



**Loughborough  
University**

**Chrome extension for  
cyber security awareness**

by Rajeev Jhaj  
Student ID: B919630

BSc Computer Science

COC251  
Department of Computer Science  
Loughborough University

Supervisor: Dr Asma Adnane  
Submitted 4<sup>th</sup> May 2022

# **Abstract**

Currently, there is a large gap in knowledge of cyber security among the general public. With the growth of technology, more and more people are using the internet for personal use and for work. Therefore risks involving cyber attacks have risen with users being affected personally and businesses being affected financially, causing massive disruption to everyday life. There is a huge problem of people being unknowing victims of cyber attacks, there is an urgent need to find reliable trusted sources of information.

I aim to fill that gap with a convenient Chrome extension to simply inform and reduce the chance of cyber attacks. Users will be able to view a rating of the website's security and those who are new to cyber security can learn new skills to determine the safety of a website. This Chrome extension called 'Security Surf' is freely available on the Chrome Web Store.

# **Acknowledgements**

A special thank you to Dr Asma Adnane for her expert advice and guidance throughout the project. I would also like to say thank you to those who took the time to provide me with their feedback to improve the design and functionality throughout and at the end of the project.

# Contents

<b>1</b>	<b>Introduction</b>	<b>7</b>
1.1	Project Definition . . . . .	7
1.2	Project Brief . . . . .	7
<b>2</b>	<b>Literature Review</b>	<b>9</b>
2.1	Requirements in cyber security . . . . .	9
2.1.1	The popularity of internet browsing . . . . .	10
2.1.2	Uses of internet browsing . . . . .	10
2.2	Cyber security risks of internet browsing . . . . .	10
2.2.1	HTTPS . . . . .	11
2.2.2	Network analysis . . . . .	11
2.2.3	SQL injection and XSS attacks . . . . .	11
2.2.4	Phishing . . . . .	11
2.3	Safe internet browsing . . . . .	12
2.3.1	Online website analysis . . . . .	12
2.3.2	Applications . . . . .	14
2.3.3	Other tools . . . . .	14
2.4	Website rating and evaluation . . . . .	15
2.4.1	How is a website evaluated? . . . . .	15
2.4.2	Factors for rating a secure website . . . . .	16
2.5	Conclusion . . . . .	16
2.5.1	Where we need to improve . . . . .	16
<b>3</b>	<b>Plan</b>	<b>18</b>
3.1	Stakeholders . . . . .	18
3.2	Constraints & Assumptions . . . . .	18
3.3	Methodology . . . . .	19
3.4	Risk Assessment . . . . .	19
3.5	Requirements . . . . .	19
3.6	Timeline . . . . .	20
3.7	Design . . . . .	21
3.7.1	Name & Logo . . . . .	21
3.7.2	Page Design . . . . .	22
3.8	Flowcharts . . . . .	23
3.8.1	User Expertise . . . . .	23

3.8.2 Scoring System . . . . .	25
3.8.3 Storage . . . . .	26
<b>4 Development</b>	<b>29</b>
4.1 Extension template . . . . .	29
4.2 Creating the manifest . . . . .	29
4.3 Creating the popup and options page . . . . .	30
4.4 Creating the content script . . . . .	31
4.5 Created a test website . . . . .	32
4.6 Creating a self-signed certificate . . . . .	34
4.7 Alert Message . . . . .	36
4.8 Syncing results . . . . .	37
4.9 Chrome Web Store . . . . .	38
4.10 Detecting self-signed certificates . . . . .	38
4.11 Expanding user options . . . . .	39
4.11.1 TTL . . . . .	41
4.11.2 Whitelist . . . . .	42
4.12 Creating an issues section on the popup . . . . .	43
4.13 Adding buttons to popup . . . . .	44
4.13.1 Options button . . . . .	45
4.13.2 Save website to whitelist button . . . . .	45
4.13.3 Hide tracking button . . . . .	45
4.13.4 Report website button . . . . .	45
4.14 Adding a drop-down menu to the popup . . . . .	46
4.14.1 Certification Information . . . . .	46
4.14.2 WHOIS Information . . . . .	46
4.14.3 Trustpilot Information . . . . .	47
4.15 Hyperlink Checks . . . . .	47
4.15.1 Domain matches . . . . .	48
4.15.2 Secure hyperlinks . . . . .	48
4.15.3 Invalid Hyperlinks . . . . .	48
4.16 Adding APIs . . . . .	49
4.16.1 Google Safe Browsing . . . . .	49
4.16.2 IP Geolocation API . . . . .	51
4.16.3 urlscan.io . . . . .	51
4.17 Cookies . . . . .	52
4.18 Detecting adverts on pages . . . . .	52
4.19 Learning more about website security . . . . .	53
4.20 Help Button . . . . .	55
4.21 Design . . . . .	56
4.22 Traffic Light Icons . . . . .	57
4.23 About section . . . . .	59

<b>5 Testing</b>	<b>60</b>
5.1 Requirements Testing . . . . .	60
5.1.1 Functional . . . . .	61
5.1.2 Non-Functional . . . . .	62
5.2 User Testing . . . . .	63
5.2.1 Results . . . . .	64
<b>6 Evaluation</b>	<b>72</b>
6.1 Improvements & Future Development . . . . .	72
6.1.1 Premium Tier . . . . .	73
6.2 Problems & Issues . . . . .	74
6.2.1 Improving from user feedback . . . . .	74
<b>7 Conclusion</b>	<b>75</b>
7.1 Teaching . . . . .	75
7.2 Maintenance . . . . .	76
<b>A Installation</b>	<b>80</b>
A.1 Using Chrome Web Store . . . . .	80
A.2 Manual Installation . . . . .	80
<b>B Survey Questions</b>	<b>81</b>
B.1 Introduction . . . . .	81
B.2 Design . . . . .	82
B.3 Functionality . . . . .	83
B.4 Summary . . . . .	84

# List of Figures

2.1 Screenshot of website report from DigiCert . . . . .	12
2.2 Screenshot of website report from synk . . . . .	13
3.1 Project Gantt Chart . . . . .	21
3.2 Draft Icon . . . . .	21
3.3 Final design of icon. By Luca Scomparin 2022 . . . . .	21
3.4 Options page design . . . . .	22
3.5 Popup page design . . . . .	23
3.6 Flowchart showing expertise level . . . . .	24
3.7 Pseudocode showing expertise level . . . . .	24
3.8 Flowchart showing the scoring system . . . . .	25
3.9 Pseudocode showing the scoring system . . . . .	26
3.10Flowcharts showing how storage will work . . . . .	27
3.11Pseudocode showing how storage will work . . . . .	28
4.1 Developing the options page . . . . .	31
4.2 Google Cloud VM Instances . . . . .	33
4.3 Unsecure Test Website . . . . .	34
4.4 Chrome's warning before visiting site . . . . .	35
4.5 Website with self-signed certificate . . . . .	35
4.6 Certificate details . . . . .	36
4.7 Warning message alert . . . . .	37
4.8 Score showing in popup . . . . .	37
4.9 Chrome Web Store Preview . . . . .	38
4.10Options Page with extra options . . . . .	40
4.11Popup Page with buttons . . . . .	44
4.12Popup Page with a drop-down menu . . . . .	46
4.13Popup Page with statistics . . . . .	48
4.14Google's phishing website for testing . . . . .	50
4.15Issue appears when visiting dangerous site . . . . .	50
4.16Issue appears when a suspicious iFrame is found . . . . .	53
4.17Learning more about certificates . . . . .	54
4.18Learning more about WHOIS . . . . .	54
4.19Learning more about Trustpilot . . . . .	55
4.20Information the help button displays . . . . .	55

4.21 Updated design of the popup page . . . . .	57
4.22 Updated design of the options page . . . . .	57
4.23 Icons with traffic light bubbles . . . . .	59
4.24 About section on options page . . . . .	59
5.1 First page of questionnaire . . . . .	64
5.2 Survey results for age . . . . .	65
5.3 Survey results for the cyber security level of user . . . . .	66
5.4 Survey results for how user-friendly the popup page is . . . . .	66
5.5 Survey results for how easy it is to understand the popup page . . . . .	67
5.6 Survey results for how user-friendly the options page is . . . . .	67
5.7 Survey results for how easy it is to understand the options page . . . . .	68
5.8 Survey results for how useful they found the extension . . . . .	69
5.9 Survey results for how useful they found the security warning . . . . .	69
5.10 Survey results for how useful they found the extension . . . . .	70
5.11 Survey results for recommending the extension . . . . .	70
B.1 Popup image for survey . . . . .	82
B.2 Options image for survey . . . . .	83

## List of Tables

2.1 Security features of a website . . . . .	17
3.1 Functional Requirements . . . . .	20
3.2 Non-Functional Requirements . . . . .	20
5.1 Functional Testing . . . . .	61
5.2 Non-Functional Testing . . . . .	62

# Chapter 1

## Introduction

Title: Cyber Security Education & Training - Chrome extension

Project Code: HA-04

Programme: BSc Computer Science

Module Code: COC251

Principal Supervisor: Asma Adnane (a.adnane@lboro.ac.uk)

### 1.1 Project Definition

Cyber security or information security training is now mandatory in most businesses. There are different types of training available at different costs, however, a small effort is made to check the efficiency of those training and whether or not they are really educating the learners. In this project, you will review the different types of cybersecurity training and evaluate their efficiency, and analyse how the training and education sessions could be improved.

A handwritten signature in black ink, appearing to be 'RA' or similar, located below the project definition section.

### 1.2 Project Brief

Currently, there are plenty of security threats on lots of websites that too many people are becoming vulnerable to. Human error is causing companies and the general public to hand over private information unknowingly. Cybersecurity Ventures expects the cost of cyber crime to reach 10.5 trillion USD by 2025. [Morgan, 2020] It is important the public are more informed about what they are doing when they are online.

It is for these reasons that I have chosen to create a Chrome extension that shall analyse websites. This extension will run in the Chrome browser in the background of



daily tasks and pop up when it notices vulnerabilities in security. More specifically it informs them of any issues and helps train the public into becoming more secure users of the internet.

I have chosen this project as I am interested in cyber security when browsing the internet and in creating Chrome extensions.

I will build a GUI which will popup when the user wants to view more information about a website. The algorithm created will monitor various parts of the website like the SSL certificate and the hyperlinks it contains.

## **Chapter 2**

# **Literature Review**

In this chapter, I explore the current research that has been done in the area of cyber security awareness among the general public. I examine the most popular security threats people are vulnerable to on the internet and seek out what exists to protect against them. I primarily focus on internet browsing; how oblivious most people are to the threats that can come from visiting a website and the vulnerabilities that lie in the websites that are used. I explore the methods of educating users to give them a better understanding of the risks they take when they interact with a website allowing them to make a better decision on how they continue to use the website.

### **2.1 Requirements in cyber security**

When computers were first being built, they provided many new possibilities for the world, however, this also brought along a different type of crime known as cyber crime. Cyber crime meant hackers would find vulnerabilities in systems and exploit them for their benefit in a variety of ways. Thus began the ongoing battle to constantly create more secure systems to reduce the chance of cyber attacks. Every day, systems are becoming more secure but new ways of breaking these systems are also being discovered and used. Over the years companies and governments have had to produce requirements, standards and regulations all to ensure these systems are always as secure as possible. Bayuk and Horowitz show the necessary defence-in-depth, bulls-eye target that needs to be in place in a digital system to provide security to the asset of interest. [Bayuk and Horowitz, 2011] Nowadays the security of a digital system is a key consideration during the design and development stages. It takes time and needs to be done accurately, and even once that is done, there still needs to be constant maintenance to keep it secure whilst in use. Even once the system has finished its use, the proper shutdown of the system has to be done carefully as it is common for legacy systems with vulnerabilities to be the cause of cyber attacks.

### **2.1.1 The popularity of internet browsing**

Chromebooks are devices running Chrome OS, the main use of these devices is to simply run the Chrome browser for the user. Whilst Chromebooks are not everyone's favourite device to browse the internet, they are strongly recommended for use in the education sector. [O'Donnell and Perry, 2013]

The production of these devices from a company as big as Google shows that they believe internet browsing is and will continue to be used heavily by a large number of people. The fact that they are being used by a lot of young people in education currently could suggest it is believed the future generation will continue relying on websites for their needs as they grow up.

### **2.1.2 Uses of internet browsing**

Internet browsing has endless possibilities nowadays, some of the most important uses include online banking, online shopping and being a source of information. Money is of course valuable to people, people choose to hold their money physically or digitally. The rising amount of people holding their money on online banks shows their trust in their banking company and the infrastructure that the internet is built on. Something similar can be said for online shopping with how people trust digital transactions. Many do not understand how it works yet companies and governments rely heavily upon it and people are expected to use it.[Wang et al., 2009]

One of the most basic uses of internet browsing is using it as a source of information. Usually, children don't question the reliability of the source of the information they find. Most adults will do this but it can be confusing which sources are reliable and which aren't. For example, an article from the BBC is usually trusted as they see news from the same company on their TVs. However, content from somewhere like Wikipedia, only a hand full of people understands that this content can be edited by anyone and may not be fully reliable.

## **2.2 Cyber security risks of internet browsing**

A website to many people is usually just a source of information, like a web page for displaying current local weather. A website is known to be secure if it has certain properties, like HTTPS, this signifies the traffic from the website's server to the client's device is encrypted. This is just one of the nearly infinite ways of determining if a website is secure. Most users are prone to making simple mistakes through human nature which can have detrimental effects on their own privacy or an organisation's security. [Safa et al., 2016] Simple mistakes may include leaving passwords lying around or downloading unknown software. More directly fatal mistakes would include falling for a social engineering method or phishing. It's important that this is changed and instead of people being one of the most popular factors in causing a cyber attack, they become part of the solution where they are taught how to browse safely. [Zimmermann and Renaud, 2019]

### **2.2.1 HTTPS**

To know if a website uses HTTPS; it should say in the URL, the browser might display a symbol or the browser might have the ability to present the user with the website's TLS/SSL certificate. HTTPS should not be mistaken for full security on a website as a lot of people may assume. Encryption exists across the connection, but that's not the only thing that is important. While encryption prevents anyone listening on the network to see the content of the packets sent, it does not authenticate the server providing the data. An HTTP website will not have any TLS certificates but sometimes it can be more secure than an HTTPS website if it was created using a self-signed certificate. A self-signed certificate is a form of encryption between the website and user but it's not verified by other parties so still can be a risk as it poses as a secure site but may want to do this for malicious reasons. [GlobalSign, 1996] A quick way of checking a portion of a website's reliability is to use the Wayback Machine, it allows us to view the archived website, therefore we can see how much it has changed recently. If much has changed, we can conclude it might be a risk to use, whereas if the website has been around for a long time, it might be genuine. [Gonella and Nericcio, 1996]

### **2.2.2 Network analysis**

Wireshark is a popular tool among hackers to sniff networks, this is a term for analysing the packets that go through a LAN, if the website the user is using isn't encrypted, the attacker can view all the data being sent through and can use it to perform an attack. [Combs, 1998]

### **2.2.3 SQL injection and XSS attacks**

SQL injections and cross-site scripting are similar issues. An SQL injection is inserted into a field of a website and manipulates the database the website uses in some way. An XSS attack is where the attacker injects a website with a malicious script into a trusted website to manipulate it in some way. Both methods were popular vulnerabilities in websites once, nowadays they're commonly protected against securer websites. [Cheah, 2019]

### **2.2.4 Phishing**

Social engineering can be the most fatal/ dangerous type of attack. This is because the vulnerability relies more upon the person rather than the system. A social engineering attack can be done in many ways, for example, in person or digitally. It relies upon finding out private or secure information which can be used to gain access to a system. A decoy website that is designed to look exactly like the real website can have a URL that looks exactly the same but could use special characters which most humans wouldn't notice. The user would enter private information through phishing and this can be used for a variety of attacks now it's gained access to the system. [Hinson, 2008]

## 2.3 Safe internet browsing

Safe internet browsing is where a user visits and uses a website without any personal information being stolen or manipulated. There are many ways to determine whether a website is safe. I don't believe a website can ever be 100% secure but we can perform a number of checks to find out if the website is vulnerable to certain attacks. For example, checking if the connection is using HTTPS is a simple way of ensuring the traffic isn't plainly read on the network.

### 2.3.1 Online website analysis

Currently, there are many online tools to check how secure a website is. The websites below offer pen testing tools which provide a security checkup on the URL provided by a user.

#### DigiCert

The company DigiCert offers a website where you can enter a website's URL, the company will analyse the website and provide the user with details about the website's security. To name a few details they provide, the company would provide the user with detailed certificate information, it checks for common vulnerabilities like Heartbleed, Poodle (TLS), Poodle (SSLv3), FREAK, BEAST, CRIME, DROWN, it checks the certificate chain and checks the server configuration. [Digicert, 2020]

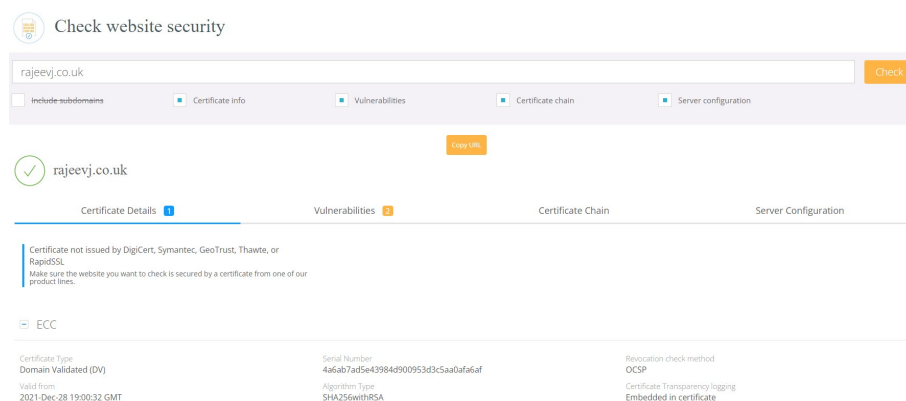


Figure 2.1: Screenshot of website report from DigiCert

#### ImmuniWeb

This website is full of detailed analytics and can appear overwhelming to most people, including experts. It provides an overall final grade, this grade is a combination of a number of factors, the website explains its scoring method. [ImmuniWeb, 2019a] There are sections for discovered subdomains, a web server security test, a web software

security test, a GDPR compliance test, a PCI DSS compliance test, HTTP headers security test, content security policy test, cookies privacy and security analysis and finally, external content privacy and security analysis. [ImmuniWeb, 2019b]

## Sucuri

Sucuri is a company which claims to clean and protect websites. The analysis on this website is clearer but not as informative. It provides the IP address, the CDN, CMS, the security risk, malware found, blacklist status, firewall information, website monitoring information and hardening improvements. The security risk is measured on a simple scale which can be understood quite well by most people. [Sucuri, 2010]

## Pentest Tools

Pentest Tools has the ability to perform light and full scan (at a cost). The report for a light scan provided a simple summary but was not very helpful. It produced an overall risk level of "high", "medium" or "low". It performed 13 tests and categorised them into the 3 levels depending on how they passed. Most of the tests were related to security headers, one involved simply finding a robots.txt file which is used to avoid overloading a site with requests. The full scan had a more comprehensive testing area. [Tools, 2013]

## Synk

Snyk is a vulnerability scanner which performs a passive web security scan in order to detect issues like outdated server software and insecure HTTP headers. It provides a security score as a letter and explains the scoring system. A link to the full report opens a page to a report from WebPageTest by Catchpoint. The synk website displays only the JS libraries with vulnerabilities and info on security headers making it limited. [synk, 2015]

**Security Analysis for:**  
**https://rajeevj.co.uk**

Snyk's security scan found the following vulnerabilities affecting your website. Ready to fix your vulnerabilities? Automatically find, fix, and monitor vulnerabilities for free with Snyk.

