against phishing, malware, and other cyber threats**.

The extension leverages threat intelligence APIs, including **VirusTotal** and **Google Safe Browsing**, to analyze URLs and file signatures in real time. It employs browser APIs such as **webRequest** for network request interception and **downloads** for file monitoring, ensuring that users are safeguarded against malicious content before exposure. The HTML & JS based user interface provides an intuitive dashboard where users can **view scan results, receive threat notifications, and manage security settings**, including **customizable whitelisting** of trusted URLs and blacklisting of untrusted URLs.

By combining robust backend security mechanisms with a user-friendly interface, SAFESURF aims to bridge the gap in existing browser security tools. It empowers users with **proactive threat detection, real-time alerts, and customizable security controls**, ensuring a **safer and more secure** browsing experience.

CHAPTER 1
1. INTRODUCTION
1.1 Motivation

With cyber threats like phishing and malware increasing rapidly, many users unknowingly visit harmful websites or download malicious files that can compromise their systems. Existing security tools typically focus on either URL scanning or file scanning, but not both. This leaves a security gap, allowing cyber attackers to exploit the weakest point.
SafeSurf was developed to fill this gap by integrating real-time URL and file scanning in a single Chrome extension, offering users a more complete and effective security solution.

Cyber attackers often combine malicious web pages and infected downloads to bypass traditional protection layers. For instance, a phishing site may trick the user into downloading a disguised executable or document, making it essential to have a dual-layered defense system. The lack of such integrated tools inspired the creation of SafeSurf, which offers proactive, real-time alerts using trusted threat intelligence platforms. Additionally, SafeSurf empowers users by allowing them to control blacklist and whitelist settings, adding a customizable touch to their browsing safety.

1.2 Problem Statement

Current security extensions lack the capability to effectively combine both URL scanning and downloaded file scanning in a unified solution.

Many security tools today focus either on scanning URLs or on scanning downloaded files, but they typically do not integrate both functions. URL scanning identifies potentially dangerous websites, while file scanning looks for threats in downloaded files, such as viruses or malware. However, no solution currently offers real-time, integrated scanning for both of these elements together, leaving gaps in overall protection.

This gap in functionality leaves users vulnerable to multi-vector threats, increasing the risk of cyber attacks, data breaches, and system compromises.

Malicious websites and downloaded files often work together to launch attacks. For example, a harmful website might lead to the download of a malicious file, which could infect the system. Tools that only scan URLs or only scan files miss the full picture. Users could be exposed to threats like drive-by downloads or exploit kits, where both website and file components play a role in the attack. This lack of integrated protection increases the risk of attacks going undetected.

The need for an all-in-one security extension that can scan both URLs and downloaded files simultaneously is critical to enhancing user protection.

An integrated solution that scans both URLs and files in real-time would offer more comprehensive security. Such a tool could block access to dangerous sites and also scan any downloaded files for threats.

1.3 Project Objectives

The main goal of the SafeSurf Chrome extension is to provide a secure, real-time protection mechanism for users while browsing the web. This is achieved by integrating URL and file download scanning into a single, lightweight, and efficient browser extension. The following are the key objectives of this project:

1. **Real-Time Threat Detection:** To detect and alert users about potentially harmful URLs and downloadable files in real time using the VirusTotal API.
2. **Dual-Function Security:** To offer an integrated approach to web security by combining both URL and file scanning, which is often missing in traditional browser security tools.
3. **Lightweight Extension:** To ensure that the extension remains lightweight and does not slow down the browsing experience, making it suitable for daily use.
4. **User-Friendly Interface:** To develop a simple and intuitive interface where users can manage threat reports, whitelist trusted sites, and get real-time alerts.
5. **Customizable Security Settings:** To allow users to configure whitelists and blacklists based on their preferences, adding flexibility to the extension.
6. **Leverage Threat Intelligence:** To make use of reputable threat intelligence platforms like VirusTotal and Google Safe Browsing to enhance detection accuracy and stay updated with the latest threats.
7. **Encourage Proactive Web Safety:** To educate and empower users by proactively notifying them of risky interactions online rather than reacting after a threat has been executed.

1.4. Project Report Organization

This project report is structured into the following chapters to present a comprehensive view of the SafeSurf Chrome Extension:

- **Chapter 1: Introduction** – Provides background information, motivation behind the project, the problem statement, objectives, and the overall structure of the report.

- **Chapter 2: Literature Survey** – Discusses existing tools and technologies related to web security, and highlights their limitations which SafeSurf aims to overcome.

- **Chapter 3: Software & Hardware Specifications** – Lists the software and hardware requirements for implementing the SafeSurf extension. Also includes detailed project objectives and the motivation behind the solution.

- **Chapter 4: Proposed System Design** – Describes the system's overall architecture, the methods used, and includes diagrams such as use case, activity, sequence diagrams, and the technology stack involved.

- **Chapter 5: Implementation & Testing** – Details the actual implementation of the extension, screenshots of the user interface, and testing methods and results.

- **Chapter 6: Conclusion & Future Scope** – Summarizes the outcomes of the project and proposes possible enhancements for future versions of SafeSurf.

CHAPTER 2
2. LITERATURE SURVEY
2.1 Existing Work

Cyber threats are evolving, with malicious websites often leading to harmful file downloads. However, most existing browser security extensions focus on either **URL scanning** or **file scanning**, leaving users vulnerable to multi-vector attacks.

Some well-known extensions include:

- **PIXM** – Uses AI to detect phishing attacks in real-time but lacks file scanning.
- **PhishDetector** – Identifies phishing websites but does not analyze downloaded files.
- **AI Phishing Assistant** – Machine learning-based phishing detection but no file scanning.
- **OPSWAT File Security** – Scans downloaded files for malware but does not monitor URLs.
- **NoPhish Phishing detection:** Detect phishing websites in real-time using machine learning techniques but does not analyze downloaded files.

