



Investigating Web Tracking Technologies and Techniques to Measure Online Privacy

Student Name: Brian Shanahan

Student ID: 17218829

Supervisor: Thomas Welsh

Course: B.Sc. Computer Systems (LM051)

Academic Year: 2021/2022

Table of Contents

1.0 Introduction.....	5
1.1 Background	5
1.2 Project Aim and Objectives.....	5
1.3 Project Schedule.....	6
1.3.1 Gantt Chart	6
2.0 Literature Review.....	8
2.1 What is Web Tracking?.....	8
2.2 Web Tracking Techniques	8
2.3 Stateful Tracking Techniques.....	9
2.3.1 HTTP Cookies	9
2.3.2 Flash Cookies	10
2.3.3 HTML5 Local Storage.....	10
2.3.4 Evercookies	10
2.4 Stateless Tracking Techniques	10
2.4.1 Fingerprinting	10
2.4.2 Network and Location Fingerprinting	12
2.4.3 Device Fingerprinting.....	12
2.4.4 Canvas Printing.....	12
2.5 Online Privacy Protection Techniques	12
2.6 Privacy Tools.....	13
2.6.1 VPNs.....	13
2.6.2 Browser Plug-Ins/ Extensions	13
2.6.3 Private Browsers.....	14
2.7 Conclusion.....	14
3.0 Research/Analysis.....	15
3.1 Possible Approaches	15
3.1.1 Browser Plug-In/Extension.....	15
3.1.2 Windows Application	15
3.1.3 Analyzing HTML files	16
3.1.4 Investigate websites with existing tools	18
3.2 Privacy Browser Extension	19
3.3 Existing Privacy Browser Extensions	20
3.3.1 Privacy Badger	20
3.3.2 uBlock Origin	21

3.3.3 Blacklist and Algorithm Methods.....	22
3.4 Languages Used In Development	22
3.4.1 HTML (HyperText Markup Language)	22
3.4.2 CSS (Cascading Style Sheets)	23
3.4.3 JavaScript.....	24
3.5 Development Tools/Environments.....	24
3.5.1 Visual Studio Code.....	24
4.0 Privacy Tool Development	25
4.1 Initial Plan	25
4.2 Development Process	28
4.3 Software Testing	32
5.0 Evaluation / Conclusion.....	36
6.0 References.....	37

Figure 1 - Project Gantt Chart.....	6
Figure 2 - How HTTP Cookies work.....	9
Figure 3 - Forming a unique fingerprint	11
Figure 4 - Fingerprint being recognised across multiple sites	11
Figure 5 - How a VPN works	13
Figure 6 - Chrome Browser Extension	14
Figure 7 - Tor Private Browser Logo Figure 8 - Chrome Incognito Logo	14
Figure 9 - The Cookies used by the University of Limerick website	16
Figure 10 - Chrome DevTool Cookie Information	17
Figure 11 - Chrome DevTool Cookie Information pt.2	17
Figure 12 - OWASP Zed Attack Proxy	18
Figure 13 – Cookieserve Cookie results	19
Figure 14 - Cookieserve Cookie results pt.2.....	19
Figure 15 - Privacy Badger Logo.....	20
Figure 16 - Privacy Badger UI.....	20
Figure 17 - uBlock Origin Logo	21
Figure 18 - uBlock Origin UI.....	21
Figure 19 - Example of HTML.....	23
Figure 20 - Example of CSS	23
Figure 21 - Example of JavaScript.....	24
Figure 22 - Visual Studio Code	25
Figure 23 - Basic Diagram for Privacy Tool	26
Figure 24 – Original Browser Extension UI Mock-Up	27
Figure 25 - Windows Application UI Mock-up.....	28
Figure 26 - Tkinter GUI Code Example	28
Figure 27 - Tkinter GUI Code Example 2	29
Figure 28 - View All Cookies window	29
Figure 29 - View All Cookies window w/ Message pop-up.....	30
Figure 30 - Domain Blocklist window.....	30
Figure 31 - Domain Blocklist window w/ Message pop-up	31
Figure 32 - Search for Trackers window	31
Figure 33 - All Cookies Table	32
Figure 34 - Search for Cookies Table	32
Figure 35 - Blocklist Table	32
Figure 36 - Menu Buttons	33
Figure 37 - Table Buttons	33
Figure 38 - Software Testing Table	35

1.0 Introduction

1.1 Background

Over the past couple of years, people's awareness of their online privacy has become more and more prominent. It has become apparent to people that the need for privacy measures to protect one's privacy is starting to look increasingly vital. Every day, people are tracked through search engines and browsers, as well as through social media and other various websites. Substantial amounts of data is tracked and collected from our online activity by large companies such as Google, Amazon, and Facebook to name a few. A plethora of our personal information is tracked from this activity including personal information, search history, websites visited, etc. This information can be used in a wide variety of ways such as analytics and advertising. However, it is possible that these companies and even some undesirable entities may exploit our data in ways that erode our privacy.

This project will critically compare the various techniques used to track people online and look at why data is tracked and collected for use elsewhere. This project will consider the benefits that tracking can provide to individuals, as well as the detrimental effects tracking can have on personal privacy. It will also examine the possible ways we can measure privacy and improve our overall privacy health by looking at various online protection methods and how they help us safely browse the Web. This project will also include the development of a privacy tool that monitors privacy health while browsing online.