[Fix for free](#)

Full report [See on WebPageTest](#) Scan time 12/17/2021 10:25:36 AM

Webpage Security Score

**B**

A+ is the best score you can get. Learn more about this score.

Fix JavaScript vulnerabilities in your project with Snyk (or try the free & open source CLI) [Test and protect my website](#)

**JavaScript Libraries with vulnerabilities**

✗ The following list of JavaScript libraries were found to contain known and public security vulnerabilities.

We highly encourage you to upgrade to fixed versions as soon as possible.

[Monitor my web application's project dependencies](#)

**Security headers**

HTTP security headers enable better browser security policies.

Successfully detected the following security headers:

- ✓ strict-transport-security

Figure 2.2: Screenshot of website report from synk

### 2.3.2 Applications

All the websites I had tested seemed inconvenient and a little complicated to use. A number of Chrome extensions have been developed to improve the security of online users. These are more accessible to users and therefore are easier to use.

#### HTTPS Everywhere

This Chrome extension attempts to convert every site with an HTTP connection into an HTTPS connection. There still lies issues surrounding HTTPS as mentioned earlier but nevertheless, this is a simple and effective way of providing a more secure browsing experience to the public. The extension doesn't do anything else apart from making a website into HTTPS so this is limited. It doesn't provide any vulnerabilities. [EFF, 2010]

#### WOT Website Security & Browsing Protection

The WOT Website Security & Browsing Protection Chrome extension attempts to be an advanced browsing, security and privacy shield. It provides a scorecard of the current website open, a browsing detection summary and reviews of the website. More premium features can be unlocked at a price. The free summary isn't detailed and doesn't detect malicious links as it claims to. [Trust, 2007]

#### IP Whois & Flags Chrome & Websites Rating

The IP Whois & Flags Chrome & Websites Rating Chrome extension is useful. The extension opens an info page displaying IP address, popularity, location and host. The other tabs include Whois, reputation, SafeWeb, Alexa, site reviews, page reviews and incognito. The Whois tab is very informative but confusing to most people. The reputation tab used WOT for analysis. The SafeWeb uses Norton for website analysis. The Alexa tab opens Amazon's report on how the site is visited. Whilst some of these tabs are quite useful, the extension is not very user friendly and can cause confusion to those who prefer just a simple summary. [MYIP.MS, 2012]

### 2.3.3 Other tools

Throughout my research, I discovered other tools that people have used to help detect vulnerabilities in websites.

#### Nikto

Nikto is an open-source web server scanner which performs comprehensive tests. It checks for over 6700 potentially dangerous files/programs, checks outdated versions of over 1250 servers, and version specific problems on over 270 servers. While the tests it does is detailed and thorough, it is for users with a high level of technical knowledge. [sullo, 2001]

## **NCSC**

The official National Cyber Security Centre in the UK provides information to the public in the UK about understanding vulnerabilities. They offer weekly threat reports to help educate people. The language they use is quite technical for most of the public to understand. They refer to the website for the NVD, the National Vulnerability Database in the US, this is a database for all known vulnerabilities that have been discovered by the U.S. government. [Centre, 2016]

## **OWASP**

The Open Web Application Security Project produced a list of the top 10 web application vulnerabilities in the last few years. They are updated every couple of years, the last 2 updates were in 2017 and 2021. The list is categorized for web application security, some are new while some are old and have been reduced down the list. This list is useful to developers while building their own websites. [Project, 2001]

## **More Website Penetration Tools**

This blog discusses tools that are used to penetrate websites. For example, Nmap is mentioned as a program to fingerprint a server OS, bypass a firewall and more. The program called the Harvester provides a similar summary as Whois. These are just a few of the programs mentioned. Whilst this is a good way of detecting the security of a website, it is not practical for most people to use. [Krishna, 2017]

## **2.4 Website rating and evaluation**

Companies rate and score websites depending upon a variety of factors, each company does this differently. This rating provided helps the user determine whether they should use it or not. This also depends on the user's purpose for the website, for example, if they were using it to learn a cooking recipe, using an unsafe website isn't as dangerous as if they were using it for online banking.

### **2.4.1 How is a website evaluated?**

Companies perform a wide range of checks on websites to determine their rating as mentioned earlier. Each company usually has their own custom algorithm to determine this. Other methods also exist for rating a website but this is less based on security and more surrounding user experience. For example, Trust Pilot, is a company which allows users to provide a rating with personal feedback on their thoughts on the website. [Muhlmann, 2007] The rating is an average of all the reviews and represents to a prospective user if the website should be used, which isn't just based upon security, it can also be based upon ethical views of the website.



### **2.4.2 Factors for rating a secure website**

Below in table 2.1, I have summarised a number of factors that can influence the decision of using a website. These factors are technical and can be used to calculate how secure the website is to use.

## **2.5 Conclusion**

It is clear that the current solutions in place for helping users aren't making a large enough impact. Too many people are unaware of the risks of using certain websites. They are unclear about what makes a website a threat and how to avoid putting themselves at risk of cyber attacks. There needs to be a better way of educating users.

### **2.5.1 Where we need to improve**

As discovered earlier, there exist lots of websites which analyse other websites, there also exists a few Chrome extensions which do the same job. The information they provide is cluttered and overwhelming to the average person. They are more aimed at experts who understand the report it provides. The information is useful but it's important to provide the user with this information in a way they can understand and clearly decide if they want to use it. It takes a more accessible program which can make more advanced decisions about whether a website should be used and be able to provide the user with simple reasoning as to what makes a website dangerous. This helps the user gain an understanding for themselves which is the ultimate aim.

Security Feature	Description	Effect
Server Configuration	Contains details like the protocols enabled, the host name, server type, IP address and open ports.	IP address can help reveal the location of the owner and open ports can help an attacker gain access to servers.
Certificate Details	Contains details like certificate type, serial number, algorithm type, key size and certificate status.	These details tell us whether the website has an HTTPS connection to the user. Can tell us if the traffic is encrypted.
Certificate Chain	There are two types of certificate authorities (CAs): root CAs and intermediate CAs.	For an SSL certificate to be trusted, that certificate must have been issued by a CA that's included in the trusted store of the device that's connecting.
Adware	These adverts automatically display advertising material such as banners or pop-ups in large areas of the screen.	They can easily be clicked and cause a user to download something harmful.
Old Library Versions	This is where a website is using outdated libraries and has known vulnerabilities.	These vulnerabilities can be exploited if the library hasn't been updated.
Payment Methods	Are they using a secure method like PayPal or are they using their own custom made one? How secure is their payment method?	The payment details of a user can easily be stolen if a payment method isn't properly secure.
Common vulnerabilities	More details of these are on the NCSC website. Popular ones like Heartbleed and Poodle (TLS) can affect website security. Each vulnerability is unique so it's important to have different ways of checking for each one; they also need to be updated and monitored regularly as they adapt so frequently.	These popular vulnerabilities are what attackers are looking out for and if the website isn't protected against them, they're high at risk. OWASP is an effective place to find the most common vulnerabilities.
Hyperlinks	Most websites have hyperlinks which lead the user to another page on the internet. These other pages may not be using HTTPS or not be in the same domain as the original page.	These hyperlinks may lead to dangerous sites or maybe fake/invalid. The more the number of links like this on the page, the more untrustworthy it is.
Cookies	Cookies track the user sessions as they browse the internet. It records where they went and what actions they did.	These cookies can exploit the user and use it for targeted advertising or sell this data to other companies without the user's knowledge.

Table 2.1: Security features of a website

## **Chapter 3**

# **Plan**

### **3.1 Stakeholders**

I will be designing my extension for people who use the Chrome browser frequently for exploring the internet. The extension will be largely aimed at users who don't have much cyber security awareness however more technical users can also use it to help them decide whether a website is safe to use. The audience can be generalised as anyone who uses the internet without realising the risks of the consequences. To be more specific, I would focus on young adults who are being introduced to the internet, as more of the physical world moves digitally, like mobile restaurant ordering systems, it's imperative that they are being taught good habits to get into. Another target market is the older generation who have been given devices to browse the internet in an age where everything seems to be going online but have never been taught any cyber security awareness so this extension will help them avoid scams.

### **3.2 Constraints & Assumptions**

I will assume that users who are 'beginners' have a basic understanding of the internet and are unsure how to protect their private data whilst browsing. They are vulnerable as they do not understand how the internet or browsers work. I will not only aim to provide a rating of how secure each page is but I also aim to teach beginners the basics of the internet and what to look out for.

I will assume users who refer to themselves as 'experts' have a good understanding of technical terminology. For them, this extension acts as a second opinion on whether to trust a website. These users may know what factors to look out for in a website but this extension summarises this information in one place making it more accessible and allowing them to make quicker decisions.

### 3.3 Methodology

I decided I would follow the waterfall software development life cycle, to begin with as this is a simple and linear structure for planning and development. The program I am making will not have changing requirements and has a long time frame making the waterfall methodology a good fit.

As I progress into the development and near the end, I could find myself moving to agile development as requirements adjust depending on progress. These methodologies are very similar and combine together well. [Crespo-Santiago and Cosme, 2011] As I receive feedback from users and test features, I could go back and redesign the interface or how the algorithms are designed.

The waterfall methodology is simply structured and suitable for reaching a deadline. However, it does exclude the users from the development process. Instead, I intend to get continuous feedback stakeholders users throughout the development to help my extension be user friendly, well designed and functioning correctly.

### 3.4 Risk Assessment

As my extension will be used by beginners, it could be mistaken to be a tool provided by Google Chrome which gives the impression it is highly accurate and reliable. This extension may have bugs by the time it is in production. If a website is safe and my extension mistakenly says it is unsafe, this is problematic but not very dangerous. However, if the opposite happens where my extension says a website is safe but is actually dangerous, then a person who has full trust in my extension could fall prey to a scam or cyber attack.

It is important I make it clear in a disclaimer that this extension was not made by an official authority and has not been verified so users do not put all their trust into this extension.

### 3.5 Requirements

The requirements are divided into two sections. A section is for functional requirements which gives the extension its purpose. These requirements include security features which give the user an informed summary of the website.

Functional ID	Functional Requirement
ID-F1	Extension can detect if the website is using an HTTP or HTTPS connection
ID-F2	Extension can detect if the website is signed using a self-signed certificate
ID-F3	Extension can detect if the hyperlinks on the website are in the same domain they are already on
ID-F4	Extension can detect if the hyperlinks on the website are using HTTPS
ID-F5	Extension can detect if the hyperlinks on the website are valid URLs
ID-F6	Extension can use WHOIS/ RDAP information to determine how safe a website is
ID-F7	Extension can analyse cookies in websites
ID-F8	Extension can detect adverts and popups

Table 3.1: Functional Requirements

The second section is for non-functional requirements. These requirements are extra features that are not compulsory for the extension to work but gives extra functionality to the user giving them a better user experience.

Non-Functional ID	Non-Functional Requirement
ID-NF1	Extension should have a simple user interface, easy to use
ID-NF2	Extension should have option to decide on level of expertise
ID-NF3	Extension should have option to disable popups
ID-NF4	Extension should have option to change how long before the popups show up again for all websites
ID-NF5	Extension should have option to whitelist trusted websites
ID-NF6	Extension icon should change based on website rating using a traffic light colour system
ID-NF7	Extension should have a welcome page on first installation to be used for a tutorial
ID-NF8	Extension should clear the user data annually for extra security

Table 3.2: Non-Functional Requirements

### 3.6 Timeline

There are multiple stages to this project; the introduction, development, testing, evaluation and the conclusion. I intend to spend a lot of time on the planning and development to ensure the later stages become easier to manage. Below in figure 3.1 I have planned my time over the next year.

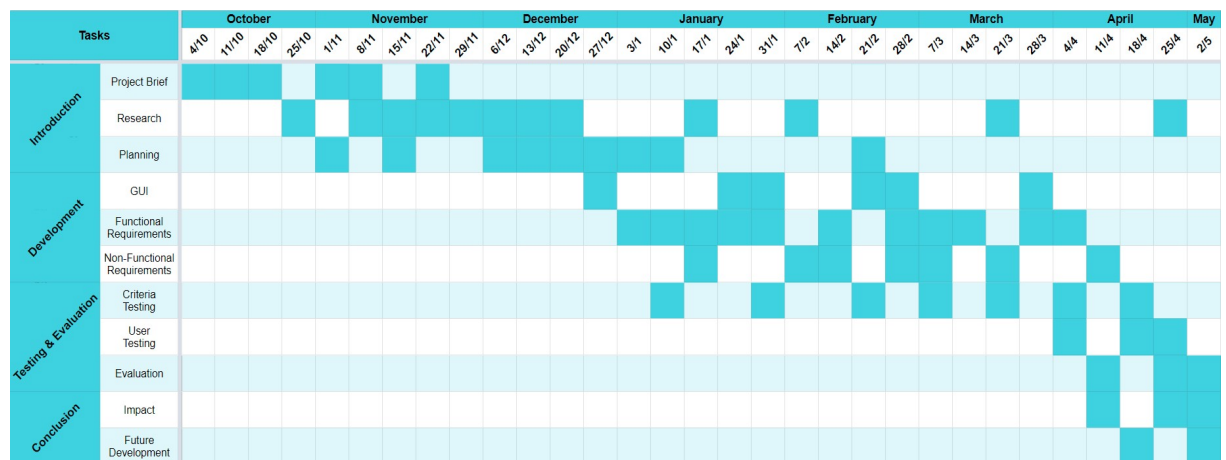


Figure 3.1: Project Gantt Chart

## 3.7 Design

### 3.7.1 Name & Logo

I chose the name "Security Surf" for my Chrome extension to represent surfing and browsing the internet securely.

To design my logo for the Chrome extension I researched into the area of surfing, this showed images of surfboards and waves. The idea of security brought up images of locks. I decided to combine these ideas with a lock with waves. Below in figure 3.2 is an outline I designed for the icon to look like.



Figure 3.2: Draft Icon

I then sent this outline to a designer to produce a more professional image with a description of what the icon should represent. I was sent back an amazing design of a lock with waves with portrayed my Chrome extension perfectly shown in figure 3.3.



Figure 3.3: Final design of icon. By Luca Scomparin 2022

### 3.7.2 Page Design

Figma was used to design the extension pages, it is a popular design tool for developers, easily accessible on a website with quick controls to design simple pages.

For the options page, this will include a lot of the non-functional options the user will be able to control. To begin I chose the four main options, the rating level, visible popups, rechecking pages and whitelisting sites. The Figma design is shown in figure 3.4.



Options

## Security Surf

How would you rate yourself?

Do you want to turn on security popups?

How long would you like before we recheck the page's security?

Whitelist

Save

About this extension

Figure 3.4: Options page design

For the popup page, this will include most of the information as well as some non-functional options the user will be able to control. This is shown in 3.5. In large text is the rating as this is the main purpose of the extension. Then next to it will be some statistics about the website which helped contribute towards the score. Below are buttons for the user to act on the website. Beneath is the issues box which will help the user learn about why the website is unsafe in language they will understand.

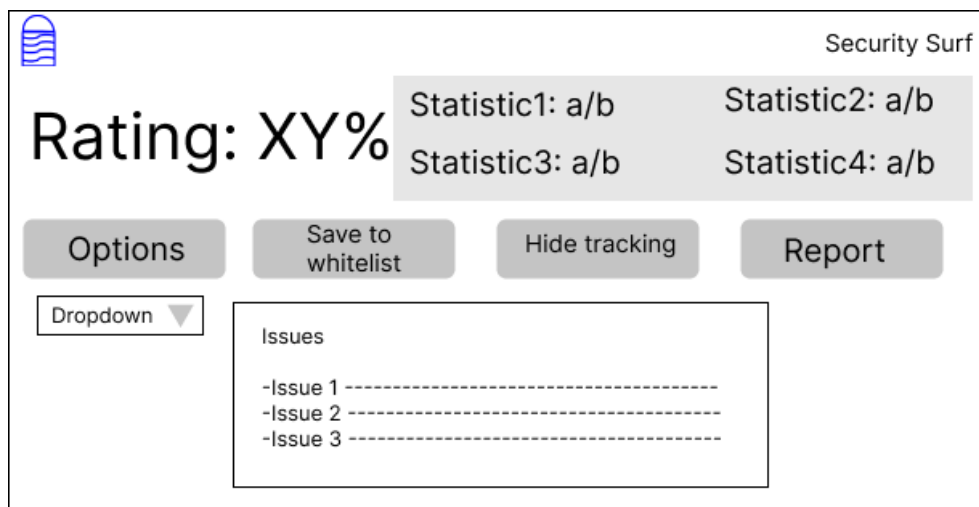


Figure 3.5: Popup page design

## 3.8 Flowcharts

### 3.8.1 User Expertise

Below in figure 3.6 is a flowchart to show how my algorithm will work as it collects data from websites. It will create an issue array from this depending on the user's understanding, 'beginner' or 'expert'. Figure 3.7 shows the flowchart made into Pseudocode.



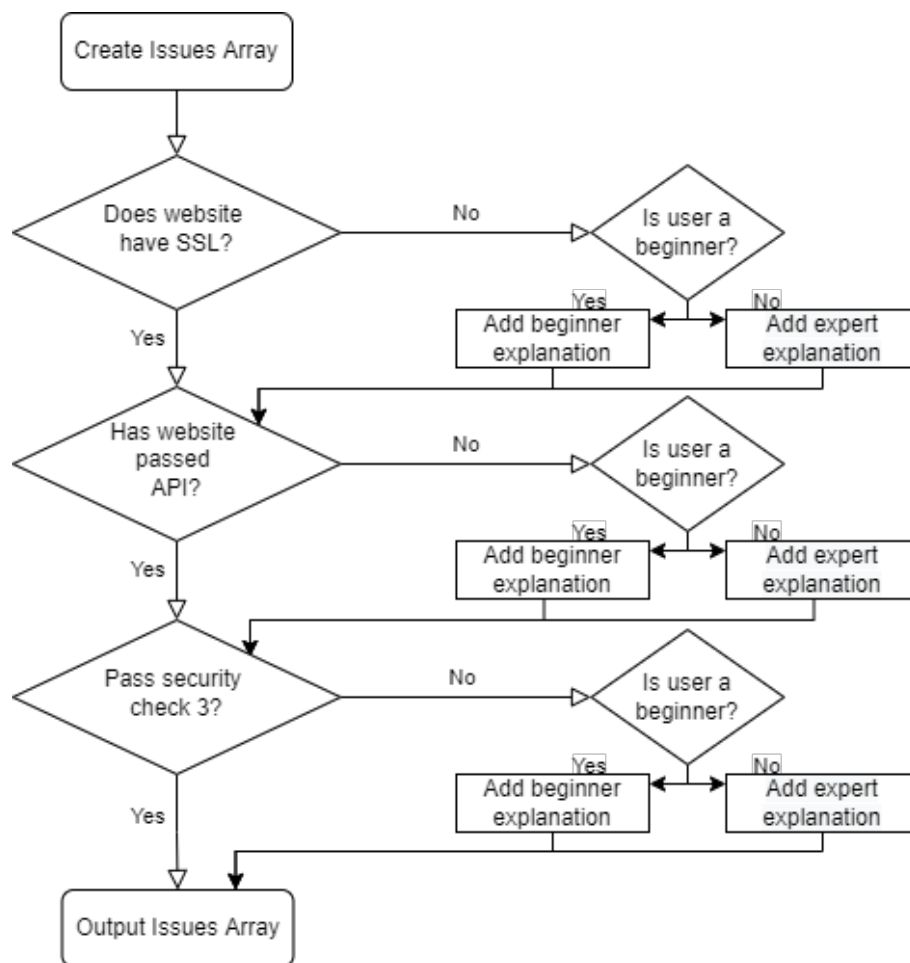


Figure 3.6: Flowchart showing expertise level

```

1  issueArray=[]
2  if websiteSSL==false then
3    if userLevel=='beginner' then issueArray.add(beginnerExplanationSSL)
4    else issueArray.add(expertExplanationSSL)
5  if websitePassAPI==false then
6    if userLevel=='beginner' then issueArray.add(beginnerExplanationAPI)
7    else issueArray.add(expertExplanationAPI)
8  if securityCheck3==false then
9    if userLevel=='beginner' then issueArray.add(beginnerExplanation3)
10   else issueArray.add(expertExplanation3)
11  output (issueArray)

```

Figure 3.7: Pseudocode showing expertise level

### 3.8.2 Scoring System

Below in figure 3.8 is a flowchart to show how my algorithm will work as it scores websites. This may change depending on how many checks it is feasible to add. Too many can slow down the browser too much. Figure 3.9 shows the flowchart made into Pseudocode.