Need for SAFESURF

To address this gap, **SAFESURF** integrates **both URL and file scanning** in a single browser extension. It uses **VirusTotal API** to protect users in real-time. The extension provides **instant alerts, customizable whitelisting, and seamless protection** against phishing, malware, and other cyber threats, ensuring a safer browsing experience.

2.2 Limitations of Existing Work

Despite the availability of these security tools, existing browser extensions have several limitations that leave users vulnerable to modern cyber threats. The primary drawback is that most security extensions focus on **either URL scanning or file scanning**, failing to provide a **unified, real-time protection mechanism**. Malicious websites often lead to harmful file downloads, but without an integrated security approach, users remain exposed to **multi-stage cyberattacks** such as drive-by downloads and phishing-based malware infections.

Moreover, many of these tools rely heavily on **static blacklists**, which can quickly become outdated as cybercriminals continuously generate new phishing domains and malware variants. This limits their effectiveness against **zero-day threats** and newly emerging attack techniques. Additionally, some security extensions require **manual scanning** instead of offering automated, real-time protection, making them less efficient for proactive threat detection.

Another significant limitation is the **lack of user customization options**. Many existing solutions do not allow users to **whitelist trusted websites or adjust scanning settings**, which can lead to unnecessary warnings or false positives. Furthermore, some tools may introduce **performance overhead**, slowing down browsing speed due to inefficient scanning mechanisms.

Given these challenges, there is a need for a **comprehensive security extension** that integrates **real-time URL and file scanning**, leveraging threat intelligence APIs and AI-driven threat detection to offer **seamless, automated protection** against phishing, malware, and other web-based attacks.

Extension Name	URL Scanning	File Scanning	Real-time Protection	Open Source	Limitation
PIXM	✅	❌	✅	❌	Detects phishing in real-time but doesn't scan downloaded files.
PhishDetector	✅	❌	❌	❌	Identifies phishing websites, no real-time protection or file analysis.
AI Phishing Assistant	✅	❌	✅	❌	Machine learning-based detection but no file scanning.
OPSWAT File Security	❌	✅	✅	❌	Scans only files for malware; does not monitor or analyze URLs.
NoPhish	✅	❌	✅	❌	Detects phishing URLs with ML but no support for analyzing downloaded content.

CHAPTER 3
3. SOFTWARE & HARDWARE SPECIFICATIONS
3.1 Software Requirements

This section includes the software tools, technologies, and APIs used in the development and deployment of the SafeSurf Chrome Extension:

- **Operating System:** Windows 10 or later / Chrome OS / Linux
- **Web Browser:** Google Chrome (latest version)
- **Languages Used:** HTML, CSS, JavaScript
- **APIs:**
    - VirusTotal API (for threat detection)
- **Code Editor:** Visual Studio Code
- **Version Control:** GitHub (for collaborative development and version management)
- **API function calls:** Javascript API validation and function calls
- **Chrome Developer Tools:** For debugging and extension testing

These software tools are necessary to build a functional, responsive, and secure extension for web threat detection.

3.2 Hardware Requirements

To ensure the smooth operation of the SafeSurf extension during development and user testing, the following minimum and recommended hardware specifications were considered:

- **Processor:**
  - Minimum: Intel Core i3
  - Recommended: Intel Core i5 or higher

- **RAM:**
  - Minimum: 4GB
  - Recommended: 8GB or higher

- **Storage:**
  - Minimum: 250MB of free space (for project files and supporting tools)
- **Display:**
  - Minimum resolution of 1366x768 (for testing the UI in various screen sizes)

- **Internet Connectivity:**
  - Required for real-time scanning using external threat intelligence APIs like VirusTotal

CHAPTER 4
4. PROPOSED SYSTEM DESIGN
4.1 Proposed Methods

SafeSurf is a lightweight and efficient Chrome extension that provides dual-layer protection by scanning both URLs and downloaded files in real-time. It utilizes trusted threat intelligence sources such as VirusTotal and Google Safe Browsing to protect users during web browsing.

Key Methodologies:

1. Real-Time Monitoring

- Continuously tracks visited URLs and downloaded files.
- Uses Chrome's webRequest and downloads APIs to detect activity.

2. Threat Detection Using APIs

- **URL Scan:** Captures and sends URLs to VirusTotal. Alerts the user instantly if flagged.
- **File Scan:** Computes SHA256 hash of downloaded files and checks against VirusTotal.

3. User Alerts & Notifications

- Displays alert popups if any threat is detected, preventing interaction.

4. Whitelist/Blacklist Support

- Users can define trusted and blocked URLs to refine protection.