1.2 Project Aim and Objectives

- Research the various techniques used by websites to track people.
- Research the possible ways of measuring privacy on websites.
- Research privacy protection techniques (e.g., Extensions, VPNs, etc.) and how they can help improve your privacy.
- Propose and develop a tool that allows you to monitor privacy health while browsing online.

1.3 Project Schedule

In this section I break down the overall plan and schedule for my final year project. I created a Gantt Chart to give a more visual representation of the planned work for the year ahead and to give myself milestones to reach every few months. I also outline the various deadlines and project deliverables that are to be followed in the project timetable.

1.3.1 Gantt Chart

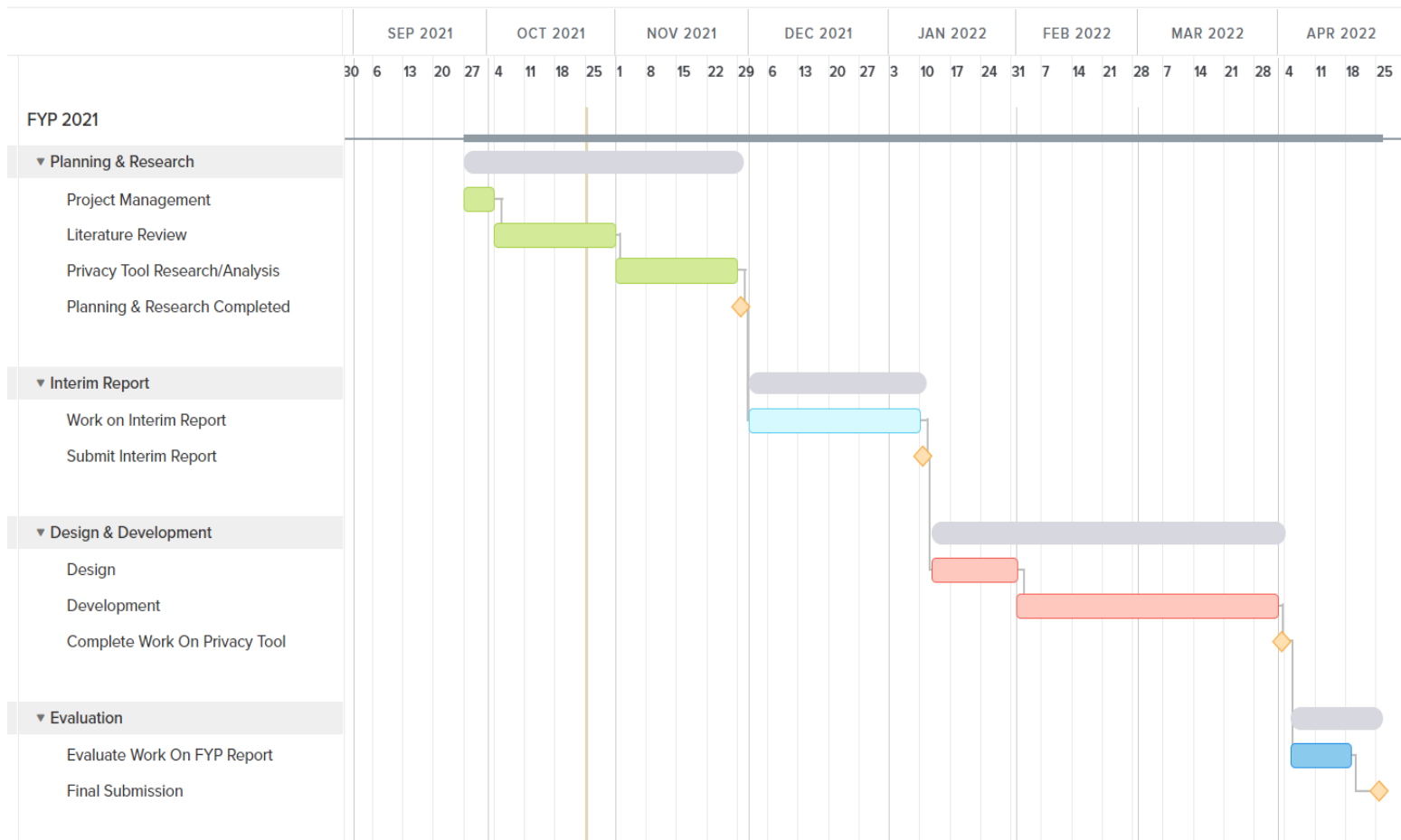


Figure 1 - Project Gantt Chart

1.3.2 Project Timetable

Start Date	Task	Objectives met	Deadline
Week 4	Project Management	Plan out project schedule and deadlines	Week 5
Week 5	Literature Review	Examine various research papers and literature	Week 8
Week 9	Presentation	Present research work done to date	Week 9
Week 9	Analysis	Research different types of privacy tools and how to develop one	Week 12
Week 12	Deadline for Deadline Agreement Form + Deadline for Supervisor Marking Scheme Form	Submit both forms	25 th November
Christmas Break	Interim Report	Write the Interim Report for FYP	10 th January
January	Design	Design Privacy Tool	February
February	Development	Develop Privacy Tool	March
April	Evaluation	Evaluate FYP	April
	Submit Draft Report	Submit Draft Report	31 st March
	FYP Showcase	FYP Showcase	Week 11 (TBC)
	Final Submission Date for Product	Submit FYP Product	19 th April
	Final Submission Date for FYP Report	Submit FYP Report	25 th April

2.0 Literature Review

My literature review will focus on papers that revolve around the various Web Tracking Technologies that are widely used today. It will also provide insight into what Web Tracking is and explore the reasons why it is utilised so much on the web. This literature review will also look at the reasons why online privacy is of such vital importance in this day and age as well as how one could measure their privacy and protect themselves while browsing online.