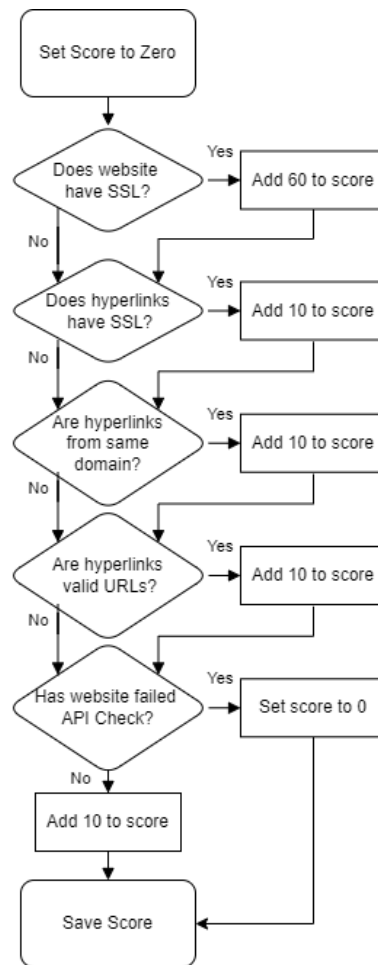


Figure 3.8: Flowchart showing the scoring system

```
1      score=0
2      if websiteSSL()==true
3          score=score+60
4      if hyperlinksSSL()==true
5          score=score+10
6      if hyperlinksDomain()==true
7          score=score+10
8      if hyperlinksValid()==true
9          score=score+10
10     if websiteAPIcheck()==false
11         score=score+10
12     else
13         score=0
14     return (score)
```

---

Figure 3.9: Pseudocode showing the scoring system

### 3.8.3 Storage

Below in figure 3.10 are flowcharts to show how my algorithm will work as it saves user data to Chrome's storage. The parallelograms refer to either getting or setting data in Chrome's storage. Figure 3.11 shows the flowchart made into Pseudocode.

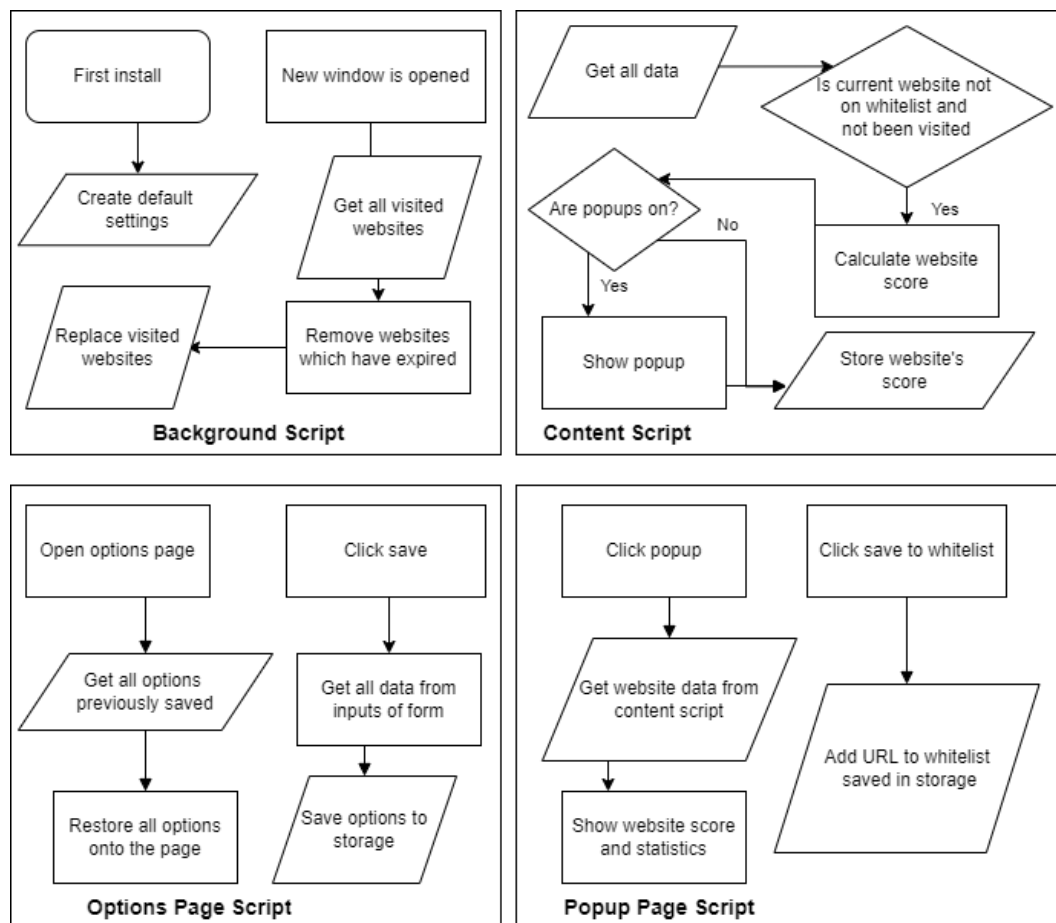


Figure 3.10: Flowcharts showing how storage will work

```
1      //Background Script
2      if firstInstall then
3          options=[default_settings]
4          Chrome.set (options)
5      if newWindow then
6          Chrome.get (visitedWebsites)
7          for website in visitedWebsites
8              if today-website.timeVisited>timeSet then
9                  visitedWebsites.remove(website)
10             Chrome.set (visitedWebsites)
11
12      //Content Script
13      Chrome.get (data)
14      if (!data.whitelist.includes(currentWebsite))
15      && (!data.visitedWebsites.includes(currentWebsite)) then
16          score= calculateScore(currentWebsite)
17          if popups=true then
18              showPopup()
19          visitedWebsites.add(currentWebsite,score)
20          Chrome.set (visitedWebsites)
21
22      //Options Page Script
23      Chrome.get (options)
24      restoreValues(options)
25      if saveButton then
26          options=getAllInputs()
27          Chrome.set (options)
28
29      //Popup Page Script
30      Chrome.get (visitedWebsites)
31      website=visitedWebsites[currentWebsite]
32      showScoreAndStatistics(website)
33      if saveToWhitelistButton then
34          whitelist.add(website)
35          options.add(whitelist)
36          Chrome.set (options)
```

---

Figure 3.11: Pseudocode showing how storage will work

# Chapter 4

## Development

### 4.1 Extension template

To begin my development for the Chrome extension I have created a private repository on GitHub. Git is useful for tracking my changes and undoing mistakes I make, this ultimately saves time and ensures my project stays organised. I am using the company GitHub as I already have other projects on there so I am familiar with the commands and the benefits of being cloud-based.

I am using Visual Studio Code as my choice of IDE for this project. It has a number of benefits like syntax highlighting, bracket-matching, auto-indentation a number of useful extensions. For example Beautify for aesthetics, Kite for auto-complete and GitLens for direct access to GitHub.

I began my Chrome extension by reusing the extension already made by Google on the 'Getting started' page for Chrome extensions. This is also where I learnt of the new changes from manifest V2 to V3 as I have previous experience creating Chrome extensions but using Manifest V2. My next stages were to remove the boilerplate code to create a template of what my extension could look like. I added the libraries for Bootstrap in an attempt to give the pages a more user-friendly design. I added the libraries of jQuery as it has fast selection and easy DOM manipulation.

### 4.2 Creating the manifest

This is the main entry when the Chrome browser runs the extension. The main files are the popup.html, the options.html and background.js. The popup.html file is the default file which opens when the extension icon is clicked, this will show information about the website. The options.html file is for the user's options to control what is being presented to the user. The background.js file is the background script which will have listeners initialise the storage on the first installation and clean up stored data periodically.

Listing 4.1: Manifest V3

```
1 {  
2   "name": "Security Surf",
```

```

3   "description": "An extension used to provide the
4   user with a security analysis of a website",
5   "version": "1.0",
6   "manifest_version": 3,
7   "background": {
8     "service_worker": "background.js"
9   },
10  "permissions": ["storage", "activeTab"],
11  "action": {
12    "default_popup": "popup.html",
13    "default_icon": {
14      "16": "/assets/SecuritySurfLogo16.png",
15      "32": "/assets/SecuritySurfLogo32.png",
16      "48": "/assets/SecuritySurfLogo48.png",
17      "128": "/assets/SecuritySurfLogo128.png"
18    }
19  },
20  "icons": {
21    "16": "/assets/SecuritySurfLogo16.png",
22    "32": "/assets/SecuritySurfLogo32.png",
23    "48": "/assets/SecuritySurfLogo48.png",
24    "128": "/assets/SecuritySurfLogo128.png"
25  },
26  "options_page": "options.html"
27 }

```

### 4.3 Creating the popup and options page

The popup page has been created, to start with, it displays a value onto the window using JavaScript which will act as the rating of the current page once clicked. The options page has a little more functionality. There is an HTML form where it collects the user's expertise level and pop-up preference. The expertise level will eventually be used to display more relevant information depending on their knowledge. The two preferences are "beginner" and "expert". This means that requirement ID-NF2 has been met. The security pop-up preference will allow the user to decide whether the pop-up will inform the user as soon as it visits a dangerous website or if the user should only be informed when the extension is clicked on. A popup constantly showing on websites may get annoying so giving this option to the user will help. This means that requirement ID-NF3 has been met. There is a button to save this information. The extension saves this data to the user's storage for their Google account so it can be restored on another browser. To save the user's data I used the following commands which use Chrome's storage API.

Listing 4.2: Saving User's Options

```

1      let extensionOptions={

```

```

2         "expertiseChosen":expertiseChosen,
3         "popupOption":popupOption}
4     chrome.storage.sync.set({"extensionOptions":extensionOptions});

```

There is another button underneath that displays 'About this extension' which displays information about me like my name and contact details. The basic layout of the options page can be seen below in figure 4.1.

**Security Surf**

**Options:**

How would you rate yourself?

Choose:

Do you want to turn on security popups?

Choose:

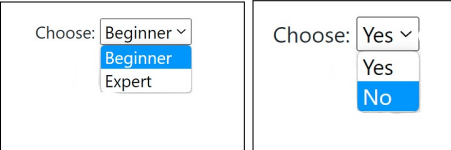


Figure 4.1: Developing the options page

## 4.4 Creating the content script

The content script, `content.js`, had the task of scoring each page the user visited. It needed access to the storage so the score can be remembered by the extension in future. This meant the extension needed access to use Chrome's storage API. [Developers, 2022] It also needed access to the web page the user was visiting. [Developers, 2021] The 'activeTab' permission granted the extension temporary access to the currently active tab which will allow it to get the data it needs to formulate a score.

In the content script, I created a basic check to see if the website used HTTPS. This checked the protocol of the website and compared it to 'HTTPS' and 'HTTP'. This means that requirement ID-F1 has been met. The next part to this was to inform the user of any large vulnerabilities immediately, this displayed a string of text in an alert on the web page.

Listing 4.3: Checking HTTPS

```

1     let score=0;
2     //Website variables
3     let url=window.location.href
4     let urlProtocol=window.location.protocol;
5

```



```
6      //Check if SSL certificate
7      if (urlProtocol=="https:"){
8          score+=100;
9          let httpsUsed=true;
10     }else if (urlProtocol=="http:"){
11         score+=0;
12         let httpsUsed=false;
13     }
14     //Inform user of any vulnerabilities
15     let issues="";
16     if (score<30){
17         issues = issues.concat("\n This website looks unsafe! Be careful!");
18     }
19     if (!httpsUsed){
20         issues = issues.concat("\n      -This website uses no encryption!");
21     }
22     //Output issues to user
23     if (issues.length){
24         alert(issues);
25     }
```

---

## 4.5 Created a test website

To check my content script was working, I needed a website which had HTTPS and one that did not. While I could have found one online, I decided to create my own website to test this. This was so that as I add more features to my extension, I had a test website that I had complete control over.

I chose Google Cloud to host this website for a number of reasons. I have experience using Google Cloud in the past, they offer new customers \$300 in free credits to use and web hosting on it has good availability, a global infrastructure, good security, good documentation and has an easy setup. This made it ideal. Figure 4.2 shows the interface for VM Instances on Google Cloud's Compute Engine.

To create the website I had to create a Google Cloud project, 'Security Surf'. Google Cloud also offers a free tier, this means I can host my website on Google cloud for free as long as I agree to their terms like using an e2-micro VM instance.

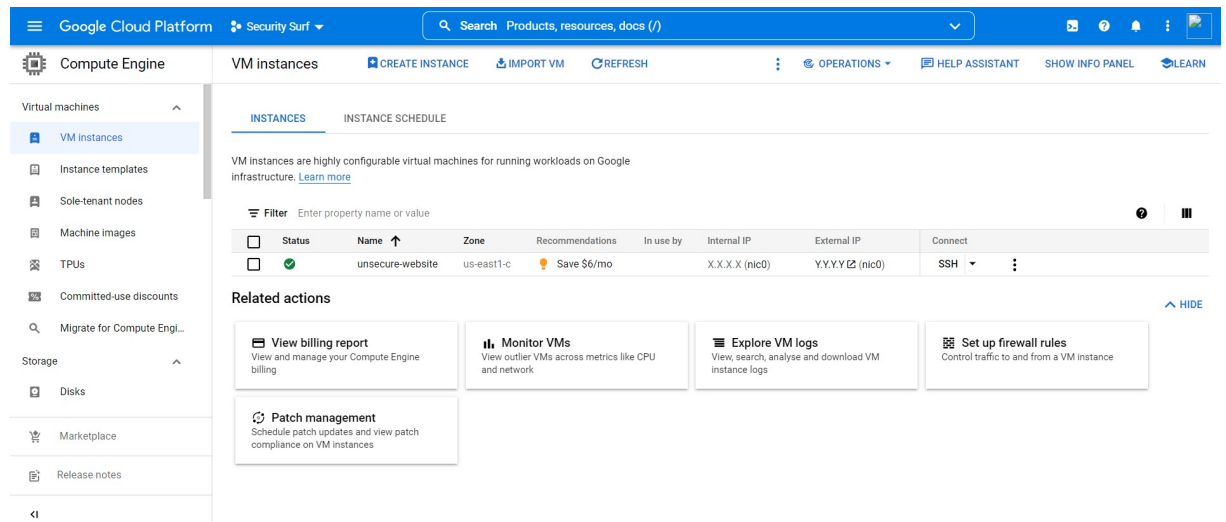


Figure 4.2: Google Cloud VM Instances

Once I created a new instance I connected to it via SSH. This provided a Linux shell which I could issue commands. To host the website I installed NGINX on the instance. I setup a simple website.

Listing 4.4: index.html

```

1      <!DOCTYPE html>
2      <html lang="en">
3      <head>
4          <meta charset="UTF-8">
5          <meta name="viewport" content="width=device-width, initial-scale=1.0">
6          <title>Rajeev FYP Site</title>
7          <link rel="stylesheet" href="style.css">
8          <script src="index.js"></script>
9      </head>
10     <body>
11         <h1>Hi, I am a test website.</h1>
12         <h2 id="demoSentence">Nice to meet you.</h2>
13     </body>
14 </html>

```

Listing 4.5: index.js

```

1      function myFunction() {
2          document.getElementById("demoSentence").innerHTML="How are you doing today?";
3      }
4      myFunction();

```

Listing 4.6: style.css

```

1      body {
2          background-color: lightblue;
3      }

```

Once this was saved the website could be viewed by going to the IP address where the virtual machine was being hosted. I decided to attach a domain to this, I already owned by personal domain "rajeevj.co.uk". I added an A record into the DNS resource records so that when I visited "fyp.rajeevj.co.uk", the website could be viewed. This website shown in figure 4.3 had no HTTPS connection so was unsecure to Google Chrome.

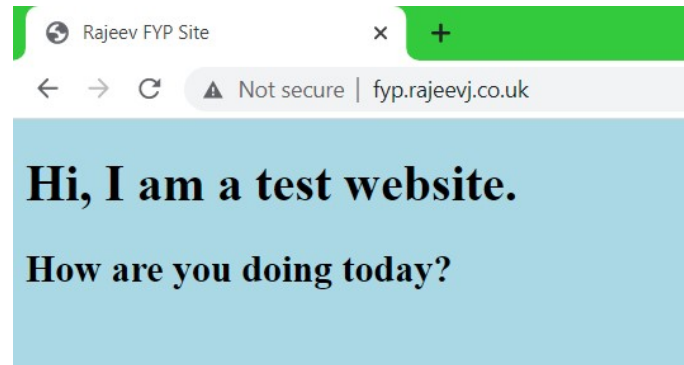


Figure 4.3: Unsecure Test Website

## 4.6 Creating a self-signed certificate

While I was developing my test website I also decided to build a website that used a self-signed certificate. As mentioned in section 2.2.1, a self-signed certificate can be dangerous. The extension will need to detect when the user visits a site like this so by developing one, I can test my extension more easily.

To build this I installed OpenSSL on the VM instance, this is an open-source command-line tool that allows users to perform various SSL-related tasks. I followed an online guide to create a self-signed certificate with OpenSSL. [Baeldung, 2022]. I combined my server certificate and root certificate into ssl-bundle.crt. I edited the nginx.conf file and restarted the nginx server.