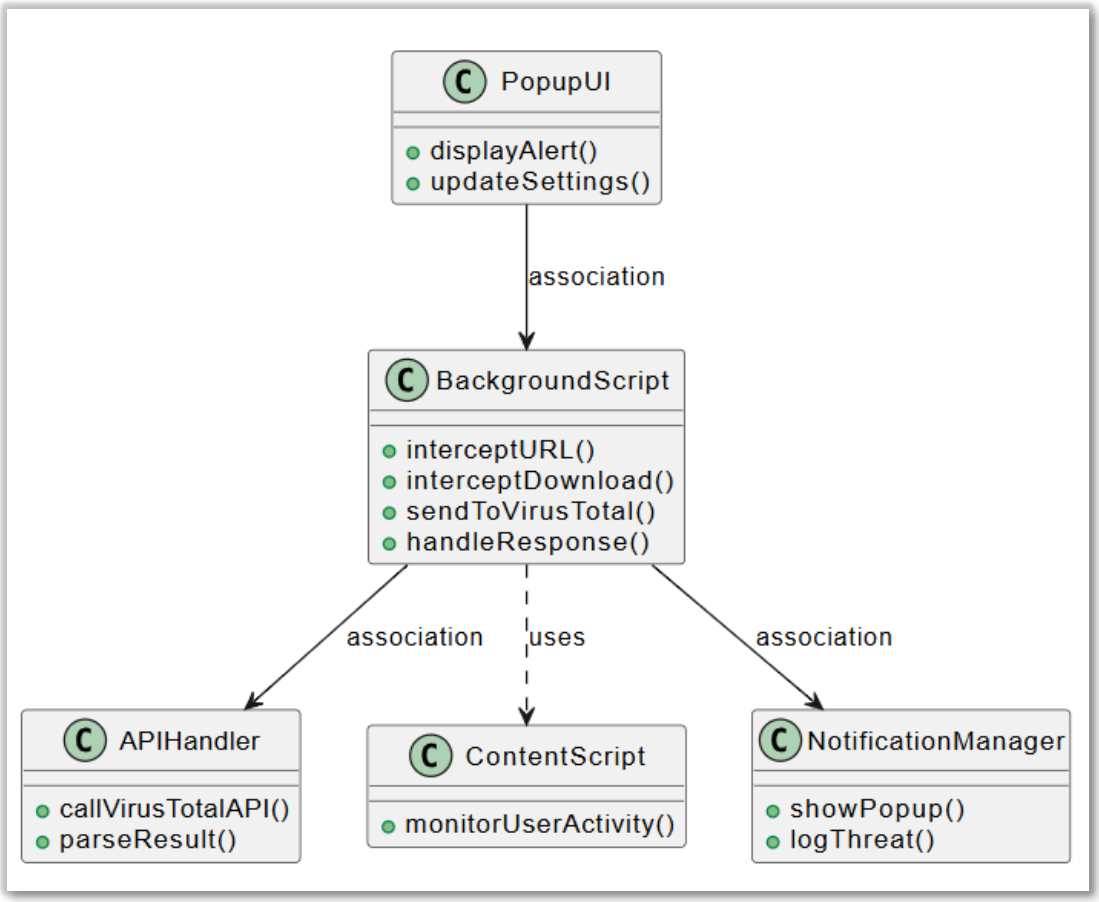
5. Lightweight and Responsive

- Asynchronous scanning to ensure smooth browsing.
- Efficient resource usage to avoid lag.

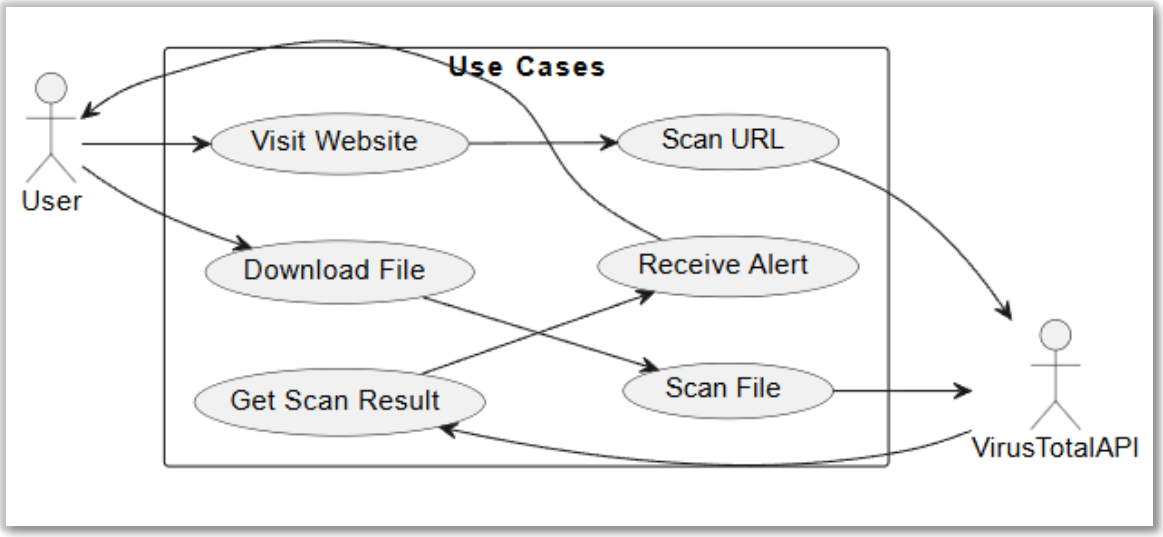
6. Component-Based Architecture

- **Content Scripts** – Capture browsing data.
- **Background Scripts** – Handle threat checks and logic.
- **Popup UI** – User interactions and settings.