2.1 What is Web Tracking?

In definition, Web Tracking can be said to be the practice through which a website collects and monitors the information and activities of the site's visitors (Crawford, 2020).

This widespread technique is used to fuel a wide variety of online services. The data gathered from users browsing sites online is utilized in areas such as online advertising and site analytics. These services can be extremely beneficial to web economy and even to web users as it can improve their online experience (Sanchez-Rola, et al., 2017). Many websites collect a wide range of user data, for example: language, login information, shopping cart contents and other user preferences. In the case of online advertisements, tracking provides users with more relevant and helpful ads. All this helps to enhance and personalize user experience on the internet (Internet Health Report, 2018).

While web tracking has proven to be quite advantageous and instrumental to web economy and website users, it also raises a lot of questions about one's personal privacy and it's possible violation. The various techniques used to retrieve data from users is often collected without their knowledge or consent. This can be viewed by many web users as a major breach of privacy (Sanchez-Rola, et al., 2017).

There are a variety of methods employed to track users on the web from both first-party and third-party sources. First-party tracking refers to tracking done by the websites that a user chose to visit while browsing online (Crawford, 2020). Third-party tracking can be defined as tracking done by other entities separate from the ones a user has visited. This type of tracking is viewed as especially pervasive, as it has the ability to track large amounts of data across multiple websites visited by web users. (Bujlow, et al., 2017)

2.2 Web Tracking Techniques

The various technologies used in web tracking can generally be categorized into two types: Stateful tracking and Stateless tracking. Stateful tracking can be defined as techniques that store information on the client side, so the method used for tracking is kept on a web users computer and is then later retrieved (Bielova, 2017). Examples of Stateful tracking include HTTP Cookies, Flash Cookies, HTML5 Local Storage and Evercookies (Kim, 2014).

Stateless tracking refers to tracking technologies that do not require any information to be stored on the client side. The Stateless tracking techniques used today are especially pervasive as they are considered far more difficult to block or limit due to their ability to

persist even after removing cookies or when using a private browsing mode. (Sanchez-Rola, et al., 2017). Examples of Stateless tracking are the various forms of Fingerprinting such as Network and Location Fingerprinting, Device Fingerprinting and Canvas Fingerprinting. (Bujlow, et al., 2017).

2.3 Stateful Tracking Techniques

2.3.1 HTTP Cookies

Perhaps the most recognizable and widely known method of web tracking is HTTP Cookies. Defined by Peters and Sikorski (1997), HTTP Cookies “are small data structures sent from a web server to your browser and saved on your hard drive in a text file. They are nothing more than a string of characters (letters and numbers) that store certain pieces of information about you.” The Cookie was invented in 1994 by Lou Montulli, an engineer at Netscape. The original intent for HTTP Cookies was to provide web users with an enhanced experience by allowing users to pick up where they left off when returning to a website they had previously been on (Kristol, 2001). An example of this can be seen in eCommerce sites, when a user returns to the site to find the items they had placed in their shopping cart from an earlier visit.

HTTP (Hyper Text Transfer Protocol) is a communications protocol used to connect to servers on the web and establishes a connection between the server and client. HTTP works by sending requests made by the client, to the server and returning web page information vice versa. However, the server itself cannot retain any information about the client after this transaction. But with the addition of Cookies, the HTTP header can receive information from the client’s side. Cookies stored on a client’s computer assign a unique ID to the client, which the server will recognize in future. This allows websites to recall information about users who have visited in the past. HTTP Cookies expire either at the end of a session, or after a period of time specified by the website that set the cookie (Sipior, et al., 2011).

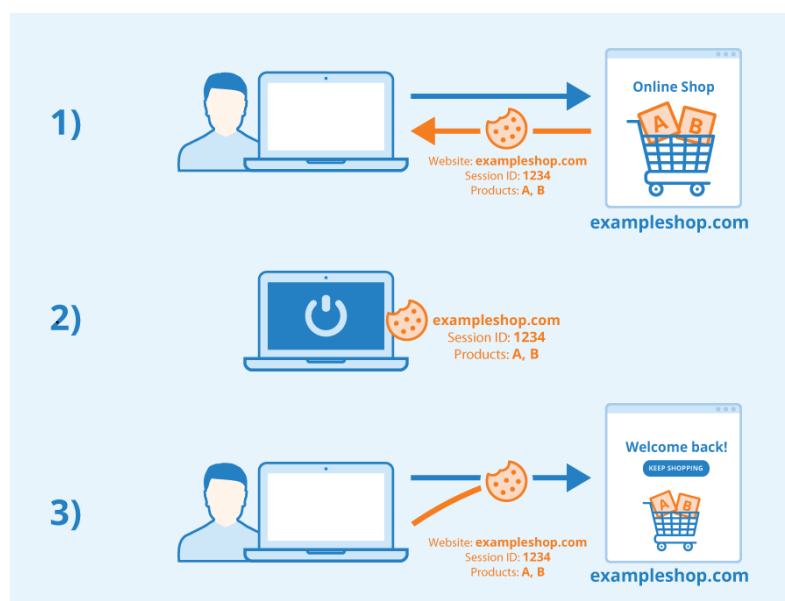


Figure 2 - How HTTP Cookies work

2.3.2 Flash Cookies

Although functionally similar to HTTP Cookies, Flash Cookies have a few key differences. Flash Cookies can contain up to 100KB of information opposed to 4KB with HTTP Cookies. They are also stored differently to HTTP Cookies and they are not controlled by the browser. Because of this, they do not have expiration dates unless specified. Flash Cookies are far more persistent as they are not controlled by the browser, so they cannot be deleted as easily as HTTP Cookies. These types of cookies can bypass private browsing modes and they are even accessible across all browsers installed in a system (Soltani, et al., 2009). Flash Cookies were created by an online advertising company called United Virtualities (UV). UV first developed the Persistent Identification Element (PIE) as a backup ID system for cookies that were deleted by users. PIE utilizes the Adobe Flash feature called ‘local shared objects’, resulting in the Flash Cookies we see today (Sipior, et al., 2011).