Listing 4.7: nginx.conf

```
server {
    listen 443 ssl;
    ssl_certificate /etc/ssl/ssl-bundle.crt;
    ssl_certificate_key /etc/ssl/domain.key;
    server_name fyp.rajeevj.co.uk;
    access_log /var/log/nginx/nginx.vhost.access.log;
    error_log /var/log/nginx/nginx.vhost.error.log;
    location / {
        root /var/www/html;
        index index.html;
    }
}
```

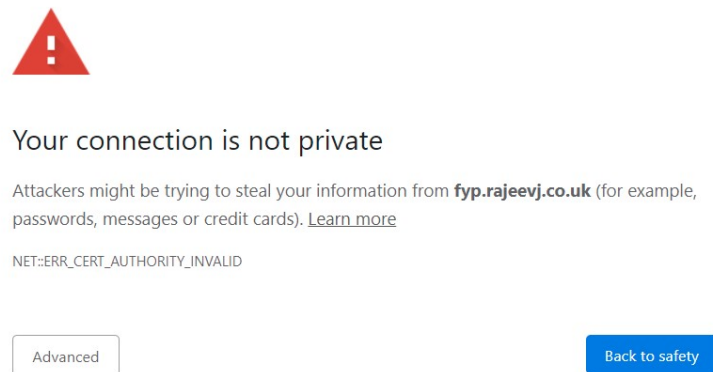


Figure 4.4: Chrome's warning before visiting site

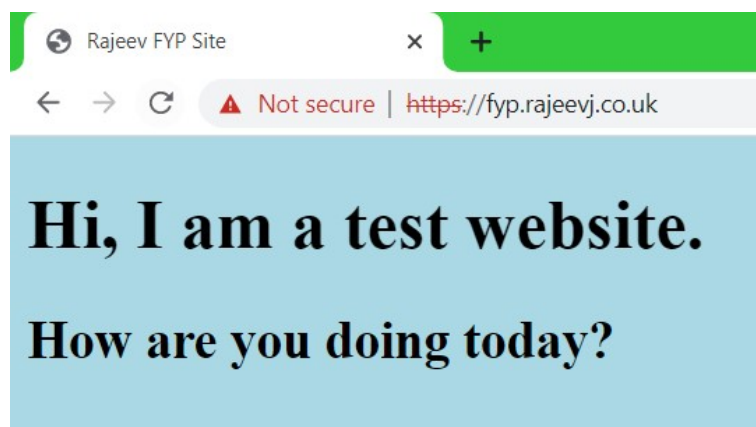


Figure 4.5: Website with self-signed certificate

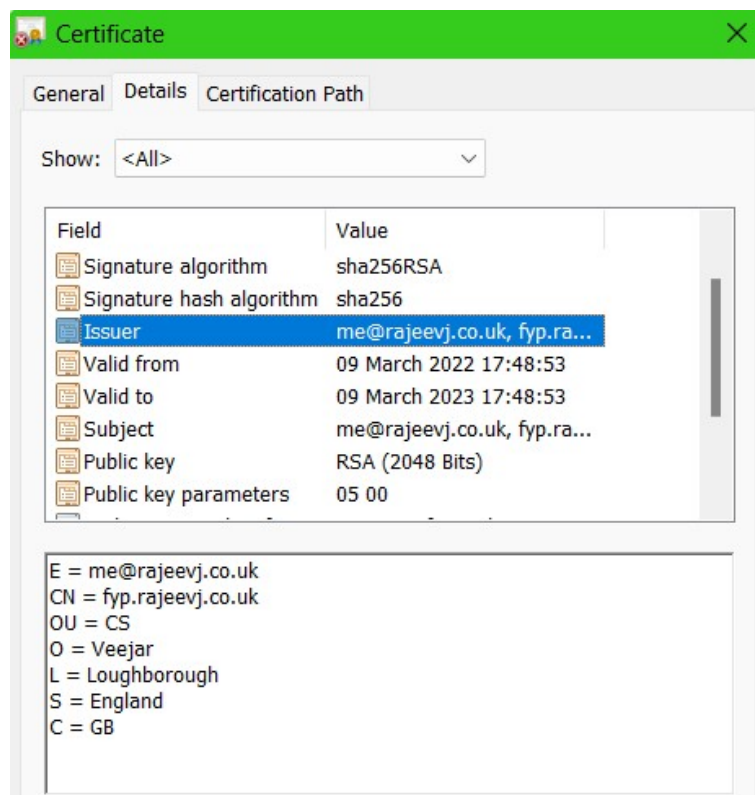


Figure 4.6: Certificate details

The certificate 'Issuer' and 'Subject' are the same. The certification path also shows the same certificate as the parent and child on the tree. These show that the website is self-signed. I now have two versions of the same site, one has no SSL and the other has a self-signed certificate.

## 4.7 Alert Message

Once the websites were ready, the extension could be tested on these sites. An alert correctly displays when the website using HTTP was visited. This is shown in figure 4.7.

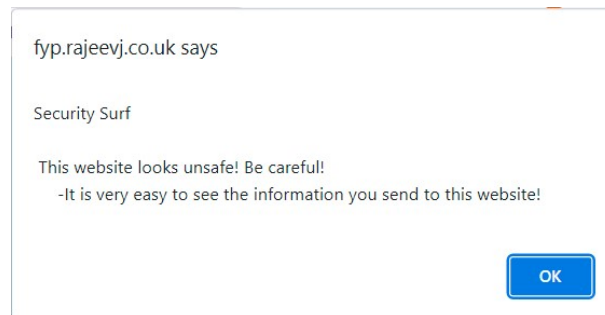


Figure 4.7: Warning message alert

## 4.8 Syncing results

In the content script I added a method of setting the score so that when the popup was opened, the file could view the score that was set for this file. To do this I stored the score in Chrome's storage.

Listing 4.8: Saving website score - content.js

```
1 chrome.storage.sync.set({ "websiteScore" :websiteScore });
```

This command uses the 'set' method which stores the score from the content script. The popup will use the 'get' method which will get the score from storage and present the data in the popup.

Listing 4.9: Getting website score - popup.js

```
1 chrome.storage.sync.get(['websiteScore'], function (result) {  
2   document.getElementById("score").innerHTML = result.websiteScore + "%";  
3 });
```

This now meant that the score was correctly being shown in the popup box as shown in figure 4.8.

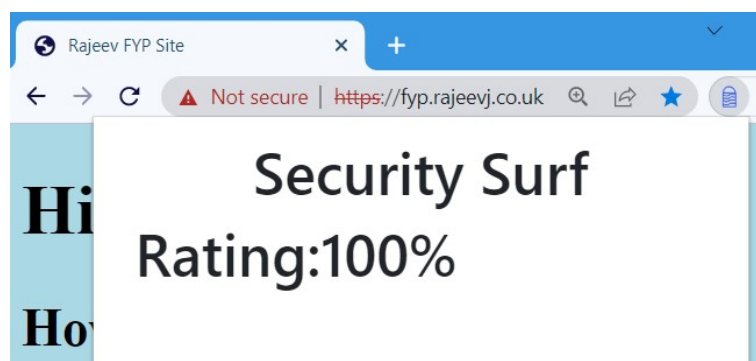


Figure 4.8: Score showing in popup

## 4.9 Chrome Web Store

In my experience with the Chrome Web Store, the team can sometimes take long reviewing new extensions. To avoid making the same mistakes again, I plan to upload a basic version to start with early on, and after it is accepted onto the store, I can update it as I develop it further.

As I was now at a point where it had a single working function, I could try to get the extension onto the Chrome Web Store. I visited the Developer Dashboard and uploaded the package. I filled in the details for the store listing, uploaded the logo and agreed to the privacy practices. Within 24 hours the extension was approved and on the store. This now meant I could start getting users straight away and when I launch the full version, everyone will get the update quickly.

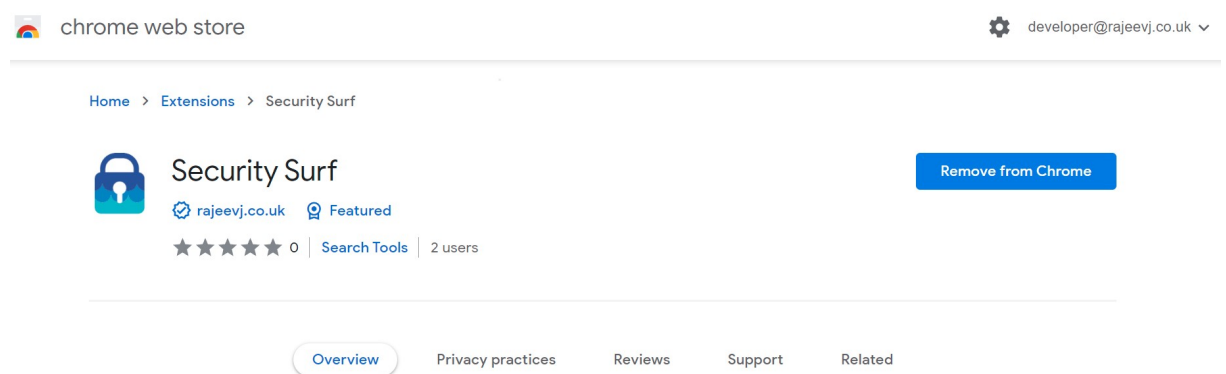


Figure 4.9: Chrome Web Store Preview

As you can from figure 4.9, my extension has my website named underneath with a tick. This represents the Established Publisher badge. [Kim, 2022] This means my identity is verified and I have established a consistently positive track record with Google services and compliance with the Developer Program Policy. Next to my verified name is also a badge which is the Featured badge. This is for extensions which adhere to Chrome Web Store's best practices and has a store listing page that is clear and helpful for users. These badges benefit me to gain more users as they can see I am a trusted and qualified developer.

## 4.10 Detecting self-signed certificates

My next step was to do further analysis of the website certificate. My extension was limited to detecting HTTP and HTTPS but could not detect websites that used self-signed certificates, my extension marked them as safe even though they are likely not to be.

One way to detect self-signed certificates was to check if the certificate 'Issuer' and 'Subject' were the same. To do this easily the browser needed to provide access to the website's certificate. More complex ways of doing this include creating a server which

downloads the certificate using OpenSSL, analysing it and returning the results each time a website was visited. This is unnecessary and may break extension policies, due to submitting user browsing activity to a third-party server.

For the browser to provide access to the certificate we needed to use an API. Chrome provides a 'chrome.certificateProvider' API but this does not allow viewing the certificate's issuer or subject fields. A JavaScript cross-browser API, 'webRequest.CertificateInfo' does provide this information to browser extensions however but it is only compatible with Firefox, not Chrome. Details of this can be found on the Chromium bugs site under issue 628819 where they discuss how implementing this into Chrome could risk security. [rolan...@gmail.com, 2022]

Since I have no way of accessing this data, I have no way of directly assessing the safety of the certificate. I will not include this extra check in my extension but this is acceptable as Chrome already does a good job of presenting these security risks to the user. These warnings can be seen in figures 4.4 and 4.5. Two security measures are already in place and so there's no need for my extension to provide a service that is already built well into the browser already. This means that requirement ID-F2 has not been met.

## 4.11 Expanding user options

My next step was to correct a bug I was having. As I stored and got the score in one place shown in code 4.8 and 4.9, this meant that when a new page was visited, the score was being overwritten and the score for the original page was incorrectly showing the score for the new page.

To prevent this I had to create a new storage method. This would be a dictionary of 'websitesVisited', this will store data about each website visited. This meant that if the user went to another tab and returned to the original tab, when the popup is clicked, data about that host name can be looked up. A domain and subdomain will be treated differently as they can be hosted differently.

By storing data about individual websites, it meant we had more control over which websites needed checking. This was a good point to implement extra features like checking the time to live for each website and giving the user a whitelist. These options are designed so as not to constantly disturb the user as they visited frequent or trusted sites. These options can be stored in a dictionary called 'extensionOptions'.

Listing 4.10: extensionOptions dictionary

```
1 extensionOptions={
2     expertiseChosen: 'beginner',
3     popupOption: 'yes',
4     TTLValue= '360000',
5     whitelist: ["www.google.com", "bbc.com"]
6 };
```

---

Listing 4.11: websitesVisited dictionary

```
1 websitesVisited={
2     fyp.rajeevj.co.uk={
3         score: '50',
```



```
4          TTL: '60000',
5          websiteSSL: 'false'
6      },
7      lboro.ac.uk={
8          score: '100',
9          TTL: '120000',
10         websiteSSL: 'true'
11     };
12 }
```

---

## Security Surf

### Options:

#### How would you rate yourself?

If you class yourself as an expert, we will explain more technically.

Choose:

#### Do you want to turn on security popups?

Do you want a warning when you visit a dangerous site? Turn on popups!

Are we popping up too much? Turn them off!

Choose:

#### How long would you like before we recheck the page's security?

This is for the websites that you visit often.

Choose:

### Whitelist

Type in the hostnames of websites that you trust:

- 

Add hostname:

Figure 4.10: Options Page with extra options

#### 4.11.1 TTL

I added a time to live value to choose for the user. This is for frequent websites that the user visits on a regular basis. Without this option available, the user may be navigating an unsecure website and a popup may appear every time they open a new page within the domain. This could get annoying so this option prevents a repeated popup showing as they navigate this website. The options the user can choose can be shown in listing 4.12 but can be changed depending on user feedback according to demand and the most popular option. This means that requirement ID-NF4 has been met.

Listing 4.12: TTL Options -options.html

```

1      <select class="form-select" id="TTLValue">
2          <option value="0">Always Check</option>
3          <option value="3600000">1 Hour</option>
4          <option value="21600000">6 Hours</option>
5          <option value="86400000">1 Day</option>
6          <option value="604800000">1 Week</option>
7          <option value="1209600000">2 Weeks</option>
8          <option value="2419200000">4 Weeks</option>
9      </select>

```

---

Listing 4.13: Storing the new website data -content.js

```

1      websitesVisited[urlHost]=
2          {"score":websiteScore, "TTL":Date.now(), "websiteSSL":websiteSSL};
3      chrome.storage.sync.set({"websitesVisited": websitesVisited});

```

---

Once the website data is in the storage, it needs to be kept for however long the user has chosen. To check this inside the background script is a listener which checks every time a new window is created. When this happens, each website's TTL is compared to the value the user chose, if the website's TTL exceeds this value, is removed from the storage. The algorithm for this is shown in listing 4.14.

Listing 4.14: Clearing website data -background.js

```

1      chrome.windows.onCreated.addListener(function() {
2          chrome.storage.sync.get(null, function(data) {
3              let chosenTTL= data.extensionOptions.TTLValue;
4              let allWebsites = data.websitesVisited;
5              //If TTL is turned on
6              if (chosenTTL>0){
7                  //Checks each website to see which are out of date
8                  for (const [websiteURL, website] of Object.entries(allWebsites)) {
9                      //Calculate Time Range
10                     let timeRange= Date.now()-website.TTL;
11                     //If it's been longer than the user sets, remove from storage
12                     if (timeRange>chosenTTL){
13                         delete allWebsites[websiteURL];
14                     }
15                 }
16                 chrome.storage.sync.set({"websitesVisited": allWebsites});
17             }//If the user always wants to check new websites, empty the list
18             else{
19                 chrome.storage.sync.set({"websitesVisited": {}});
20             }

```

```

21         });
22     });

```

---

I also added an extra feature whilst in the background script where it would open the options page when the user first installs the extension. This will not happen when there are updates as it may disturb the user if the options page opens every time there is an update.

Listing 4.15: Opening options on first install -background.js

```

1      //Initialise storage on first install
2      chrome.runtime.onInstalled.addListener(function(details) {
3          let expertiseChosen= "beginner";
4          let popupOption= "yes";
5          let TTLValue= "86400000";//1 Day in Milliseconds
6          let whiteList = ["www.google.com"];
7          let extensionOptions={
8              "expertiseChosen":expertiseChosen,
9              "popupOption":popupOption,
10             "TTLValue":TTLValue,
11             "whiteList":whiteList}
12          chrome.storage.sync.set({"extensionOptions": extensionOptions});
13          chrome.storage.sync.set({"websitesVisited": {}});
14          //If it is first install, go to options page
15          if(details.reason == "install"){
16              chrome.tabs.create({
17                  'url':'chrome-extension://'+chrome.runtime.id+"/options.html"});
18          }
19      });

```

---

### 4.11.2 Whitelist

I added a whitelist for the user to add websites into. This is for frequent trusted websites. Without this option available, the user may be frequently visiting an unsecure website and a popup may appear every time they open the page. This could get annoying so this option prevents a repeated popup showing. The options page is where users can add the domains they trust. 'www.google.com' is already listed within this as an example of how the domain should look if users are unsure (they can remove this if they like). To add a website the user can simply type in the box and click add or they can click enter on their keyboard. This is then listed on the options page, if they click the 'X' button next to a domain, this removes the domain from the list. After clicking save this stores an array of the updated whitelist in the 'extensionOptions' dictionary. This means that requirement ID-NF5 has been met.

Listing 4.16: Saving user's options -options.js

```

1      $("#saveOptions").click(function(){
2          let expertiseChosen= $('#expertiseChosen option:selected').val();
3          let popupOption= $('#popupOption option:selected').val();
4          let TTLValue= $('#TTLValue option:selected').val();
5          //Gets all the urls from the whitelist
6          let whiteList = [];
7          $('#whiteListLinks li').each(function(){
8              whiteList.push($(this).text());

```

```

9      });
10     let extensionOptions={
11         "expertiseChosen":expertiseChosen,
12         "popupOption":popupOption,
13         "TTLValue":TTLValue,
14         "whiteList":whiteList}
15     chrome.storage.sync.set({"extensionOptions": extensionOptions});
16     alert("Saved!");
17 });

```

---

Every time a new website is visited, the content script runs a check against the whitelist and the stored websites before running a new check on it. The website is checked so that the host name of the website they are currently on is not in the whitelist. The website is also checked so that the host name of the website they are currently on is not in the website's visited dictionary. The algorithm for this is shown in listing 4.17. The algorithm page is functionally complete.

Listing 4.17: Before checking score -content.js

```

1     chrome.storage.sync.get(null, function (data) {
2         let allWebsites = data.websitesVisited;
3         let whiteList = data.extensionOptions.whiteList;
4         //If the website visited is not on the whitelist
5         //And if the website visited has not already been scored before
6         if ((!whiteList.includes(window.location.hostname)) && (!
7             allWebsites.hasOwnProperty(window.location.hostname))){
8             checkScore();
9         }
10    });

```

---

## 4.12 Creating an issues section on the popup

My next step was to build the popup page in more detail. If the user had chosen to turn off popups or wanted more detail about what it said, they could click the popup to see the issues again.

To do this, the data that the content script created needed to be gotten from storage. This displays the score and uses the other variables stored to give information about the site like if SSL was used.

An important part of this was to factor in what expertise level they are, similar to how the alert worked in the content script. If the user was a beginner, they would receive a simple explanation of what the risk was, if the user was an expert, they would receive an explanation which used more technical language.

Listing 4.18: Creating issues section -popup.js

```

1     chrome.tabs.query({active: true, currentWindow: true}, function(tabs) {
2         let urlOfWebsite=tabs[0].url;
3         let hostNameOfURL= (new URL(urlOfWebsite)).hostname;
4         chrome.storage.sync.get(null, function (data) {
5             let websiteData=data.websitesVisited[hostNameOfURL]
6             let extensionOptions=data.extensionOptions;
7
8             //Display score

```

```

9         $("#ratedScore").text(websiteData.score + "%");
10
11         //Issues List
12         let issues=[];
13
14         //Check for SSL
15         if (websiteData.SSLused== "false" && (extensionOptions.expertiseChosen== "
16             beginner")){
17             issues.push("It is very easy to see the information you send to this
18                 website!");
19         } else if (websiteData.SSLused== "false" && (
20             extensionOptions.expertiseChosen== "expert")){
21             issues.push("This website uses no encryption! Data sent & recieved is
22                 in plaintext!");
23         }
24         //Adding list to HTML
25         for (item = 0; item < issues.length; item++) {
26             $("#issueList").append("<li>" + issues[item] + "</li>");
27         } //If no issues
28         if ($("#issueList li").length == 0){
29             $("#issueList").append("<h2>No Issues Found!</h2>");
30         }
31     });
32 });

```

---

### 4.13 Adding buttons to popup

The popup should have more functionality for the user. Here I would add buttons which gives the user various controls.