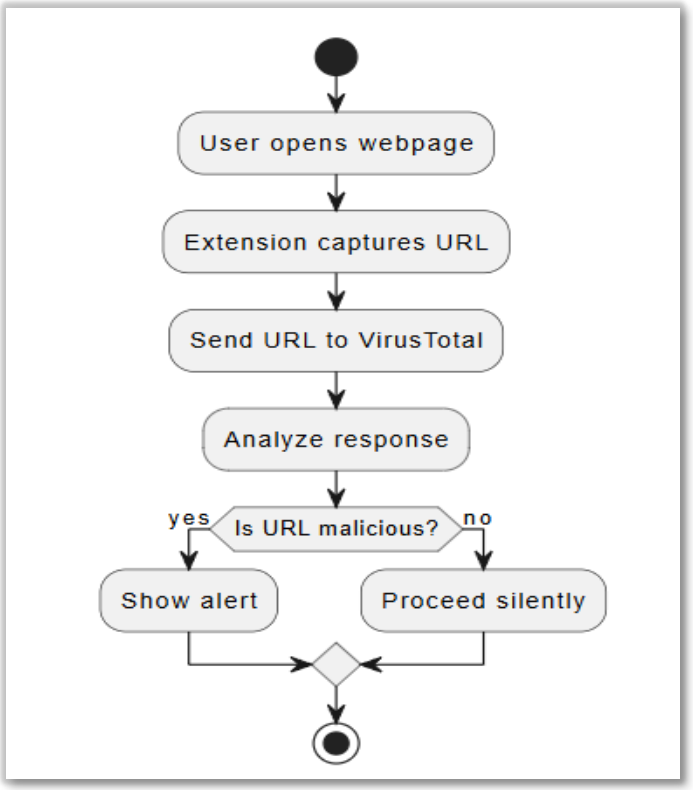
4.2 Class Diagram



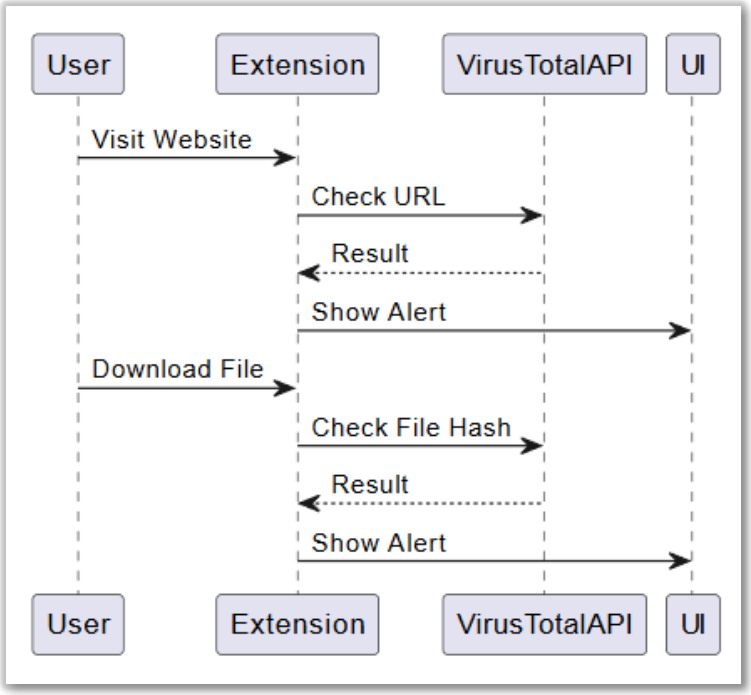
4.3 Use case Diagram



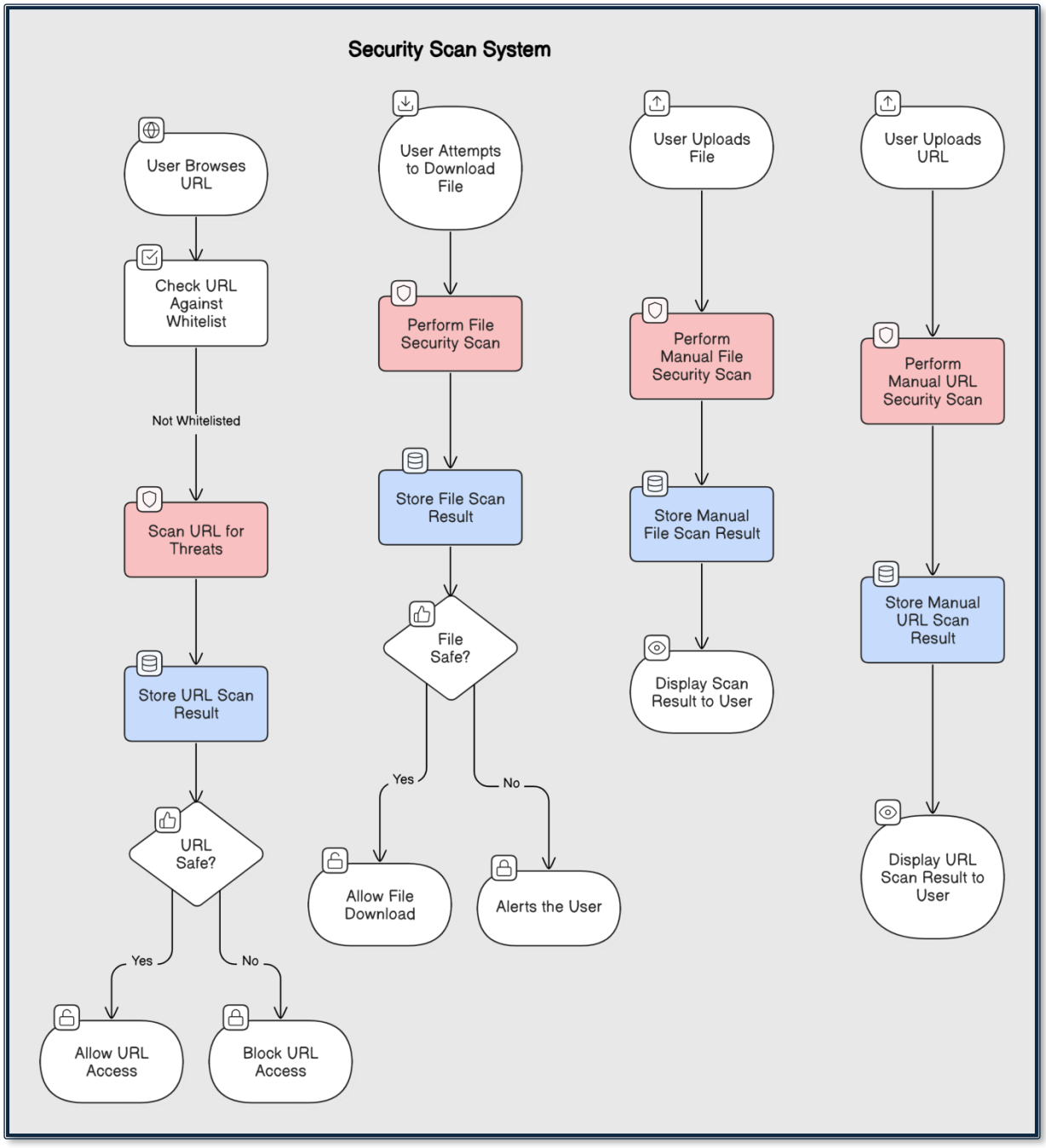
4.4 Activity Diagram



4.5 Sequence Diagram



4.6 System Architecture



4.7 Technology Description

SafeSurf is developed using modern web technologies and Chrome extension APIs to provide real-time threat detection while maintaining performance and usability.

Chrome Extension APIs

- **WebRequest API**: Intercepts and monitors outgoing web requests to capture URL traffic.
- **downloads API**: Detects and handles file download events.
- **runtime & storage APIs**: Manages communication between scripts and stores local preferences like whitelist/blacklist.

Languages & Tools

- **JavaScript**: Core logic for API interaction and asynchronous scanning.