2.3.3 HTML5 Local Storage

HTML5 Local Storage is quite similar to HTTP Cookies in how it utilizes the same communications protocol to store and retrieve information in the browser. A component of the HTML5 Web Storage API, Local Storage has some notable advantages including a storage capacity of up to 5MB. By default, objects stored in Local Storage do not have expiration dates, making them more resilient than HTTP Cookies. However, they are easier to remove compared to Flash Cookies, as they are stored in the browser (Bujlow, et al., 2017).

2.3.4 Evercookies

The Evercookie (also known as a Supercookie) is a Javascript API developed by Samy Kamkar in 2010. Evercookies are considered the most resilient form of Cookies as they can be stored in multiple storage areas simultaneously (Schmidt, 2020). A single Evercookie can utilize several storage mechanisms such as HTTP and Flash Cookies as well as any of the HTML5 Web Storage options including Local Storage, and many more. An Evercookie can respawn itself even after deletion (Bujlow, et al., 2017). When it was released in 2010, the Evercookie could utilize thirteen storage mechanisms. As long as one cookie remains after deletion in other storage areas, it will still be able to restore itself. For example, if an Evercookie is deleted from HTML5 Local Storage but still exists as a Flash Cookie, it can check for the area from which it was deleted, and respawn itself there (Kamkar, 2010).

2.4 Stateless Tracking Techniques

2.4.1 Fingerprinting

Fingerprinting is done via a wide range of techniques that can identify and track a web user, thus forming a unique ‘fingerprint’ that can be recognized by various web services. This unique fingerprint can include information such as the user’s IP address, network provider, device ID and even more obscure information like the user’s operating system, screen resolution, browser version, system fonts, language settings and many more. All these details can be put together to form a profile, resulting in a unique ID and making it easier for websites and other services to identify a user. All this is possible without the need for storing and retrieving information on the user’s device (Bujlow, et al., 2017).

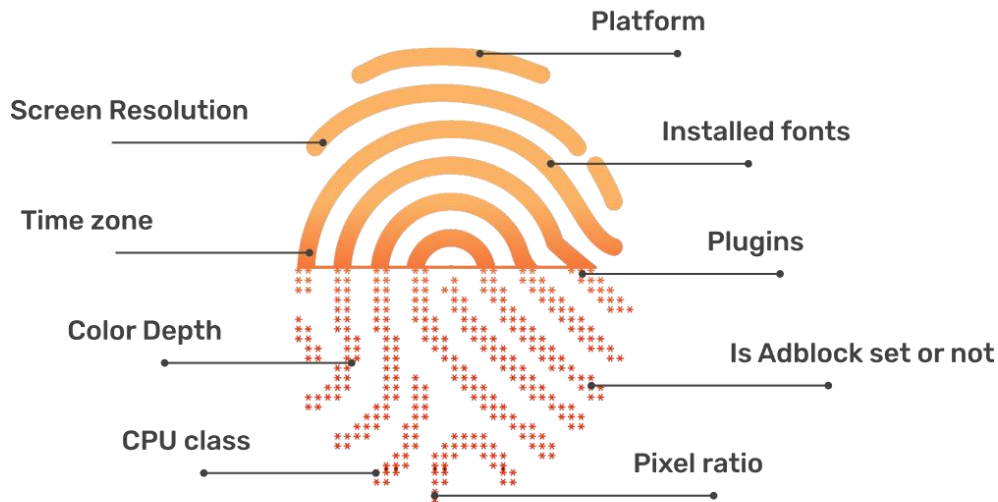


Figure 3 - Forming a unique fingerprint

Fingerprinting is considered the most pervasive web tracking technique due to its stateless nature and the only way to stop this type of tracking is to stop using the internet. While there are methods to make your online profile less unique, there currently is no way to prevent fingerprinting. Fingerprinting is used by advertising companies to build profiles on web users to provide more relevant ads but it can also be used by fraud detection services. For example, banks can use fingerprinting to identify web users who log on to their internet banking services. This allows them to stop identity thieves. While an identity thief may have a user's credentials, they will not have the same fingerprint as that user and as a result they will be blocked from logging in (Joseph, 2020).