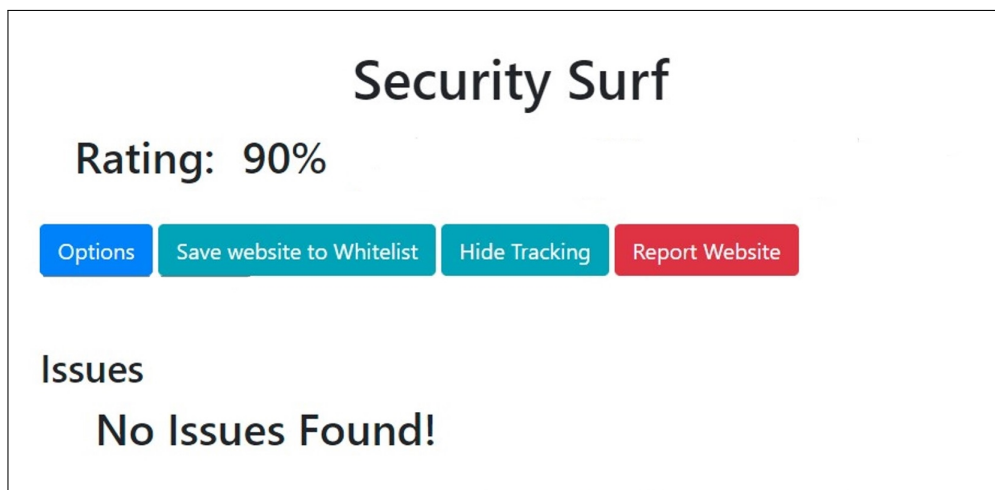


Figure 4.11: Popup Page with buttons

### 4.13.1 Options button

This button gives the user direct access to the options page.

Listing 4.19: Button for options -popup.js

```
1 //Button opens options page
2 $("#openOptionsButton").click(function(){
3   chrome.tabs.create({
4     'url': 'chrome-extension://' + chrome.runtime.id + "/options.html"});
5   });
```

---

### 4.13.2 Save website to whitelist button

This button opens is a shortcut for directly adding the website to the whitelist.

Listing 4.20: Button for saving site -popup.js

```
1 //Save button adds site to whitelist
2 $("#saveToWhitelist").click(function(){
3   whiteList.push(hostNameOfURL);
4   let extensionOptions={
5     "expertiseChosen":expertiseChosen,
6     "popupOption":popupOption,
7     "TTLValue":TTLValue,
8     "whiteList":whiteList}
9   chrome.storage.sync.set({"extensionOptions": extensionOptions});
10  alert("Added!")
11  });
```

---

### 4.13.3 Hide tracking button

This button opens the current tab in a new incognito window where Chrome won't save cookies, site data, browsing history and information entered into forms.

Listing 4.21: Button for hiding tracking -popup.js

```
1 //Button opens incognito page
2 $("#incognitoMode").click(function(){
3   chrome.windows.create({ "incognito": true, 'url': urlOfWebsite});
4   });
```

---

### 4.13.4 Report website button

This button opens the page to the National Cyber Security Centre where there is a form to report a suspicious website.

Listing 4.22: Button for reporting -popup.js

```
1 //Button opens report page
2 $("#reportWebsite").click(function(){
3   chrome.tabs.create({ 'url':
4     'https://www.ncsc.gov.uk/section/about-this-website/report-scam-website'
5     '});
6   });
```

---

## 4.14 Adding a drop-down menu to the popup

I added a drop-down menu onto the main page of the popup which would change the view. Instead of viewing the issues under the buttons, it would show different information about the website. I chose three other views, the certification information, its WHOIS information and customer reviews.

When a different value was chosen in the drop-down menu, the view would switch from issues to the corresponding view. The other views contain iFrames tags viewing information about the website using other websites.

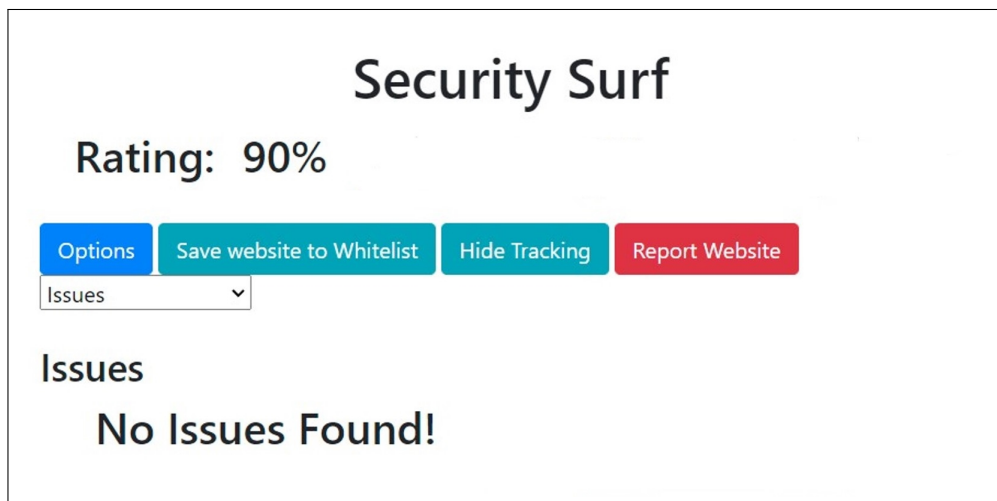


Figure 4.12: Popup Page with a drop-down menu

### 4.14.1 Certification Information

This uses an iFrame from Netcraft, it provides a site report for a given website. Netcraft is a large internet services company based in the UK. They have been in network security for over twenty-five years and are the world's largest provider of takedowns in cybercrime. The information they find for each site is reliable enough to show in my extension.

Listing 4.23: URL for certification information -popup.js

```
1 let certificateURL="https://sitereport.netcraft.com/?url="+urlOfWebsite+"#
  ssl_table_section";
2 $("#certiciateInfoIFrame").attr("src",certificateURL);
```

### 4.14.2 WHOIS Information

This uses an iFrame from WHOIS, it provides a WHOIS domain lookup report for a given website. The data they provide is convenient and simply understood with fields like registrar, expiry date and name servers.

Listing 4.24: URL for WHOIS information -popup.js

```

1      let whoIsURL="https://www.whois.com/whois/"+hostNameOfURL;
2      $("#whoIsIFrame").attr("src",whoIsURL);

```

---

### 4.14.3 Trustpilot Information

This uses an iFrame from Trustpilot, it provides customer reviews about a company. Trustpilot has over one hundred and twenty million reviews and is used globally. As the reviews can be written by anyone, that means the information users find on this website must be taken sceptically. This is written clearly above the iFrame for good measure.

Listing 4.25: URL for trustpilot information -popup.js

```

1      let trustPilotURL="https://www.trustpilot.com/review/"+hostNameOfURL;
2      $("#trustPilotIFrame").attr("src",trustPilotURL);

```

---

## 4.15 Hyperlink Checks

At this point, the only feature that the extension checked is if the website used HTTPS. The next steps were to check the hyperlinks on the website. A variety of checks can be done on these. Once all the links were collected into an array, it could be calculated which hyperlinks belonged to the same domain, which hyperlinks used HTTPS and which hyperlinks were valid.

The reason for this is because the page the user is currently on may be safe but the page that it leads the user to could be dangerous. Gathering statistics like this gives the user a closer insight into the website.

Listing 4.26: Algorithm for hyperlink checks -content.js

```

1      function hyperlinkChecks(urlHost){
2          let hostNameMatch=0;
3          let secureSSLMatch=document.links.length;
4          let falseWebsites=0;
5          //Loops through each hyperlink on webpage
6          for(var linkIndex=0; linkIndex<document.links.length; linkIndex++) {
7              let link=document.links[linkIndex].href;
8              linkURL=new URL(link);
9              if (linkURL.hostname==urlHost){
10                 hostNameMatch+=1;
11             }if (linkURL.protocol=="http:"){
12                 secureSSLMatch-=1;
13             }if (!urlChecker(link)){
14                 falseWebsites+=1;
15             }
16         }return {
17             "hostNameMatch":hostNameMatch, "secureSSLMatch":secureSSLMatch, "
18             falseWebsites":falseWebsites,"noOfLinks":document.links.length};
19     }
20     hyperlinkInfo = hyperlinkChecks(urlHost);

```

---



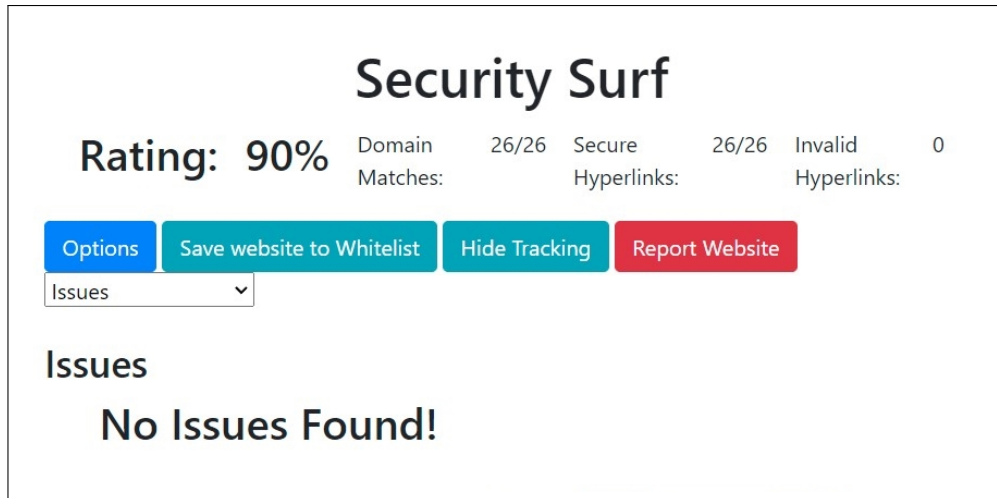


Figure 4.13: Popup Page with statistics

#### 4.15.1 Domain matches

Domain matches refer to the number of hyperlinks which go to another page within the same domain. A website that goes to lots of other websites online may be suspicious. Especially when the user expects to stay within that website while navigating around. This means that requirement ID-F3 has been met.

#### 4.15.2 Secure hyperlinks

Secure hyperlinks refer to the number of hyperlinks which uses the HTTPS protocol. A website which has none or a small number of these may be suspicious, especially if the website is being given personal information. It is also a sign that the website might be old which means that it has not been updated in a while, this could mean the website has low-security features. This means that requirement ID-F4 has been met.

#### 4.15.3 Invalid Hyperlinks

Invalid hyperlinks refer to the number of hyperlinks which followed the correct standard for a URL. To check this, REGEX is used, I found online an expression which checked the validity of a URL. As this is a good way to check for valid URLs, I also added it to the options page. This detects the URL the user wants to add to the whitelist and rejects it if the URL is not valid. This means that requirement ID-F5 has been met.

Listing 4.27: Algorithm for validating links

```

1  function urlChecker(url){
2      let safeWebsite=false;
3      var regex = new RegExp(/[-a-zA-Z0-9@:%._\+~#=]{1,256}\.[a-zA-Z0-9()]{1,6}\b([
4      if (url.match(regex)) {
5          safeWebsite=true;

```

```

6         }return (safeWebsite)
7     }

```

---

## 4.16 Adding APIs

There are a lot of external tools which can easily check website security and provide useful information. A lot of these tools are free and have responsive results. To test these I used Postman, this is an API platform I used to test API responses. The platform has over twenty million registered users and constitutes the world's largest public API hub.

### 4.16.1 Google Safe Browsing

A basic but very useful API is Google's Safe Browsing API. [Safebrowsing, 2022] I added the Safe Browsing Lookup API (v4) to my extension which ran a check of any type across any platform. To set this up I enabled the API in the Google Cloud project I created in section 4.5. I created an API key to use in credentials which worked in my extension.

This API provided a simple check to see if the URL is included on any safe browsing lists, if they are, I could rate them with a low score immediately.

Listing 4.28: Google API Call -content.js

```

1      //SafeBrowsingLookupAPI Call
2      function SafeBrowsingLookupAPI (url){
3          const googleSafeBrowsingApiKey="";
4          let websiteWithKey="https://safebrowsing.googleapis.com/v4/
              threatMatches:find?key="+googleSafeBrowsingApiKey;
5          requestBody=
6              {
7                  "client": {
8                      "clientId": "securitysurf",
9                      "clientVersion": "1.1"
10                 },
11                 "threatInfo": {
12                     "threatTypes": ["THREAT_TYPE_UNSPECIFIED", "MALWARE", "
                        SOCIAL_ENGINEERING", "UNWANTED_SOFTWARE", "
                        POTENTIALLY_HARMFUL_APPLICATION"],
13                     "platformTypes": ["ALL_PLATFORMS"],
14                     "threatEntryTypes": ["URL"],
15                     "threatEntries": [
16                         {"url": url} //Example of dangerous URL
17                         //http://testsafebrowsing.appspot.com/s/phishing.html
18                     ]
19                 }
20             };
21          const options = {
22              method: 'POST',
23              headers: {'Content-Type': 'application/json'},
24              body: JSON.stringify(requestBody)
25          };
26          fetch(websiteWithKey,options)
27              .then((response) => {
28                  return response.json();

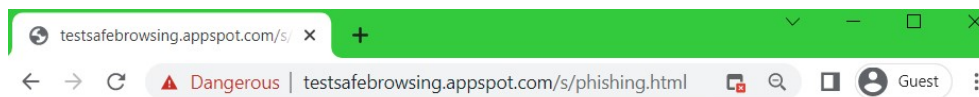
```

```

29         })
30         .then((data) => {
31             let badResults = data;
32             //If there are no bad results
33             if (Object.keys(badResults).length === 0){
34                 return(true);
35             }else{
36                 return(false);
37             }
38         })
39         .catch(function(error) {
40             return(error);
41         });
42     }
43     isWebsiteSafe=SafeBrowsingLookupAPI(url);

```

Below in figure 4.15, an issue appears when visiting a dangerous site, this site is a test website provided by Google shown in figure 4.14.



## Phishing Page Example.

This is an example of a phishing page.

Username:   
 Password:

## Social engineering content example.

This image claims that your software is out-of-date to trick you into clicking "update".

This image mimics a dialogue from the FLV software developer -- but it does not actually originate from this developer.

These buttons seem like they will produce content that relate to the site (like a TV show or sports video stream) by mimicking the site's look and feel. They are often not distinguishable from the rest of the page.

Figure 4.14: Google's phishing website for testing

## Issues

- It is very easy to see the information you send to this website!
- Google Safe Browsing says this website is dangerous to use!!

Figure 4.15: Issue appears when visiting dangerous site

### 4.16.2 IP Geolocation API

It is also good to check the location of the hosted website. If the website claims to be hosted in a country it is not being hosted in, this is suspicious. For example, a government healthcare website that claims to store all of its data in the same country could come up as being hosted somewhere else, this is a sign that the website is fake or a scam.

The IP Geolocation API [ip api, 2022] reports where a website is being hosted, it is free for non-commercial use, uses no API key and is easy to integrate. Once I received the location from the API I needed to compare it to the user's location. To do this, I used the user's timezone and added the Public Suffix List JavaScript library to calculate the user's location from this. If the countries did not match, the user would be notified of an issue in the popup box.

Listing 4.29: IP Geolocation API -popup.js

```

1      //IPGeolocationAPI Call
2      async function IPGeolocationAPI(url){
3          let locationData = new Promise ((resolve, reject) => {
4              let websiteCall='http://ip-api.com/json/'+url;
5              fetch(websiteCall)
6                  .then((response) => {
7                      return response.json();
8                  })
9                  .then((data) => {
10                     let locationData = data;
11                     resolve(locationData)
12                 })
13                 .catch(function(error) {
14                     reject(error);
15                 });
16            })
17            let location = await locationData;
18            let timezone = Intl.DateTimeFormat().resolvedOptions().timeZone;
19            let currentCountry = timezones[timezone].c[0];
20            if(currentCountry!=location.countryCode){
21                $("#issueList").append("<li>This website is being delivered from  
another country: <b>" +countries[location.countryCode]+"</b></li>  
");
22            }
23        }
24        IPGeolocationAPI(hostNameOfURL);

```

---

### 4.16.3 urlscan.io

The website urlscan.io [Gilger, 2022] is a free service to scan and analyse websites. This website contains an API which provides a lot of WHOIS data about a domain. This can be used in various ways, for example, a website that was recently registered should be used with precaution. This means that requirement ID-F6 has been met.

Listing 4.30: urlscan.io API -popup.js

```

1      //URLScanIOAPI Call
2      async function URLScanIOAPI(url){

```

```

3       let WHOISdata = new Promise ((resolve, reject) => {
4           let websiteCall='https://urlscan.io/api/v1/search/?q='+url;
5           fetch(websiteCall)
6               .then((response) => {
7                   return response.json();
8               })
9               .then((data) => {
10                  resolve(data.results[0].page);
11              })
12              .catch(function(error) {
13                  reject(error);
14              });
15          })
16      let pageData = await WHOISdata;
17  }
18  URLScanIOAPI(domainOfURL);

```

---

## 4.17 Cookies

My next step was to provide information about cookies to the user. Unsafe websites will have lots of third party cookies which track the user online and can be used for targeted advertising, selling private information and more.

To incorporate this, Google has a 'Chrome.cookies' API to query and modify cookies. Content scripts are limited to which Chrome APIs can run which meant this had to be run in the background script. I could count how many cookies were third party cookies and the higher the number, the more unsafe the website was. I could also notify the user that the website was using multiple trackers.

As I did further research I found that third party cookies were going to become unnecessary in the future. I decided not to add this feature to my extension as the feature would become irrelevant. This means that requirement ID-F7 has not been met. In Google's official Chromium Blog [Schuh, 2020], they announced that they were removing third parties in Google Chrome, which they called the 'Privacy Sandbox'. They made an update in 2021 stating their goal is to phase out third-party cookies over a three month period, starting in mid-2023 and ending in late 2023. [Goel, 2021]

## 4.18 Detecting adverts on pages

Another feature of unsafe websites is lots of adverts that pop up and overwhelm the user. The adverts often appear suddenly, in large areas and can cause links to open without cause. These adverts appear as iFrames on the website, in the style attribute of each iFrame tag, they contain lots of values including the '!important;' value.

This is a CSS rule which overrides all previous styling rules for that specific property on that element. I can use this to detect if a website has these popups. I run this check every minute as the popups do not always appear straight away, they may appear after some time. This may mean my extension does more computation than usual so it is important that I do not run this check too often, but it is also important to catch this

quickly before the user assumes the site is safe. This means that requirement ID-F8 has been met.

Listing 4.31: Checking iFrames -content.js

```

1  //Counts suspicious iFrames
2  let suspiciousIframes=0
3  setInterval(function(){
4      //Gets all iframes on page
5      let iFrames = document.getElementsByTagName("iFrame");
6      let countIframes=0;
7      for (let i = 0; i < iFrames.length; i++) {
8          let styleAttribute=iFrames[i].getAttribute("style");
9          if(styleAttribute){
10             //Counts how many "!important;" values are in the CSS of the iframe
11             let countImportant=(styleAttribute.match(/!important;/g) || []).length;
12             if (countImportant>3){//If there are more than 3
13                 countIframes+=1;//It's a sign that it's a dangerous popup
14             }
15         }
16     } //Set max value globally
17     if (suspiciousIframes<countIframes){
18         suspiciousIframes=countIframes;
19     }//If there is atleast 1 suspicious Iframe, set the score to 0
20     if (suspiciousIframes){
21         score=0;
22     }
23 }, 60000);//Checks every 60 seconds if IFrames appear

```

## Issues


- Some of the links aren't actual websites
- Google Safe Browsing says this website is dangerous to use!!
- This website has suspicious content on it! 
- This website is being delivered from another country: **Canada**

Figure 4.16: Issue appears when a suspicious iFrame is found

## 4.19 Learning more about website security

It is essential that the user learns as they browse the internet. Beginners may not understand what is being shown to them in the drop-down menu. These sections include certificate information, WHOIS information and Trustpilot reviews. In each of these sections, I added a 'Learn More' button which replaces the iFrame with some text explaining the topic. At the bottom of this text is also some links to further reading and videos to help the user to gain cyber security awareness.

## Connection Details

Certification information, find out if your connection is encrypted using SSL and more.