- **HTML/CSS**: For designing the extension's popup interface and user alerts.
- **Manifest V3**: The structure file defining permissions and background script configuration.

External Security Services

- **VirusTotal API**: Used to analyze both URLs and file hashes (SHA256) using a large database of known threats.

Modular Architecture

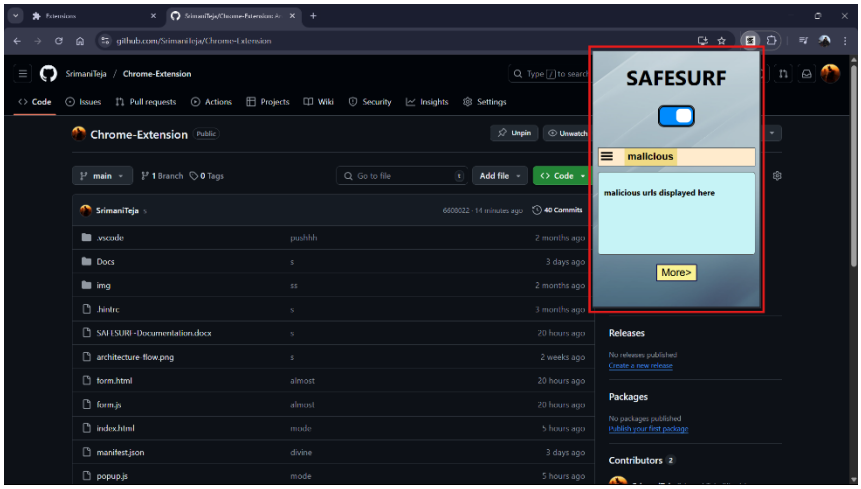
- **Content Script**: Detects user interaction with pages.
- **Background Script**: Manages the scanning logic and API integration.
- **Popup UI**: Interfaces with the user to show details.

Benefits of This Stack

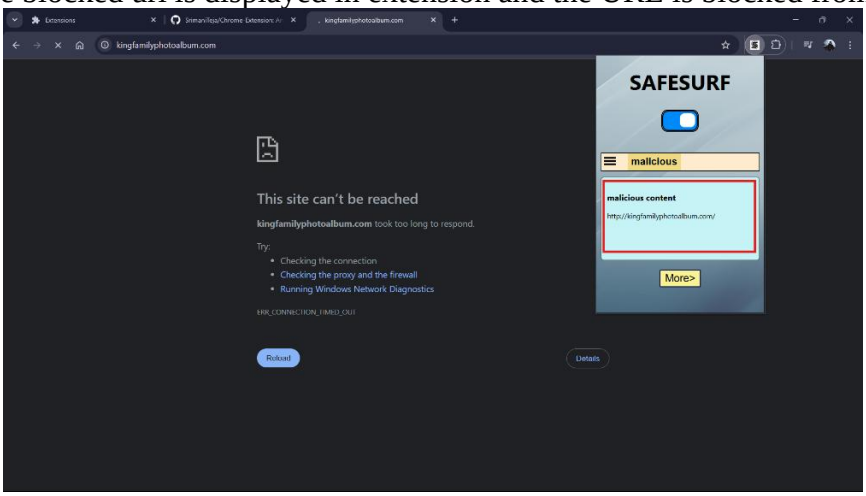
- Lightweight and fast with minimal impact on browser performance.
- Real-time detection powered by globally updated threat databases.
- User-friendly and customizable through built-in storage.
- Easily extendable to support other browsers and additional features.