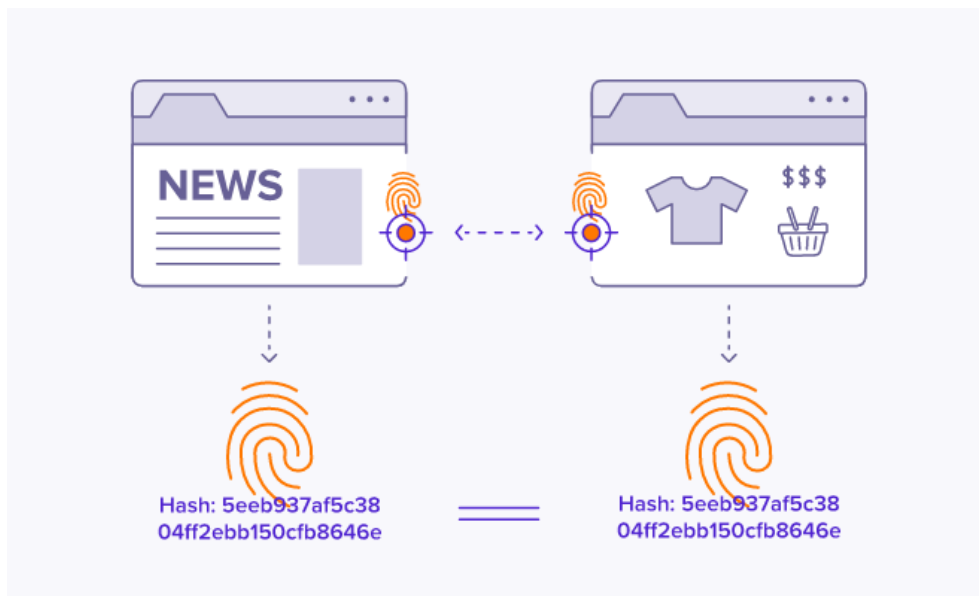


Figure 4 - Fingerprint being recognised across multiple sites

Listed below are some of the most popular methods employed to form a unique fingerprint on a web user (Bujlow, et al., 2017):

2.4.2 Network and Location Fingerprinting

Web services can identify a user by using network tools to examine incoming HTTP requests. Information such as the internet service provider and IP address of the user can be identified from this method.

2.4.3 Device Fingerprinting

This form of fingerprinting uses JavaScript to determine a range of user information to distinguish devices on the web and associate them to users. The user's operating system, screen resolution, list of system fonts, time zone and other system settings are examined to form a unique profile.

2.4.4 Canvas Printing

With the use of the HTML5 `<canvas>` element, a pixel can be placed on an area of the user's screen which can then be used to generate a fingerprint. In order to do this a method called 'ToDataURL' is used to extract a unique hash of a Base64 encrypted string that encodes the content of the `<canvas>` element. Information extracted from this method include the user's operating system, browser version, system fonts, graphics card, anti-aliasing settings and more.

2.5 Online Privacy Protection Techniques

With online tracking techniques becoming more and more pervasive, the need for protection while browsing online has become crucial. Luckily, it is possible to eliminate the vast majority of tracking techniques and at the very least, mitigate the effectiveness of the most pervasive of these techniques.

Stateful tracking techniques such as HTTP Cookies, are relatively easy to stop. The use of browser settings can allow for the deletion of cookies from a system. Browsers such as Google Chrome, Microsoft Edge, etc. also allow web users to completely block any first-party or third-party cookies if they so wish. This can however result in a worse experience for web users as it can result in them being logged out of accounts and in some cases it can even cause web pages to break. For the more pervasive techniques such as Flash Cookies or Evercookies, browser plugins and extensions can help eradicate these tracking methods from a web users system.

Stateless tracking techniques such as Fingerprinting and its many forms are far trickier to protect from. While it's not possible to completely block Fingerprinting, it is possible to reduce the effectiveness of it. The use of VPNs, Browser Plug-Ins/Extensions and Private Browsers can help web users browse with greater privacy and are especially helpful against Network and Location Fingerprinting and Device Fingerprinting. Browser Plug-Ins/Extensions can also help against Canvas Fingerprinting as they can detect tracking pixels (Bujlow, et al., 2017).

2.6 Privacy Tools

While there are many ways for websites and other services to track people across the web, there is an increasing number of ways to protect online privacy. Tools such as VPNs, Browser Plug-Ins, Private Browsers, and more, allow web users to improve their online privacy health. Listed below are some of the tools that can be used to better protect our online privacy.

2.6.1 VPNs

A Virtual Private Network helps users to browse the internet anonymously. This works by creating a tunnel that funnels through all of a user's internet traffic, allowing this traffic to become hidden (Whittaker, 2018). VPNs mask the user's IP address such that the activities of a user cannot be traced, unlike when using a public network where the IP address is traceable (Rafter, 2021). Beyond a VPN's ability to stop entities from tracking user location via IP addresses, they can't actually prevent tracking techniques such as Cookies and Fingerprinting. But thanks to its anonymous feature set, it can stop websites and other services from profiling users effectively (VPN Accounts, 2020). For example, web services that use fingerprinting and look at a user's IP address to create a profile, will end up receiving false information thanks to VPN masking the user's IP address.