[Learn more](#)

- If the website has a certificate, Chrome will show a lock in the top left.
  - This means the website has a secure connection between you and the website
  - This helps the website with authentication, it knows you are who you say you are
  - It also means anyone on the internet can't see the data that is sent or received from this device
  - Certificates are signed by trusted authorities
- If a website has no certificate, Chrome will say 'Not secure' in the top left.
  - This website has weak privacy
  - Anyone on the network can see the data you send and receive from it
- If it says Certificate Check: Self Signed Certificate
  - This website has been given a certificate by itself (instead of a trusted authority)
  - There can be multiple reasons for this, but in general proceed with caution
  - Luckily Chrome warns you of these websites before you reach the page
- Want to learn more?
  - [YouTube Explanations](#)
  - [What Is an SSL Certificate? A Beginners Guide](#)
  - [SSL and SSL Certificates Explained For Beginners](#)

Figure 4.17: Learning more about certificates

## Website Details

WHOIS information, take a look the website's hosting information and more.

[Learn more](#)

- WHOIS Information
  - WHOIS is lots of information about a single domain
  - A domain is the name of the website, like 'google.com' or 'bbc.co.uk'
  - Includes details like the domain owner, the date it was registered and information about the servers
  - Sometimes information about the domain can be hidden from the public eye
- Why are WHOIS records kept:
  - If an unsafe website is reported, authorities can see who owns the domain and remove them
  - If a website is registered recently, e.g. within the last year, it may be suspicious
  - If a website is posing to be from one country but is hosted somewhere completely different, this may also be suspicious
- Want to learn more?
  - [YouTube Explanations](#)
  - [Try looking up a website yourself!](#)
  - [What is Whois Information and why is it Valuable?](#)

Figure 4.18: Learning more about WHOIS



## Trustpilot Reviews

Take this with a pinch of salt, these opinions are written by anyone!

Learn more

- What is Trustpilot
  - Trustpilot is consumer review website
  - Customers review businesses to help other customers decide whether to use the service
- How does it help
  - Use this website when visiting websites that offer services
  - See how popular the website/business is
  - See how good the feedback is from the public
  - *Remember these are written by anyone so don't believe everything you see*
- Want to learn more?
  - [YouTube Explanations](#)
  - [Try the website out yourself!](#)
  - [What Is Trustpilot And Can You Trust Their Reviews?](#)

Figure 4.19: Learning more about Trustpilot

## 4.20 Help Button

The statistics on the main page of the popup can also be confusing or misunderstood. To fix this I added a 'Help' button on the main page which replaces the issues section with text explaining the extension's functions and meanings.

## Security Surf Explained

- Rating: This is a percentage how safe we find the website based on lots of factors
- Statistics
  - Domain Matches: Hyperlinks found on the website which lead to another website from the same domain
  - Secure Hyperlinks: Hyperlinks found on the website which are secured by SSL
  - Invalid Hyperlinks: Hyperlinks found on the website which are not recognised URLs
- Buttons
  - Options: Takes you to the options page to change your settings
  - Save to whitelist: If you trust the website, you can save us scanning it
  - Hide Tracking: Opens website in Incognito Mode where history, cookies and form data isn't saved
  - Report Website: Opens a form to report the website to official authorities
- Dropdown: Find out more details about the website
- Issues: These are some of the things we noticed when we scanned the website.

Figure 4.20: Information the help button displays



## 4.21 Design

My Chrome extension needs to be easy to use for beginners. To help with this the extension needs to have a good graphical user interface, it allows clear communication between the user and the extension. My goal is to make the user's interactions with my extension as simple and efficient as possible. To give this clarity, I redesigned the popup and options pages. This means that requirement ID-NF1 has been met. I added bootstrap classes into HTML tags and created a colour scheme according to the blue logo.

I also added icons onto the buttons on the homepage of the popup. These were provided from Google's Material Symbols and allowed me to choose from over 2500 glyphs.

Listing 4.32: An example of Bootstrap & Google's Material Symbols being used - popup.html

```

1      <div class="m-1 p-1"
2          style="font-variation-settings: 'FILL' 0, 'wght' 100, 'GRAD' 0, 'opsz' 20;"
3          <button id="openOptionsButton" type="button" class="btn btn-primary">Options
4              <span class="material-symbols-outlined">settings</span></button>
5          <button id="saveToWhitelist" type="button" class="btn btn-success">
6              Save website to Whitelist
7              <span class="material-symbols-outlined">bookmark</span></button>
8          <button id="incognitoMode" type="button" class="btn btn-dark">Hide Tracking
9              <span class="material-symbols-outlined">cookies</span></button>
10         <button id="reportWebsite" type="button" class="btn btn-danger">Report Website
11             <span class="material-symbols-outlined">report</span></button>
12         <button id="helpButton" type="button" class="btn btn-warning">Help
13             <span class="material-symbols-outlined">help</span></button>
14
15         <select class="m-1 form-select btn btn-secondary dropdown-toggle"
16             name="chooseInfoBox" id="chooseInfoBox">
17             <option value="websiteIssues">Issues</option>
18             <option value="certInfo">Connection Details</option>
19             <option value="WhoIsInfo">Website Details</option>
20             <option value="trustPilot">Trustpilot Reviews</option>
21         </select><br><br>
22     </div>

```

---

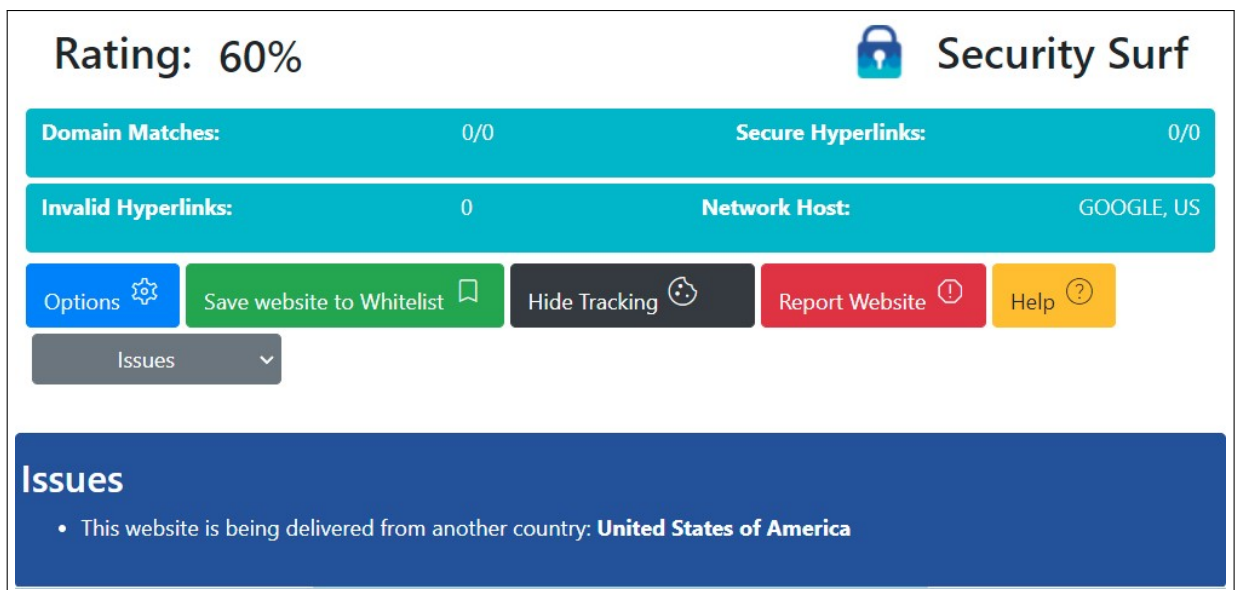


Figure 4.21: Updated design of the popup page

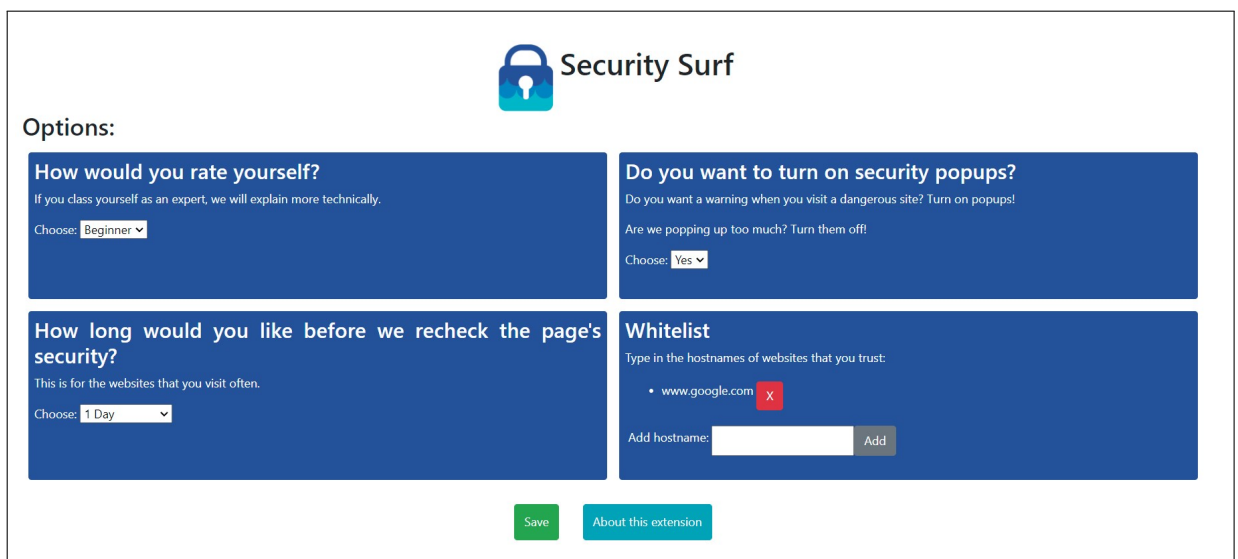


Figure 4.22: Updated design of the options page

## 4.22 Traffic Light Icons

As an extra feature, I decided to add a traffic light system to the extension's icon. When the user visits a website, the user can immediately tell whether the extension suspects the site to be safe, moderately safe or unsafe.

If the website is considered safe, the icon will show a green bubble in the corner of the extension's icon. If it is moderately safe, the bubble will be amber and if it is unsafe, the bubble will be red. If the user navigates to a new website, the icon will change and when they click back to the original site, the colour corresponding to that tab will reappear. As I could not use the necessary Chrome API in the content script, Chrome's Message Passing was used to send the content to the background script where it could change the icon. This means that requirement ID-NF6 has been met.

Listing 4.33: Sending message -content.js

```

1
2      function storeWebsiteInfo(websiteScore,urlHost,websiteSSL,hyperlinkInfo,
3          isWebsiteSafe,suspiciousIframes){
4          //Set Icon Colour traffic light
5          chrome.runtime.sendMessage({"urlHost": urlHost, "score":score});
6          chrome.storage.sync.get("websitesVisited", function(websiteResults){
7              let oldResults = websiteResults.websitesVisited;
8              oldResults[urlHost]={
9                  "score":websiteScore,
10                 "TTL":Date.now(),
11                 "websiteSSL":websiteSSL,
12                 "hyperlinkInfo":hyperlinkInfo,
13                 "isWebsiteSafe":isWebsiteSafe,
14                 "suspiciousIframes":suspiciousIframes
15             };
16             chrome.storage.sync.set({"websitesVisited": oldResults});
17         });
18     }

```

Listing 4.34: Changing the icon's colour -background.js

```

1
2      //Change Icon depending on website score
3      chrome.runtime.onMessage.addListener(
4          function(request, sender) {
5              let iconText=".";
6              //Green Light
7              if (request.score>75){
8                  chrome.action.setBadgeText( { text: iconText, tabId:sender.tab.id } );
9                  chrome.action.setBadgeBackgroundColor({color: '#0FFF50',
10                      tabId:sender.tab.id});
11              }//Amber Light
12              else if (request.score>35){
13                  chrome.action.setBadgeText( { text: iconText, tabId:sender.tab.id } );
14                  chrome.action.setBadgeBackgroundColor({color: '#F9AF12',
15                      tabId:sender.tab.id});
16              }//Red Light
17              else if (request.score<=35){
18                  chrome.action.setBadgeText( { text: iconText, tabId:sender.tab.id } );
19                  chrome.action.setBadgeBackgroundColor({color: '#FF3131',
20                      tabId:sender.tab.id});
21              }//New website has no score, default image
22              else{
23                  chrome.action.setBadgeText( { text: "", tabId:sender.tab.id } );
24                  chrome.action.setBadgeBackgroundColor({color: '#2f2f2f',
25                      tabId:sender.tab.id});
26              }
27          }
28      );

```

```
23         return(true);  
24     }  
25 );
```

---



Figure 4.23: Icons with traffic light bubbles

## 4.23 About section

On the options page I added a small section which when clicked it would open a box. This box contains my name, my website and a small disclaimer. This disclaimer is there so users understand that this extension is not perfect and should not be completely relied on to tell if a website is dangerous. 3.4

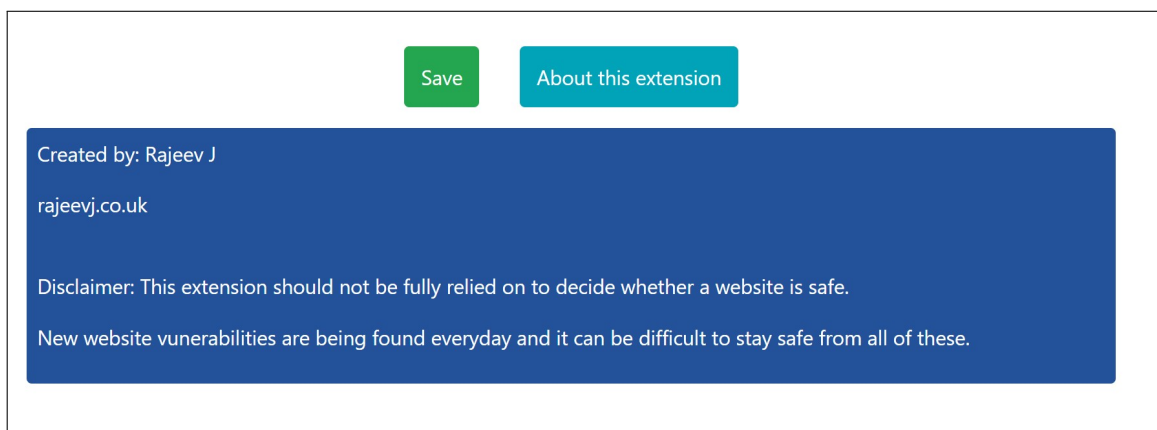


Figure 4.24: About section on options page

## Chapter 5

# Testing

Testing is important to do while developing so that bugs can be discovered quickly. Lots of testing guarantees the quality of my extension and makes it reliable. As my tool is used in cyber security, it is important that it is robust and keeps up with the changes that constantly happen. There are constantly new ways of creating fake websites and exploiting vulnerabilities on websites and the features I have built must keep up with this demand.

Throughout my development I completed unit testing on each function I created, this tested for normal and erroneous values. Some boundary testing was completed but this will be completed further during stress testing.

### 5.1 Requirements Testing

Requirements testing involves ensuring my extension meets the criteria I set in the planning phase. Below in table 3.1 I display the results for the functional requirements. Table 3.2 displays the results for the non-functional requirements of this extension.

### 5.1.1 Functional

ID	Functional Requirement	Input	Output	Comment
ID-F1	Extension can detect if the website is using an HTTP or HTTPS connection	http://fyp.rajeevj.co.uk & https://www.lboro.ac.uk/	See figure 4.8 for result of secure site	Met criteria.
ID-F2	Extension can detect if the website is signed using a self-signed certificate	https://fyp.rajeevj.co.uk	See section 4.10 for discussion	Will not be implementing this feature.
ID-F3	Extension can detect if the hyperlinks on the website are in the same domain they are already on	https://www.lboro.ac.uk/	See figure 4.13	Met criteria.
ID-F4	Extension can detect if the hyperlinks on the website are using HTTPS	https://www.lboro.ac.uk/	See figure 4.13	Met criteria.
ID-F5	Extension can detect if the hyperlinks on the website are valid URLs	https://www.lboro.ac.uk/	See figure 4.13	Met criteria.
ID-F6	Extension can use WHOIS/ RDAP information to determine how safe a website is	http://fyp.rajeevj.co.uk	See figure 4.21	Met criteria.
ID-F7	Extension can analyse cookies in websites	N/A	See section 4.17 for discussion	Will not be implementing this feature.
ID-F8	Extension can detect adverts and pop-ups	https://ww.123movies.hair/	See figure 4.16	Met criteria.
ID-F9	Extension can detect a dangerous website	http://testsafebrowsing.appspot.com/s/phishing.html	See figure 4.14 & 4.15	Met criteria.

Table 5.1: Functional Testing

All of the functional requirements were met apart from ID-F2 and ID-F7. ID-F2 was to get my chrome extension detecting self-signed certificates, this was not feasible as the APIs available did not allow it and the browser already did a good job of notifying the user. ID-F7 was to analyse the cookies, as Chrome heads towards a 'Privacy Sandbox', this feature would quickly become irrelevant.

The requirements related to APIs, while they worked in the testing environment, this does not mean they will always work. Google is a reliable company so it can be expected the Google Safe Browsing API will have no downtime. The other two APIs are not as important in function so any downtime will not affect the extension largely.

Requirement ID-F8 was built to detect suspicious iFrames. There are a wide variety of methods that developers use to display unsafe banners and adverts that pop up, the method I used worked for the few websites I tested however it can not be relied on that it will work on every page that a user visits. Any user that visits a website like this should notice these large adverts and so may not need the extension to notify them that they are there.

### 5.1.2 Non-Functional

ID	Non-Functional Requirement	Input	Output	Comment
ID-NF1	Extension should have a simple user interface, easy to use	User Testing	See results in section 5.2	Simple design but could be improved.
ID-NF2	Extension should have option to decide on level of expertise	Options page has drop-down menu for "Beginner" or "Expert". <a href="http://fyp.rajeevj.co.uk">http://fyp.rajeevj.co.uk</a>	See figure 4.18 which displays the message for different user levels	Met criteria.
ID-NF3	Extension should have option to disable popups	Options page has drop-down menu for "Yes" or "No". <a href="http://fyp.rajeevj.co.uk">http://fyp.rajeevj.co.uk</a>	When "Yes", see figure 4.7, when "No", popup does not appear.	Met criteria.
ID-NF4	Extension should have option to change how long before the popups show up again for all websites	TTL value chosen as 1 minute. <a href="http://fyp.rajeevj.co.uk">http://fyp.rajeevj.co.uk</a>	Popup returns after 1 minute.	Met criteria, requires user to open a new window to run delete function.
ID-NF5	Extension should have option to whitelist trusted websites	<a href="http://fyp.rajeevj.co.uk">http://fyp.rajeevj.co.uk</a> added	Before adding, popup appears. After adding, no popup and warning message appears if clicking popup.	Met criteria.
ID-NF6	Extension icon should change based on website rating using a traffic light colour system	<a href="https://www.lboro.ac.uk">https://www.lboro.ac.uk</a> Safe. <a href="http://fyp.rajeevj.co.uk">http://fyp.rajeevj.co.uk</a> Moderately Safe. <a href="http://testsafebrowsing.appspot.com/s/phishing.html">http://testsafebrowsing.appspot.com/s/phishing.html</a> Dangerous	Icon changes to green, amber or red respectively	Met criteria.
ID-NF7	Extension should have a welcome page on first installation to be used for a tutorial	User first installs extension from store	Options page opens	Need to create a web page that opens on the first install.
ID-NF8	Extension should clear the user data annually for extra security	Wait a fixed amount of time	N/A	Need to create function that wipes the user data

Table 5.2: Non-Functional Testing

All the non-functional requirements were fully met apart from requirements ID-NF1, ID-NF4, ID-NF7 and ID-NF8. ID-NF1 is about usability and design choices, further discussion of this is in the next section 5.2.