CHAPTER 5
Implementation & Testing
5.1 Frontend Screenshots

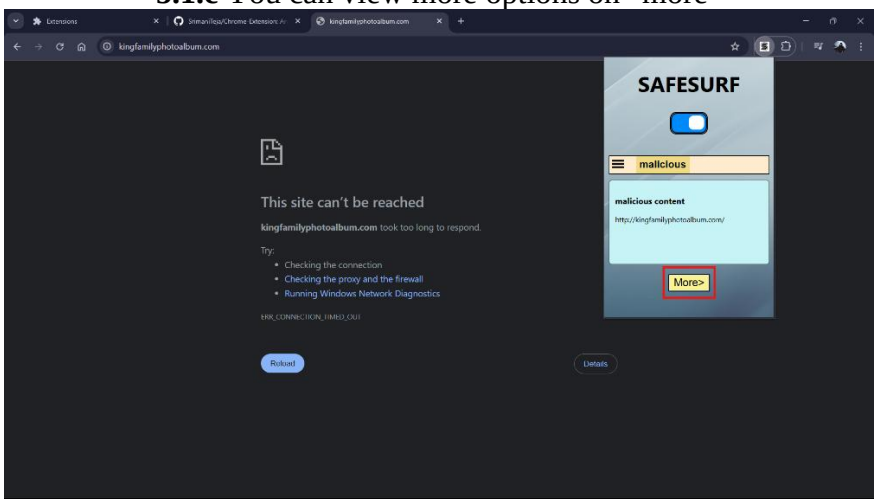
5.1.a The SafeSurf extension tab



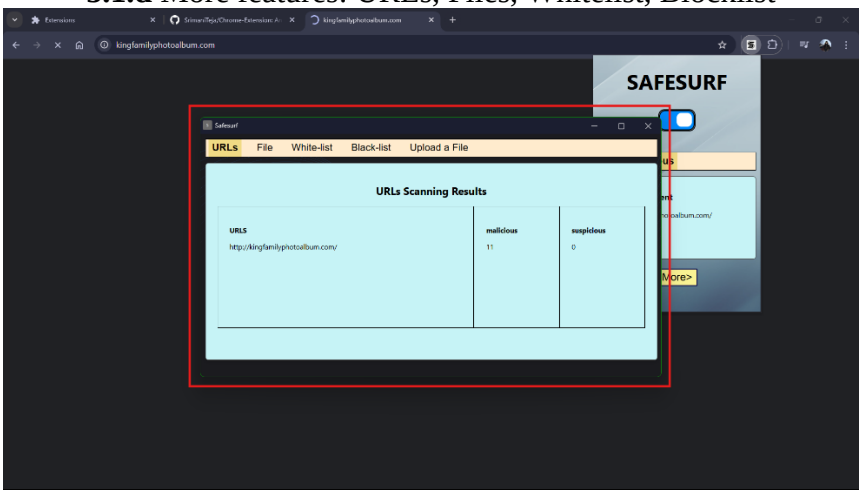
5.1.b The blocked url is displayed in extension and the URL is blocked from loading



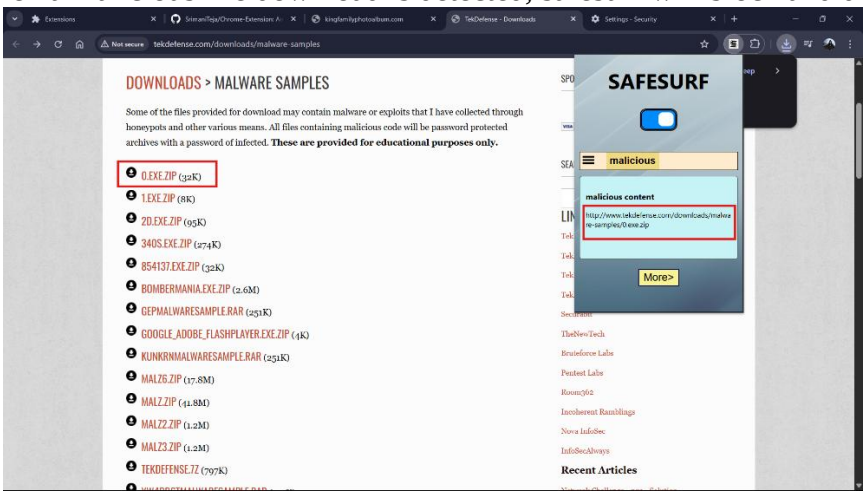
5.1.c You can view more options on "more"



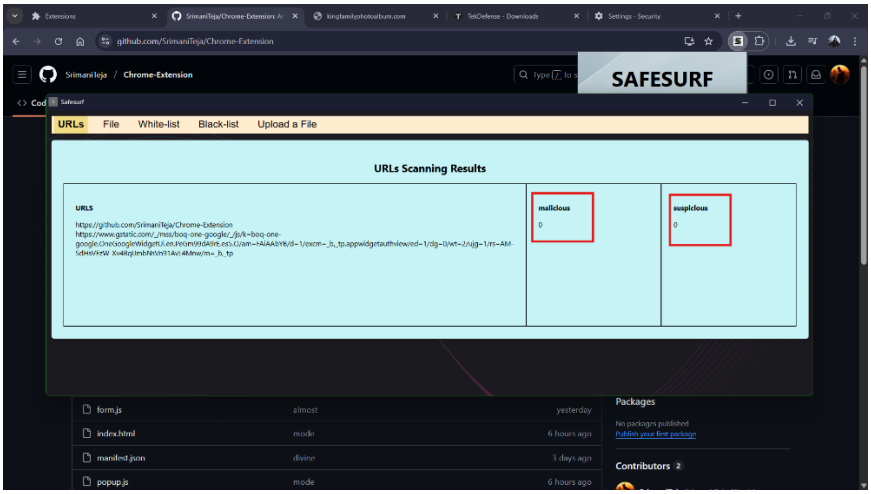
5.1.d More features: URLs, Files, Whitelist, Blocklist