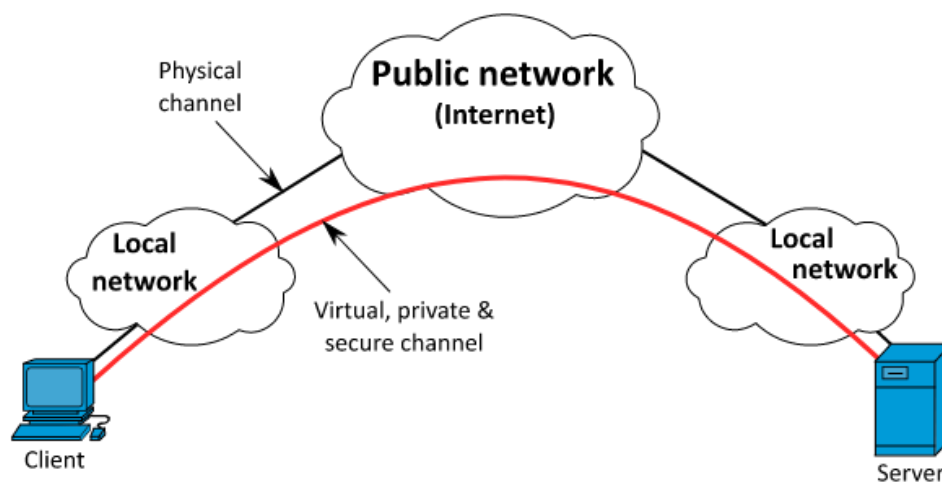


Figure 5 - How a VPN works

2.6.2 Browser Plug-Ins/ Extensions

Browser Plug-Ins/ Extensions are extremely helpful in eliminating most Stateful Tracking Techniques such as HTTP Cookies, Flash Cookies and Evercookies. They can also detect tracking pixels on web pages that utilize Canvas Fingerprinting. Browser Plug-Ins/ Extensions like Privacy Badger are programmed to identify tracking heuristics which indicate that an entity is attempting to track a web user (Arrieta, et al., 2020). There are two main methods used by Browser Plug-Ins/ Extensions to block tracking techniques. One method is the blacklist method. The blacklist is used to identify the most common third-party trackers and when a blacklisted domain or URL is encountered while a web user is browsing online, it will be blocked by the Browser Plug-In/ Extension. Another method used to block tracking

techniques is through the use of algorithms. This method looks at the heuristics of various tracking techniques and examines their behaviour. If an entity is found to be tracking a web user across multiple sites, the algorithm will determine that the web user is being tracked and block the tracker (Nibbeling, 2019).



Figure 6 - Chrome Browser Extension

2.6.3 Private Browsers

Private browsers can be very helpful as they can block most Stateful tracking techniques and can help against some Fingerprinting methods such as Network and Location Fingerprinting and Browser Fingerprinting. Examples of private browsers are Tor and Brave. While popular browsers like Chrome offer private browsing modes such as Incognito mode, they aren't quite as effective as the dedicated private browsers. Private browsers like Tor work by utilising a network of tunnels to make browsing more private and secure for web users. Essentially, it is a proxy that can hide a web users IP address and browsing history from online trackers. It uses what is called 'Onion Routing' to bounce a web users traffic across a network of three relays, which encrypts the traffic and masks the IP address of a web user.

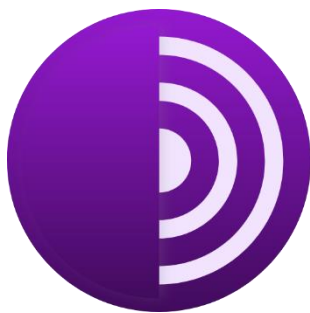


Figure 7 - Tor Private Browser Logo



Figure 8 - Chrome Incognito Logo

2.7 Conclusion

The purpose of this literature review was to learn about the various tracking techniques that are utilized by web services today, as well as gain a better understanding of how web users can improve their online privacy health while they browse online. Web tracking is seen by many to be nothing but a detriment to people's privacy, however it has proven itself to be beneficial too. This can be seen in areas such as online fraud detection and web analytics, and it can result in a far more positive web experience for users (e.g. remembering login information, shopping cart, etc.).

For combatting the more pervasive side of web tracking, there are many options available to web users to better protect their privacy while browsing online. Privacy tools such as VPNs, browser plug-ins/extensions and private browsers all help to improve privacy health for web users. However, during my research on these privacy tools I find their to be a lack of free tools that strike a balance between ease of use and overall effectiveness. More free tools are needed for protecting online privacy that are simple to use and accessible while still being just as effective at blocking or limiting tracking as some of the more complex tools out there today.

3.0 Research/Analysis

3.1 Possible Approaches

For my FYP I propose the design and development of an online privacy tool that monitors privacy health. For this section of the report, I will examine some possible approaches to the developing this privacy tool. I will also consider some other approaches which would involve investigating tracking techniques and analyzing how they work.

3.1.1 Browser Plug-In/Extension

As previously stated in my literature review, a privacy browser plug-in/extension is a useful tool that helps eliminate the vast majority of tracking methods and can minimize the effectiveness of the more pervasive techniques like stateless tracking. The terms ‘Plug-In’ and ‘Extension’ are used interchangeably across the web, however, after a lot more research, I found that there are a few key differences between the two.

Plug-Ins refer to third party libraries that are embedded within a website. These plug-ins can be used to add a wide variety of functionalities and features to a specific web page. However these added functionalities do not extend to other web pages or to the browser, as they only affect the page in which they are embedded. Well known examples of plug-ins are Flash, Silverlight and Apple QuickTime (Richard, 2010). Some plug-ins related to privacy would be Complianz and Secure Privacy, which give websites functionalities such as cookie notices and makes them compliant with GDPR and other lawful requirements.

Extensions on the other hand affect the browser and can add new functionalities as well as change existing ones. Extensions are essentially add-ons for the browser and can change the both browser and the websites a web user visits. Popular examples of extensions include dark theme mode and the picture-in-picture extension (Richard, 2010). There are quite a few extensions available today that help protect the privacy of web users. Browser extensions such as Privacy Badger, Ghostery, Adblock and UBlock Origin to name a few, are extremely helpful in boosting privacy health.