ID-NF4 refers to the TTL value the user sets. The website data is only deleted when

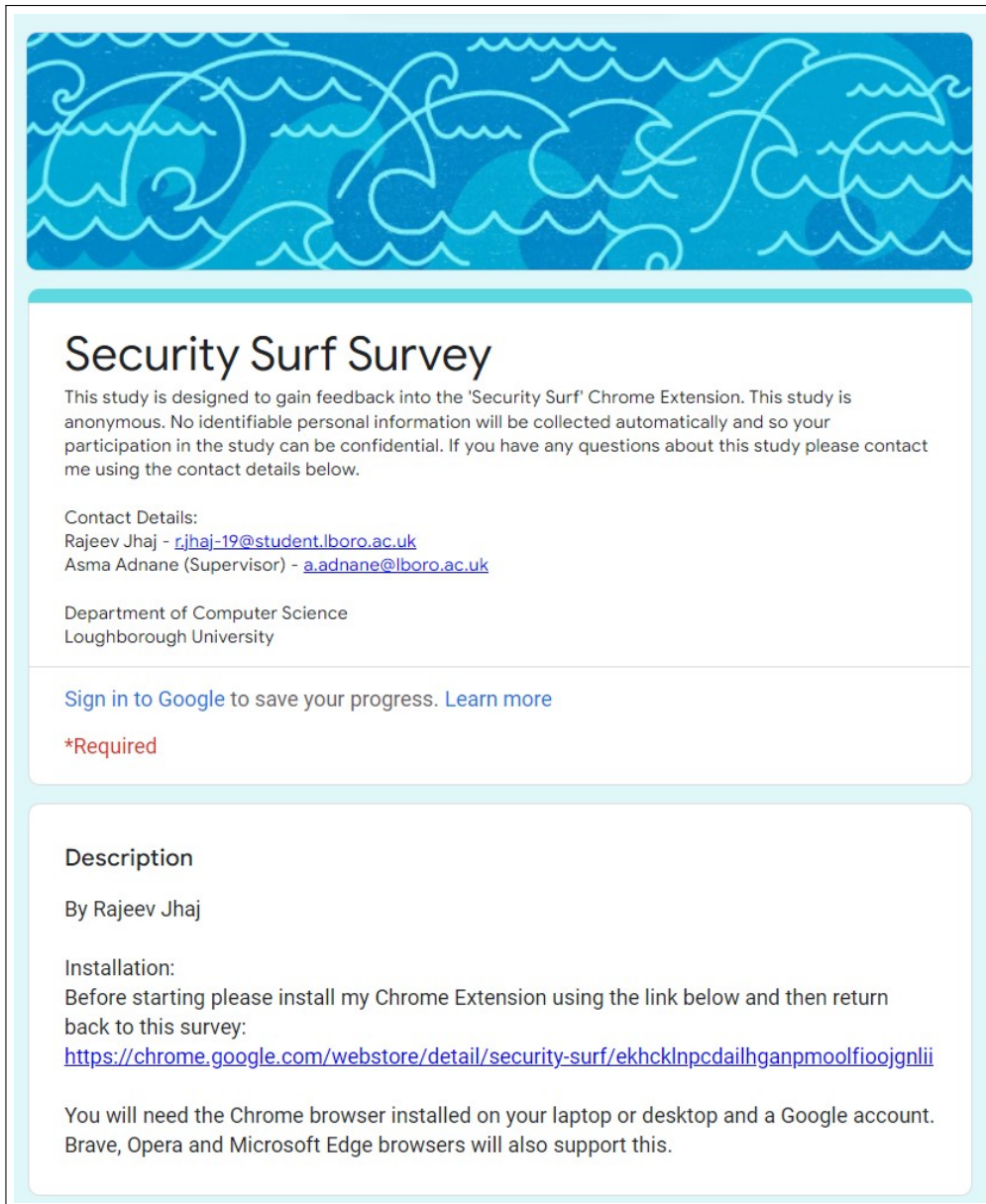
a check runs, this check function only runs when a new window is opened. If the user chooses a low TTL value but does not open a new window regularly, then the websites are not wiped as often as the user prefers. To get around this, I could change the check function to run every time there is a tab action, however, this would run the check too often and may slow down the browser unnecessarily. If the websites are not wiped often enough this does not pose much of a risk, I could add a notice in the help guide stating this.

ID-NF7 has not been completed. In a future update, I can create a web page which explains the functionality of the extension in simple terms when the user first installs the extension. ID-NF8 has also not been completed. This requires a similar function to the one which clears the user data periodically but wipes the user's options like the whitelist and the websites visited.

## **5.2 User Testing**

It is important to do beta testing to get feedback from the target audience. Once the extension was on the Chrome store it meant that stakeholders had an easy way of installing it. They installed the version which had the design features finished in section 4.21. To get feedback from users I sent out a questionnaire.



The image shows a web page for a survey titled "Security Surf Survey". At the top is a blue header with a white wave pattern. Below the header, the title "Security Surf Survey" is displayed in a large, bold, black font. Under the title, a paragraph explains that the study is anonymous and provides contact details for Rajeev Jhaj and Asma Adnane. A link to "Sign in to Google" is provided to save progress. A red asterisk and the word "Required" are shown below the link. The "Description" section follows, stating the survey is by Rajeev Jhaj and providing installation instructions, including a link to the Chrome Web Store. It also mentions that Chrome, Brave, Opera, and Microsoft Edge browsers are supported.

**Security Surf Survey**

This study is designed to gain feedback into the 'Security Surf' Chrome Extension. This study is anonymous. No identifiable personal information will be collected automatically and so your participation in the study can be confidential. If you have any questions about this study please contact me using the contact details below.

Contact Details:  
Rajeev Jhaj - [r.jhaj-19@student.lboro.ac.uk](mailto:r.jhaj-19@student.lboro.ac.uk)  
Asma Adnane (Supervisor) - [a.adnane@lboro.ac.uk](mailto:a.adnane@lboro.ac.uk)

Department of Computer Science  
Loughborough University

[Sign in to Google](#) to save your progress. [Learn more](#)

**\*Required**

**Description**

By Rajeev Jhaj

Installation:  
Before starting please install my Chrome Extension using the link below and then return back to this survey:  
<https://chrome.google.com/webstore/detail/security-surf/ekhcklnpcdailhganpmoolfioojgnlii>

You will need the Chrome browser installed on your laptop or desktop and a Google account. Brave, Opera and Microsoft Edge browsers will also support this.

Figure 5.1: First page of questionnaire

Questions can be viewed in Appendix B.

### 5.2.1 Results

I received responses from 7 people. This is adequate to help improve my extension's design and functionality however having more responses would have been beneficial. Not getting enough responses from a large audience means I can not be statistically

valid when going through the results. A larger sample size improves the credibility of my results, helps to identify any anomalies and the average becomes more reliable. However, getting a large sample size is challenging and takes time.

The method of data collection was online responses. This method is easy and convenient for respondents, they can complete the survey at any time and place with an internet connection and appropriate device. On the other hand, gathering data in person has its own benefits. More rich data is collected this way, like monitoring how users navigate websites and the extension, they might act unexpectedly and I can modify my extension in response. Through this method, I can receive higher data quality but it is more difficult to collect data in this way.

## Section 1 - Introduction

What is your age?

7 responses

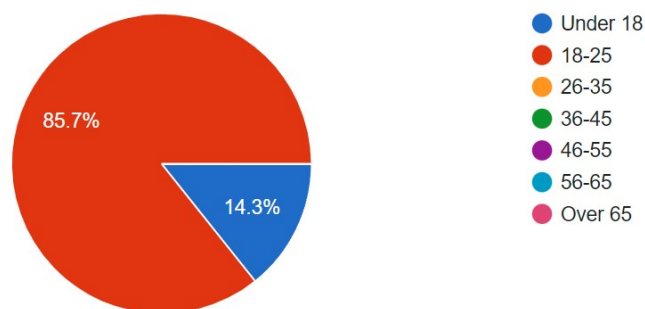


Figure 5.2: Survey results for age

All my responses were from people under the age of 25. This could bias my results toward the younger generation who have a better understanding of technology than the older generation. This means my feedback is from some portion of my stakeholders as I wanted this to be used by both the younger and older generation for different reasons discussed in section 3.1.

How would you rate your cyber security skills?

7 responses

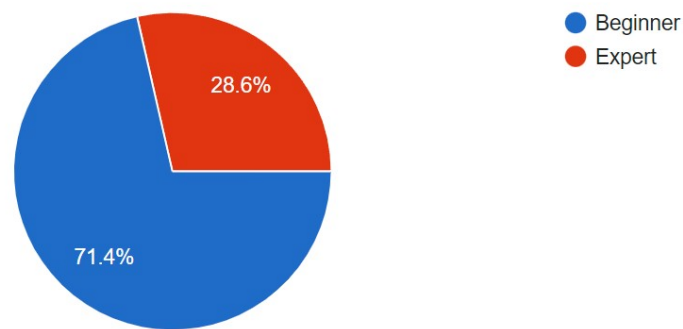


Figure 5.3: Survey results for the cyber security level of user

71.4% of my responses were from people who considered themselves beginners and 28.6% of my responses were from people who considered themselves experts. This is a good balance which means I can get a variety of opinions from people who do not understand the terms used and some that do.

## Section 2 - Design

How user friendly is this design?

7 responses

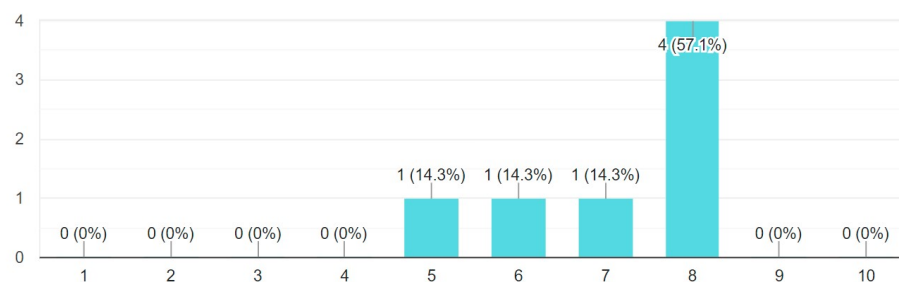


Figure 5.4: Survey results for how user-friendly the popup page is

A lot of users liked the design. They emphasised that the design was simplistic, clear and colourful. The colours helped 2 users navigate better but 1 thought there were too many bright colours used. I could consider making the colours darker but these were

bootstrap buttons that are widely used without issue. One user suggested that I could a more organic design as there were too many sharp edges. Another user suggested formatting the statistics table better which I could agree with. Another user suggested removing the texts from the buttons as I had icons which suggested what each button does, this is controversial as a lot of users may understand from the icons but this guarantees all of them do understand.

How much of this page do you understand?

7 responses

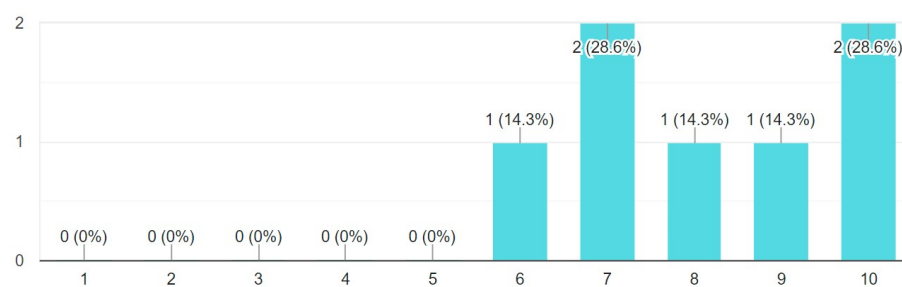


Figure 5.5: Survey results for how easy it is to understand the popup page

Users understood mostly everything on the popup page. 2 users mentioned that the terms used on the statistics box were not understood like "domain matches" and "invalid hyperlinks". In hindsight, these were badly named and could be renamed in a future update. Alternatively, I could add a small caption underneath or when the word is being hovered explaining the meanings.

How user friendly is this design?

7 responses

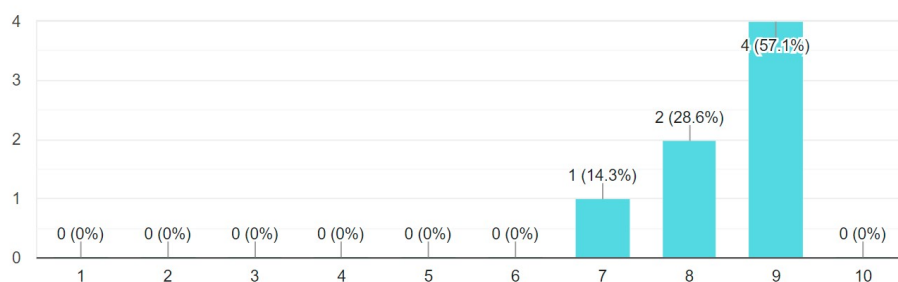


Figure 5.6: Survey results for how user-friendly the options page is

A lot of people thought the page was very user friendly, more than the popup. One

user liked its 'simplicity' and another liked that 'everything is very clear and all functionality is separated'. One opinion said the design was 'too simple', but 'this makes it easy to use and navigate'. This is ideal for my target audience, it should be very easy to use.

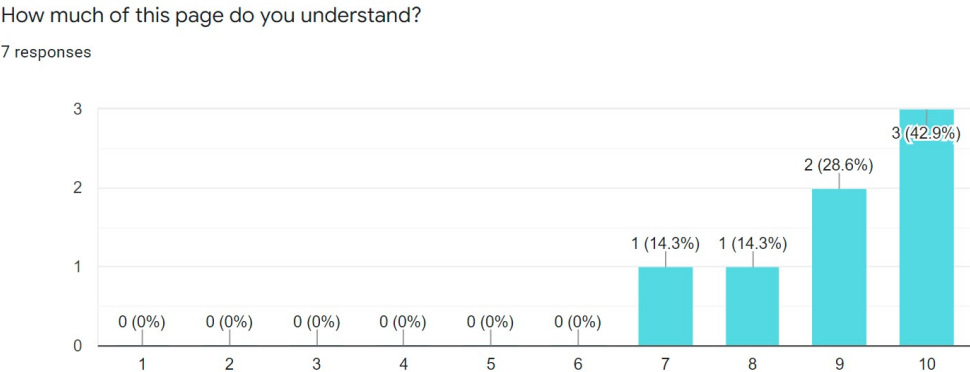


Figure 5.7: Survey results for how easy it is to understand the options page

A lot of people thought the page was easily understood and clear which is what I was aiming for. One user was unsure how the design changes when items were added to the whitelist, the blue box becomes longer and grows with the number of items, this is good until the list becomes too long, I may need to add a scroll bar into this box.

When providing final comments about the design one person mentioned that I could add more "round corners and a lighter colour similar to material design". Another user said they liked the drop-down menu I provided where it gives them further information, they suggested that the statistics table and buttons could be moved to a separate section as it feels a bit confusing. I agree with all these suggestions and is something I can work on in the next update.

### Section 3 - Functionality

Did the extension provide enough information to help you decide if the website was safe?

7 responses

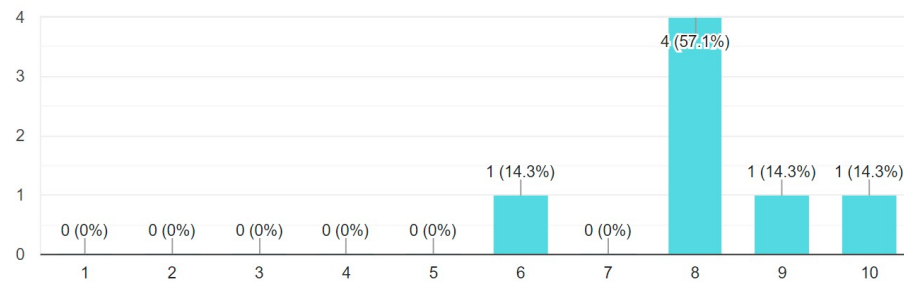


Figure 5.8: Survey results for how useful they found the extension

When asked which parts were useful, they found the rating, the popup notifications and the issues section were useful. The rating was mentioned 3 times which makes it clear that designing a good scoring algorithm is important.

When asked which parts were not useful two users found the network host listed in the statistics table was not useful. I will consider removing this or replacing it with a better statistic.

If you saw a security warning when you visited a website, did it provide useful information?

4 responses

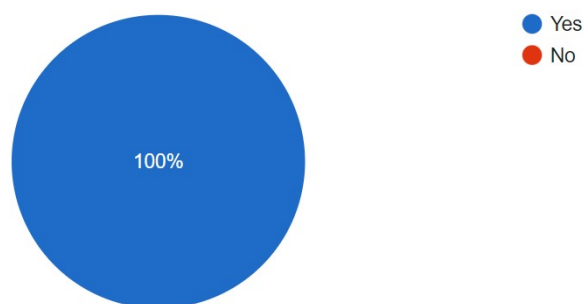


Figure 5.9: Survey results for how useful they found the security warning

All the users found that the security warning that appeared on a website was a useful

feature. I will continue to add information to this when a website is flagged for an issue.

One user found that the connection details and Trustpilot pages in the drop-down menu did not load for them. The WHOIS page did load and they found that very helpful. This is a bug that I could not recreate on my own device. I will need to do more in-depth testing on a range of devices on a range of browsers to debug this issue.

## Section 4 - Summary

How useful would you say the extension is?



7 responses

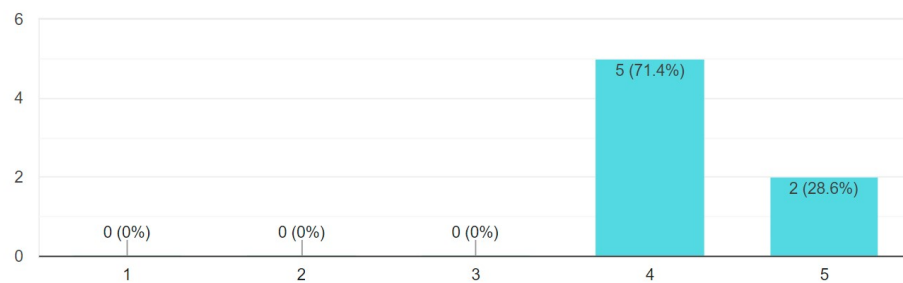


Figure 5.10: Survey results for how useful they found the extension

Most users found that my extension was highly useful which is fantastic to see. There were not many useful comments surrounding the information they learned. This is due to not having much time with the extension. I could repeat my survey after some time with the extension installed to get further insights into this.

How likely are you to recommend Security Surf to someone?

7 responses

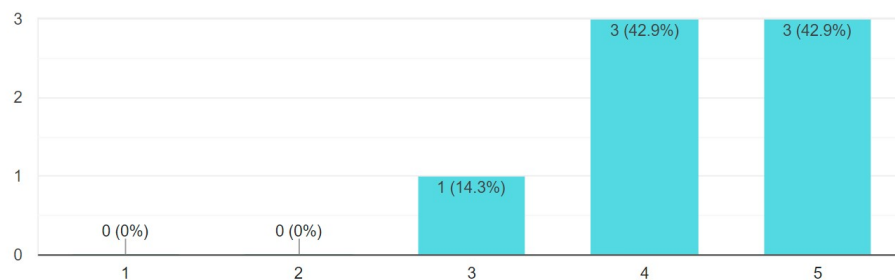


Figure 5.11: Survey results for recommending the extension

Most users would recommend my extension to someone. One comment said the

extension was "a great idea" and that it was useful for determining the integrity of websites that were less popular.