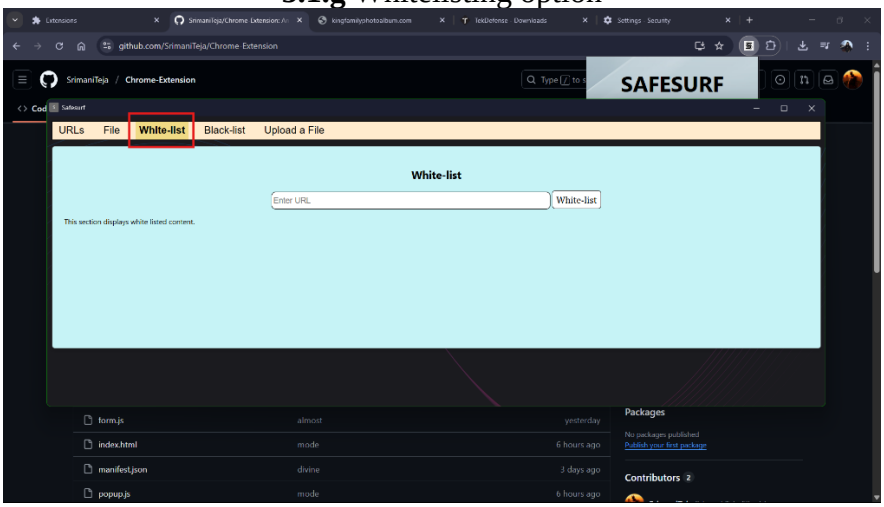
5.1.e When a malicious file download is detected, safesurf will block and displays it



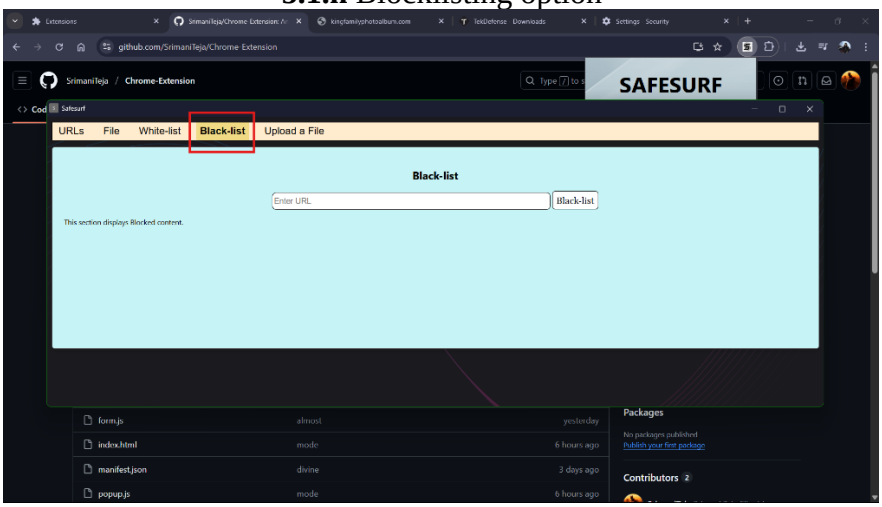
5.1.f Safe URL score



5.1.g Whitelisting option



5.1.h Blocklisting option

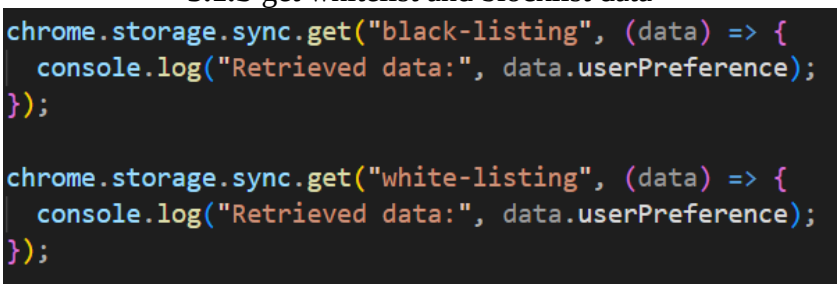


5.2 Backend Code Snippets

5.1.a manifest.json

```
{} manifest.json > [ ] permissions
    VIVEKANANDA, 3 days ago | 2 authors (You and one other) | GetBot AI:
1   {
2     "name": "Safe-Surf",
3     "description": "Base Level Extension",
4     "version": "1.0",
5     "manifest_version": 3,
6     "permissions": [
7         "activeTab",
8         "tabs",
9         "webRequest",
10        "webNavigation",
11        "management",
12        "scripting",
13        "downloads",        VIVEKANANDA, 3 days ago •
14        "storage"
15    ],"host_permissions": [
16        "<all_urls>"
17    ],
18    "action": {
19      "default_popup": "index.html"
20      , "default_icon": "./img/logo2.png"
21    },
22    "background": {
23      "service_worker": "service.js",
24      "type": "module"
25    }
26  }
```

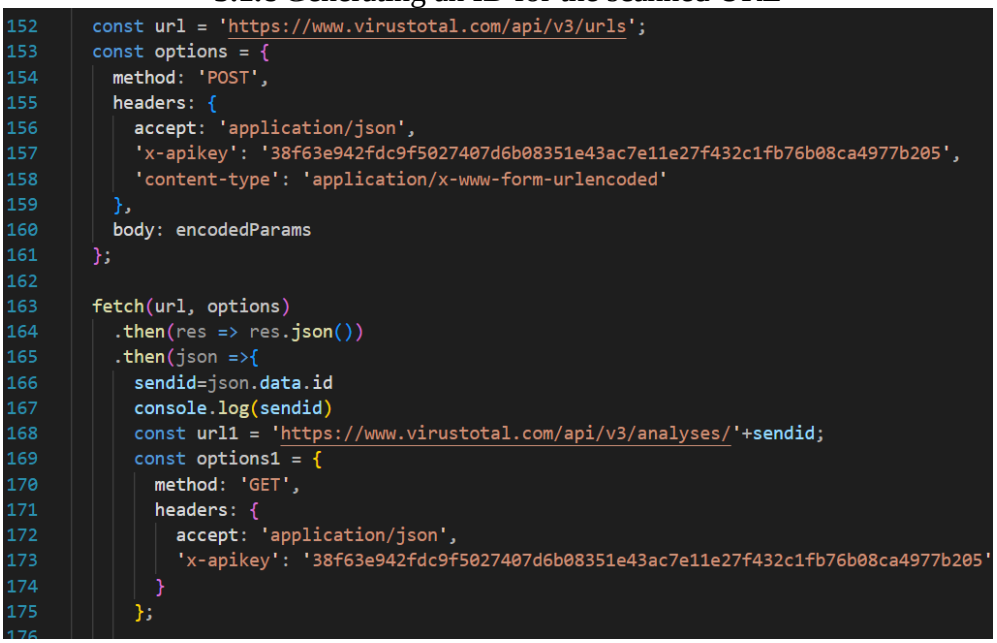
5.1.b get whitelist and blocklist data



```
chrome.storage.sync.get("black-listing", (data) => {
  console.log("Retrieved data:", data.userPreference);
});

chrome.storage.sync.get("white-listing", (data) => {
  console.log("Retrieved data:", data.userPreference);
});
```