3.1.2 Windows Application

As an alternative to the browser extension, but within the same vein and general approach, a Windows application that works alongside the browser is another option for developing a privacy tool. This method would involve an application to work seamlessly with the browser and be allowed access to the trackers found on the system. This approach would be functionally identical to the browser extension however, it could require extra workarounds

due to the application being a separate entity from the browser. For example, certain permissions may need to be given to the application by the browser for it to function correctly.

3.1.3 Analyzing HTML files

One method of analyzing HTML files is through Chrome DevTools. This allows anyone to inspect a web page and analyze it more closely. The structure of the HTML code can be viewed as well as the CSS stylings and Javascript used. Sources, network and application data can also be inspected. With these developer tools, it is possible to view the cookies being used by a certain domain. Below is an example of how cookies can be inspected through developer tools.

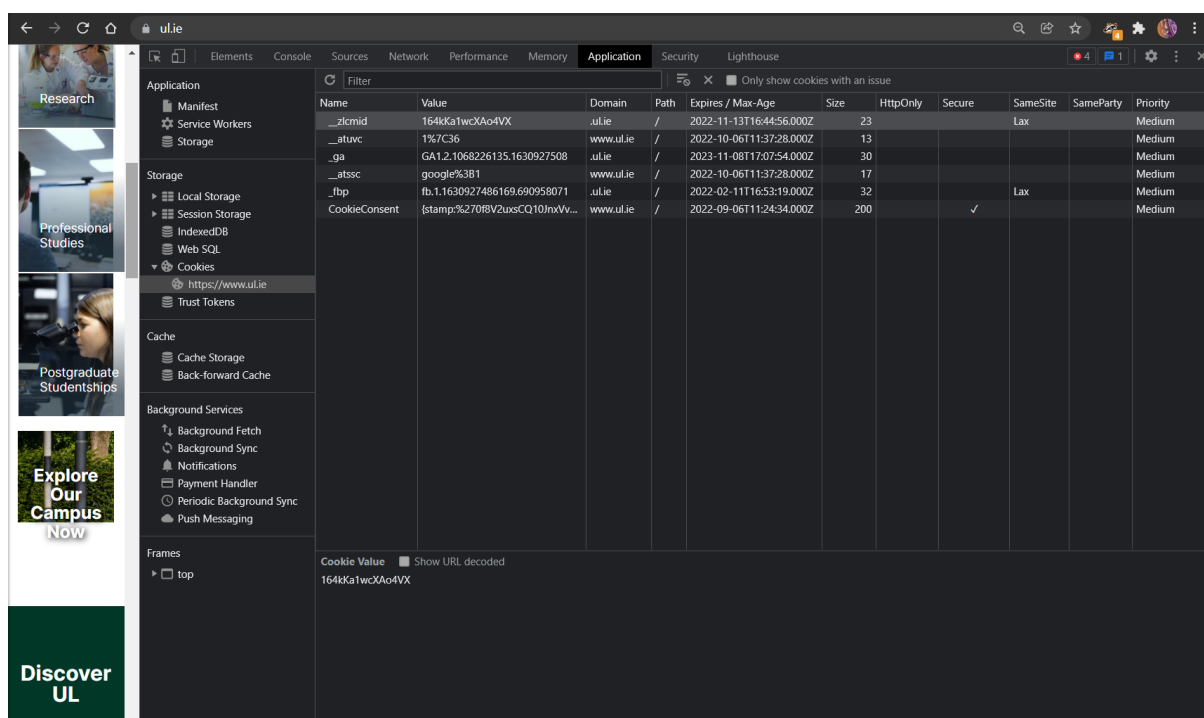


Figure 9 - The Cookies used by the University of Limerick website

A range of information on cookies can be analyzed through this method. Data such as the name, the expiry date and the size of a cookie can be examined as well as a range of other information. This data is sorted into a table within the developer tools for easy reading.

Name	Value	Domain	Path	Expires / Max-Age
__zlcmid	164kKa1wcXAo4VX	.ul.ie	/	2022-11-13T16:44:56.000Z
__atuv	1%7C36	www.ul.ie	/	2022-10-06T11:37:28.000Z
_ga	GA1.2.1068226135.1630927508	.ul.ie	/	2023-11-08T17:07:54.000Z
_atssc	google%3B1	www.ul.ie	/	2022-10-06T11:37:28.000Z
_fbp	fb.1.1630927486169.690958071	.ul.ie	/	2022-02-11T16:53:19.000Z
CookieConsent	{stamp:%270f8V2uxsCQ10JnxVv...	www.ul.ie	/	2022-09-06T11:24:34.000Z

Figure 10 - Chrome DevTool Cookie Information

Size	HttpOnly	Secure	SameSite	SameParty	Priority
23			Lax		Medium
13					Medium
30					Medium
17					Medium
32			Lax		Medium
200		✓			Medium

Figure 11 - Chrome DevTool Cookie Information pt.2

Name – the name given to the cookie.

Value – the unique value given to the cookie which is then used to identify the web user.

Domain – refers to the hosts that have permission to receive the cookie.

Path – refers to the path of the cookie.

Expires / Max-Age – this is the expiration date given to the cookie.

Size – this refers to the size of the cookie in bytes.

HTTP – If this is set to true, then the cookie can only be used over HTTP and disallows the use of JavaScript modification.

Secure – If this is set to true, then the cookie can only be sent through a secure HTTPS connection to the server.

SameSite – uses the experimental SameSite attribute and values contain strict or lax.