## Chapter 6

# Evaluation

My extension does a good job of detecting unsafe websites. It is a simplistic design for anyone to use including beginners. I have met most of my functional and non-functional requirements.

The functions that do not rely on other components are reliable, like the HTTPS check and the hyperlinks check. Functions that use APIs and produce iFrames are less reliable as they are sources of content. It is difficult to monitor these and ensure they do not have any downtime, it is also important that they are providing accurate responsive data.

Abstraction is used in my program as the users do not know all the features that go into checking each website but it calculates a score. Simple security features like checking SSL and hyperlinks are explained but other checks like using Google's API are not necessary for users to be aware of but it is useful to be there.

### 6.1 Improvements & Future Development

While my extension works well and helps users to a high standard there are still a lot of features that can be added. For example, a big limitation is that my extension only works on Chrome. The chrome browser makes up approximately 65% of the desktop browser market share worldwide.[statcounter, 2022] Safari and Firefox are used by a large number of people and none of them has access to my extension as it is limited to Chromium browsers. Browser Extensions by Mozilla is supported across multiple browsers including Firefox and Safari and usually requires few changes to get them supported fully.

There are more features that can be added to my extension. When uploading files to a website, it is important that the user trusts the website they are uploading their files to, this is why I would like to add a listener to detect when a user is uploading any type of file. When this is detected another safety check is run on the website for extra security.

If the user is subject to a scam through online browsing activity, they may want to try to work out where this cyber attack came from. To help with this I could store all the websites that scored low in hope that one of these in the history is the cause of the

attack and can be reported more easily and quickly.

Chrome's storage for sync has a limit of 512 items. If the user sets their TTL value as high, the storage could become full of the websites visited. To prevent this from becoming full, I could run a check that when the storage was nearly full, it would remove half of the oldest websites stored. A similar feature to this is when the extension has been installed for a year, it resets some of the options like the whitelist. Some of the websites on there may have changed a lot in a year so it is important that the user does not miss running these checks.

Currently on the options page, to select their TTL value, there are fixed options. I may add a feature where they could choose a custom time length (with maximum and minimum values). This may be an unnecessary detail to add this level of customisation but users have the option if they need it. More granular control could be given by adding custom TTL values for each website they visit. A website that they largely trust can be checked every 3 months but a website they partially trust can be checked every week. However again this much control may be unnecessary as the structure of websites changes minimally once setup.

An important feature I am missing is an installation page. This may really help beginners who do not understand technical terms. My extension has a lot of features and a small tutorial showing them the basics may help them to get full functionality from the extension.

When the user reports a website, the website to the National Cyber Security Centre opens to report a suspicious website. It may be helpful to automatically auto-fill the URL of the page they were on into the report. Depending on the country of the user, I could open a different site depending on the best place to report a website depending on the user's location.

The user could be given more control over the whitelist. Each subdomain is treated differently but this control can be given to the users. For domains which are consistently trusted, the user can choose to apply the whitelist to all subdomains of the domain or not.

I built my extension using HTML, JavaScript and CSS. A web development framework may have been better to use. It provides better tools, easier debugging, easier maintenance, higher security and better storage as the code length is smaller. It also allows a better front end to be built which gives them a better user experience overall.

### **6.1.1 Premium Tier**

All of the APIs I used were free. If there was a large number of users using the extension, it could be considered a good idea to create a premium tier. The money gives users access to higher quality APIs with extra features. Requests would respond quickly and the extension could handle a larger number of requests. It could also pay for more storage than Chrome storage offers. This could build a browsing history and overtime data analysis can be run off this data to help improve the extension and help the user learn more about their browsing habits.

## 6.2 Problems & Issues

One risk that users take when they choose a long TTL value in options is that if the website develops new vulnerabilities since they first visited it, the extension can not run any new checks on it. This is a risk that users take by choosing a value that long but it also means the user doesn't receive constant warning updates if these checks constantly happened.

### 6.2.1 Improving from user feedback

In section 5.2.1 about the results from user testing, there is lots to learn from. For the design, I needed to reformat the design of the statistics box and rename the terms inside it as they were confusing. It was also suggested to add more round corners and have a lighter colour scheme, this is something I could test with A/B testing.

For the functional side, I could review the scoring system. Users found the rating very useful for determining safety so reviewing this would help ensure I was accurately calculating a score. It was suggested to remove the network host from the statistics table, this was meaningless to the users so I could find more relevant information to replace this with.

I could repeat this survey in the future. This would be after I have spent some time doing more in-depth testing on a range of devices and browsers. This would help debug issues with the iFrames not appearing correctly. Repeating this survey after some time with the extension also gives me the chance to see what cyber security skills they had learnt from the extension.

## Chapter 7

# Conclusion

The Chrome extension I have created is available from the Chrome web store to install free of charge to anyone around the world who has a Google account and device which has the Chrome or a Chromium browser installed. It helps them to reduce the risk of a cyber attack by allowing them to understand the level of safety on the website. It is up to the user to decide whether to continue using the website depending on their purpose for it and the rating the extension has provided.

My extension allows users to make an informed decision. By helping users to understand this information, it will help prevent cyber attacks in their personal life and in businesses they work. My extension may change the way they navigate and use websites in the future.

My extension has been created to help people and businesses who lack cyber security awareness. It helps inform them of the level of risk of using a website they are trying to visit. If a website is rated poorly, when entering the website they will be presented with a warning with risks about the site. The extension provides lots of data about the website and does a good job of helping users to understand what all this data means.

### 7.1 Teaching

This extension was built to help teach others. Whilst it doing a good job at detecting unsafe websites, its also helping to teach along the way as it is displaying the issues. If the user does not understand what the terms mean, they can learn more about why it is unsafe.

Teaching beginners is useful to help spread awareness of cyber security. They are the most likely users that will be subject to a cyber attack so providing them with an interface that is friendly and simple is imperative. The aim while designing the extension was that it should not look complicated and that goal has been reached.

Experts of cyber security also benefit from this extension. It provides users an extra hand to summarise security information on the website to help them make a well-informed decision whether to trust the website. It is quicker to have this information to hand than to manually gather it.

## **7.2 Maintenance**

In the future I will monitor the number of users on a regular basis. This is so that I can quickly respond to any feedback on the Chrome Web Store. On the store people can also provide feedback, constructive feedback can be used to build extra features and fix bugs. Having the Established Publisher badge and the Featured badge displayed on my page will help to grow the number of users.

# Bibliography

- [Baeldung, 2022] Baeldung (2022). Creating a self-signed certificate with openssl. <https://www.baeldung.com/openssl-self-signed-cert>. Accessed 2022-02-22.
- [Bayuk and Horowitz, 2011] Bayuk, J. L. and Horowitz, B. M. (2011). An architectural systems engineering methodology for addressing cyber security. *Systems Engineering*, 14(3):294–304.
- [Centre, 2016] Centre, N. C. S. (2016). Understanding vulnerabilities. <https://www.ncsc.gov.uk/information/understanding-vulnerabilities>. Accessed 2021-12-30.
- [Cheah, 2019] Cheah, D. (2019). What is sql injection and xss? <https://medium.com/@tattwei46/what-is-sql-injection-and-xss-2a3f2e7ea0d>. Accessed 2022-02-11.
- [Combs, 1998] Combs, G. (1998). Wireshark. <https://www.wireshark.org/>. Accessed 2022-02-11.
- [Crespo-Santiago and Cosme, 2011] Crespo-Santiago, C. A. and Cosme, S. d. l. C. D. (2011). Waterfall method: a necessary tool for implementing library projects. *HETS Online Journal*, 1(2):86–99.
- [Developers, 2021] Developers, C. (2021). The activetab permission. <https://developer.chrome.com/docs/extensions/mv3/manifest/activeTab/>. Accessed 2022-02-04.
- [Developers, 2022] Developers, C. (2022). chrome.storage. <https://developer.chrome.com/docs/extensions/reference/storage/>. Accessed 2022-02-04.
- [Digicert, 2020] Digicert (2020). Check website security. <https://ssltools.digicert.com/checker/views/checkInstallation.jsp>. Accessed 2021-12-17.
- [EFF, 2010] EFF (2010). Htpps everywhere. <https://www.eff.org/https-everywhere>. Accessed 2021-12-28.
- [Gilger, 2022] Gilger, J. (2022). urlscan.io a sandbox for the web. <https://urlscan.io/about/>. Accessed 2022-04-26.

- [GlobalSign, 1996] GlobalSign, G. (1996). The dangers of self-signed ssl certificates. <https://www.globalsign.com/en/ssl-information-center/dangers-self-signed-certificates>. Accessed 2022-02-11.
- [Goel, 2021] Goel, V. (2021). An updated timeline for privacy sandbox milestones. <https://blog.google/products/chrome/updated-timeline-privacy-sandbox-milestones/>. Accessed 2022-04-27.
- [Gonella and Nericcio, 1996] Gonella, N. and Nericcio, L. (1996). Cybersecurity case library. *ABOUT THE CCI*, page 28.
- [Hinson, 2008] Hinson, G. (2008). Social engineering techniques, risks, and controls. *EDPAC: The EDP Audit, Control, and Security Newsletter*, 37(4-5):32–46.
- [ImmuniWeb, 2019a] ImmuniWeb (2019a). Scoring methodology. <https://www.immuniweb.com/websec/#scoring>. Accessed 2021-12-18.
- [ImmuniWeb, 2019b] ImmuniWeb (2019b). Website security test. <https://www.immuniweb.com/websec>. Accessed 2021-12-17.
- [ip api, 2022] ip api (2022). Ip geolocation api. <https://ip-api.com/>. Accessed 2022-04-25.
- [Kim, 2022] Kim, D. (2022). Find great extensions with new chrome web store badges. <https://blog.google/products/chrome/find-great-extensions-new-chrome-web-store-badges/>. Accessed 2022-04-20.
- [Krishna, 2017] Krishna, A. (2017). Complete guide on website penetration testing. <https://www.getastra.com/blog/security-audit/website-penetration-testing/>. Accessed 2021-12-30.
- [Morgan, 2020] Morgan, S. (2020). Cybercrime to cost the world \$10.5 trillion annually by 2025. <https://cybersecurityventures.com/hackerpocalypse-cybercrime-report-2016/>. Accessed 2021-10-20.
- [Muhlmann, 2007] Muhlmann, P. H. (2007). Trustpilot. <https://uk.trustpilot.com/>. Accessed 2022-02-11.
- [MYIPMS, 2012] MYIPMS (2012). Ip whois & flags chrome & websites rating. <https://chrome.google.com/webstore/detail/ip-whois-flags-chrome-web/kmdfbacgombndnllogoijhnggalgmkon>. Accessed 2021-12-28.
- [O'Donnell and Perry, 2013] O'Donnell, B. and Perry, R. (2013). Quantifying the economic value of chromebooks for k-12 education. *White Paper*.
- [Project, 2001] Project, O. W. A. S. (2001). Owasp top 10 - 2021. <https://owasp.org/Top10/>. Accessed 2021-12-30.
- [rolan...@gmail.com, 2022] rolan...@gmail.com (2022). Issue 628819: Extend webrequests oncompleted callback to include information about tls connections. <https://bugs.chromium.org/p/chromium/issues/detail?id=628819>. Accessed 2022-04-21.

- [Safa et al., 2016] Safa, N. S., Von Solms, R., and Fletcher, L. (2016). Human aspects of information security in organisations. *Computer Fraud & Security*, 2016(2):15–18.
- [Safebrowsing, 2022] Safebrowsing, G. (2022). Safe browsing lookup api (v4). <https://developers.google.com/safe-browsing/v4/lookup-api>. Accessed 2022-04-24.
- [Schuh, 2020] Schuh, J. (2020). Building a more private web: A path towards making third party cookies obsolete. <https://blog.chromium.org/2020/01/building-more-private-web-path-towards.html>. Accessed 2022-04-27.
- [statcounter, 2022] statcounter (2022). <https://gs.statcounter.com/browser-market-share/desktop/worldwide>. Accessed 2022-04-29.
- [Sucuri, 2010] Sucuri (2010). Free website security check & malware scanner. <https://sitecheck.sucuri.net/>. Accessed 2021-12-18.
- [sullo, 2001] sullo (2001). Nikto2. <https://cirt.net/Nikto2>. Accessed 2021-12-30.
- [synk, 2015] synk (2015). Website vulnerability scanner. <https://snyk.io/website-scanner/>. Accessed 2021-12-21.
- [Tools, 2013] Tools, P. (2013). Website vulnerability scanner. <https://pentest-tools.com/website-vulnerability-scanning/website-scanner>. Accessed 2021-12-18.
- [Trust, 2007] Trust, W. O. (2007). Website security & browsing protection. <https://chrome.google.com/webstore/detail/wot-website-security-brow/bhmmomiinigofkjcapegjndpbikblnp?hl=en-GB>. Accessed 2021-12-28.
- [Wang et al., 2009] Wang, C.-C., Chen, C.-A., and Jiang, J.-C. (2009). The impact of knowledge and trust on e-consumers' online shopping activities: An empirical study. *J. Comput.*, 4(1):11–18.
- [Zimmermann and Renaud, 2019] Zimmermann, V. and Renaud, K. (2019). Moving from a 'human-as-problem' to a 'human-as-solution' cybersecurity mindset. *International Journal of Human-Computer Studies*, 131:169–187.



# **Appendix A**

## **Installation**

### **A.1 Using Chrome Web Store**

1. User needs a device with Google Chrome browser or a browser built on Chromium, this includes Brave, Opera and Microsoft Edge.
2. User needs to be signed into Google account.
3. Visit the link below:  
<https://chrome.google.com/webstore/detail/security-surf/ekhcklnpcdailhganpmoolfoojgnlii>
4. Click 'Add to Chrome' to install

### **A.2 Manual Installation**

1. User needs a device with Google Chrome browser or a browser built on Chromium, this includes Brave, Opera and Microsoft Edge.
2. User needs to be signed into Google account.
3. The Security Surf (V1.15) extension must be stored locally in an unzipped folder
4. Visit 'chrome://extensions' or click the Chrome menu, hovering over 'More tools' then selecting 'Extensions'
5. Enable developer mode by clicking the toggle switch next to 'Developer mode'.
6. Click the 'Load unpacked' button and select the extension directory to install

## **Appendix B**

# **Survey Questions**

### **B.1 Introduction**

- I confirm I have installed the extension
- What is your age?
  - Under 18
  - 18-25
  - 26-35
  - 36-45
  - 46-55
  - 56-65
  - Over 65
- How would you rate your cyber security skills?
  - Beginner
  - Expert

## B.2 Design

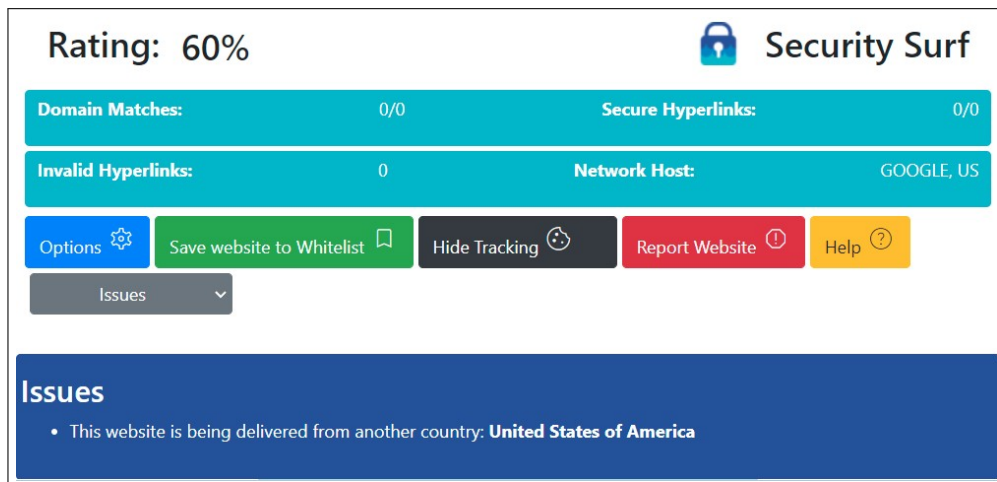
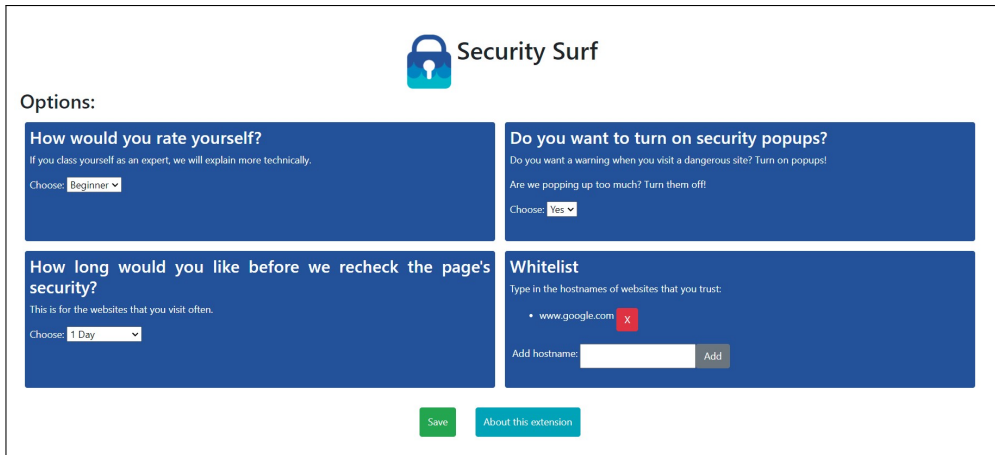


Figure B.1: Popup image for survey

- How user friendly is this design?
  - Scale from 1 to 10
  - 1=Terrible
  - 10=Perfect
- What do you like and dislike about this design?
- How much of this page do you understand?
  - Scale from 1 to 10
  - 1=Don't understand anything on this page
  - 10=Understand everything on this page
- Which parts are clear and which parts are confusing?



Options:

**How would you rate yourself?**  
If you class yourself as an expert, we will explain more technically.  
Choose:

**Do you want to turn on security popups?**  
Do you want a warning when you visit a dangerous site? Turn on popups!  
Are we popping up too much? Turn them off!  
Choose:

**How long would you like before we recheck the page's security?**  
This is for the websites that you visit often.  
Choose:

**Whitelist**  
Type in the hostnames of websites that you trust:  
• www.google.com   
Add hostname:

Figure B.2: Options image for survey

- How user friendly is this design?
  - Scale from 1 to 10
  - 1=Terrible
  - 10=Perfect
- What do you like and dislike about this design?
- How much of this page do you understand?
  - Scale from 1 to 10
  - 1=Don't understand anything on this page
  - 10=Understand everything on this page
- Which parts are clear and which parts are confusing?
- Please provide any feedback on the design that you like or can be improved

### B.3 Functionality

For this section please visit any online website you often use.

- Did the extension provide enough information to help you decide if the website was safe?
  - Scale from 1 to 10
  - 1=Do not understand any of the information provided
  - 10=I understood all of the information provided
- Which parts were useful to help you decide whether the website is safe?

- Which parts were not useful to help you decide whether the website is safe?
- If you saw a security warning when you visited a website, did it provide useful information? Leave blank if you did not experience this warning.
  - Yes
  - No
  - Other:
- Please provide any feedback on the functions of the extension you like or can be improved

## **B.4 Summary**

- How useful would you say the extension is?
  - Scale from 1 to 5
  - 1=Not at all
  - 10=Very Useful
- Learning: Has this extension taught you anything about website security? If yes, what?
- How likely are you to recommend Security Surf to someone?
  - Scale from 1 to 5
  - 1=Not Likely
  - 10=Likely
- Do you have any last comments about the Chrome extension as a whole?