5.1.c Generating an ID for the scanned URL



```
152  const url = 'https://www.virustotal.com/api/v3/urls';
153  const options = {
154    method: 'POST',
155    headers: {
156      accept: 'application/json',
157      'x-apikey': '38f63e942fdc9f5027407d6b08351e43ac7e11e27f432c1fb76b08ca4977b205',
158      'content-type': 'application/x-www-form-urlencoded'
159    },
160    body: encodedParams
161  };
162
163  fetch(url, options)
164    .then(res => res.json())
165    .then(json =>{
166      sendid=json.data.id
167      console.log(sendid)
168      const url1 = 'https://www.virustotal.com/api/v3/analyses/'+sendid;
169      const options1 = {
170        method: 'GET',
171        headers: {
172          accept: 'application/json',
173          'x-apikey': '38f63e942fdc9f5027407d6b08351e43ac7e11e27f432c1fb76b08ca4977b205'
174        }
175      };
176
```

5.1.d Fetching results using the ID

```
        fetch(url1, options1)
          .then(res => res.json())
          .then(json => {
            console.log(json.data.attributes.stats)
            safetydata.push(json.data.attributes.stats.malicious)
            safetydata.push(json.data.attributes.stats.suspicious)
            if(json.data.attributes.stats.malicious>0 || json.data.attributes.stats.suspicious>0){
              chrome.action.openPopup();
              blockedurls.push(sendurl)
              chrome.storage.local.set({ blocked: blockedurls}, () => {
                console.log("blocked",blockedurls);
              });

            }
            chrome.storage.local.set({ data: safetydata }, () => {
              console.log("data:",safetydata);
            });
          })
          .catch(err => console.error(err));
      })
      .catch(err => console.log(err));
  }
```

5.1.e Syncing the variables

```
function updateStoredData() {
  chrome.storage.local.set({ message: allurls }, () => {
  });
}

// Update data whenever the popup is opened
chrome.runtime.onMessage.addListener((message, sender, sendResponse) => {
  if (message.action === "updateData") {
      updateStoredData();
      sendResponse({ status: "Data updated!" });
  }
});



chrome.runtime.onMessage.addListener((message, sender, sendResponse) => {
    if (message.action === "whitelist") {
      let updatedValue = message.newValue;
      whitelist.push(updatedValue)
    }
    else if(message.action === "blacklist") {
      let updatedValue = message.newValue;
      blacklist.push(updatedValue)
    }

    console.log("whitelist",whitelist,"blacklist",blacklist)
});
```

CHAPTER 6
Conclusion & Future Scope

SafeSurf offers a modern and efficient solution for secure web browsing by combining real-time URL and file scanning into a single Chrome extension. By leveraging crowd-sourced threat intelligence APIs like VirusTotal and Google Safe Browsing, the extension can detect threats before they reach the user. With its asynchronous design, SafeSurf operates smoothly in the background without compromising user experience. Its modular architecture and lightweight design make it both scalable and highly maintainable.

Future Scope

Cross-Browser Compatibility: Extend support to Firefox, Microsoft Edge, and other Chromium-based browsers.

User Reporting System: Allow users to manually report suspicious URLs or files to improve community defense.

Scheduled Scans: Add a feature to periodically scan saved URLs or downloaded files in background.

Enhanced Whitelist/Blacklist Controls: Introduce category-based filtering and auto-suggestion for safe domains.

REFERENCES

1. **VirusTotal (2004).** A multi-engine scanning platform for detecting malware.

- Aggregates antivirus scans and threat intelligence from multiple vendors to analyze URLs and files for potential malware and malicious activities.

2. Virustotal File Scanning (2004). Cloud-based malware scanning for downloads.

- Provides a centralized file scanning service that allows users to check downloaded files against multiple antivirus engines before execution.

3. PhishTank (2006). A community-driven phishing database.

- Maintains a publicly accessible list of reported phishing websites, helping users and security tools detect fraudulent web pages.

4. Avast Online Security (2012). Browser extension for real-time URL threat detection.

- A security extension that warns users about potentially unsafe websites by leveraging blacklists and heuristic analysis.

5. Malwarebytes Browser Guard (2019). Enhancing browser security against malicious content.

- A browser extension designed to block malware, trackers, and phishing websites, but lacks file analysis features.

6. Lippmann, R., et al. (2021). Layered security approaches for web-based threats.

- Discusses multi-layered security strategies that combine URL scanning, sandboxing, and network monitoring for robust web protection.

7. Z. Chen, et al. (2022). A real-time threat detection framework for browsers. IEEE Transactions on Cybersecurity.

- Proposes a framework that integrates network traffic monitoring with browser security to detect malicious activities in real time.

APPENDIX

The entire source code of this project is available on GitHub - **Source Code**