SameParty – If this is set to true, this allows the cookie to be set or sent in contexts where all ancestor frames belong to the same First-Party Set.

Priority – This field determines the priority of cookies. The values used in this field are set to either low, medium or high (Basques, 2015).

3.1.4 Investigate websites with existing tools

There are a plethora of tools available today that allow web users to investigate web pages and help us gather a variety of information. These tools can give us a better insight into the background of web pages and allow us to learn new things. With this approach, I would investigate web pages and gather data from them via the use of existing tools that are available to the public. Two possible options for tools are OWASP Zed Attack Proxy (ZAP) and Cookieserve.

OWASP Zed Attack Proxy (ZAP)

ZAP is an integrated penetration testing tool that is easy to use and helps find vulnerabilities in web applications. It is open source and completely free, because of this it is one of the world's most popular web application testing tool. It creates a proxy between the client and the website being tested and monitors all actions while the website is being navigated through. These actions are then scanned by the tool.

There are two types of scans that can be used: passive scan and active scan. Passive scan only looks at the HTTP requests or responses that are being sent to or from the website. This type of scanning checks for any vulnerabilities or potential issues in the web application. ZAP also allows for the ability to attack web applications using commonly known techniques to help find more vulnerabilities. This is known as active scanning. When testing on a web application, active scans can insert malicious scripts and modify site data (Alper, 2019).

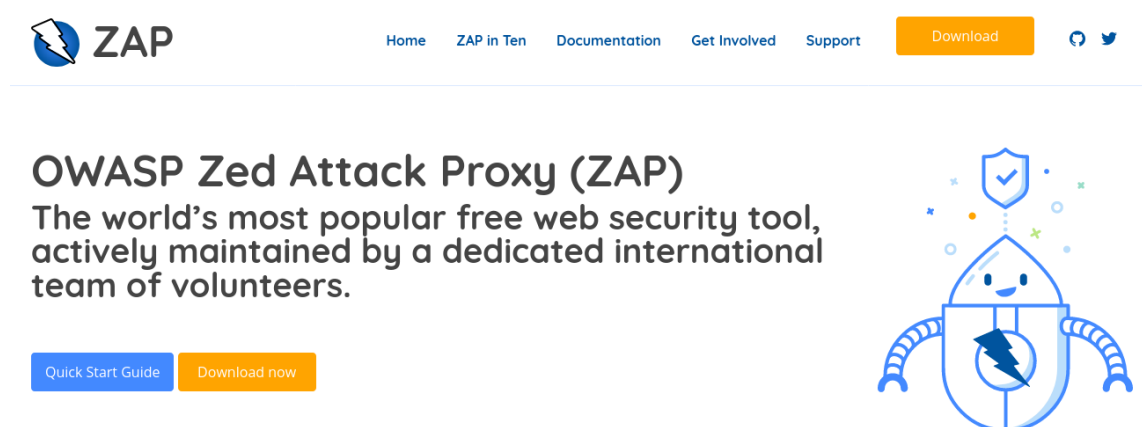


Figure 12 - OWASP Zed Attack Proxy

Cookieserve

Cookieserve is a free online tool that allows web users to look at the cookies employed by a chosen web page. This is done by inputting the URL of a web page into the tool and generating a report. The report gives a detailed view of all the cookies used by the site as well

as information on the type of cookie, it's description, and more. The report is presented in table form with the fields: cookie (the name of the cookie), domain, type, description and duration. Cookieserve categories each cookie found on a web page and gives a detailed description about what they do. Below is a partial report on the cookies being sent from Twitter.com.

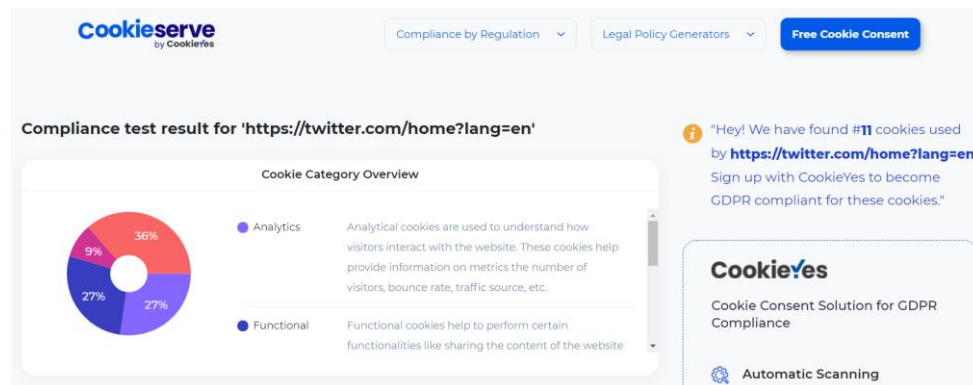


Figure 13 – Cookieserve Cookie results

Cookie	Domain	Type	Description	Duration
personalization_id	.twitter.com	Advertisement	Twitter sets this cookie to integrate and share features for social media and also store information about how the user uses the website, for tracking and targeting.	2 years
guest_id	.twitter.com	Analytics	Twitter installs the guest_id cookie to enable Twitter integration and for social media advertising. This cookie helps to track user behaviour for marketing, to enable sign in and personalize the user's Twitter experience across devices.	2 years

Figure 14 - Cookieserve Cookie results pt.2

3.2 Privacy Browser Extension

In the following sections of this report, I will present research findings on the topic of browser extensions. I will be taking a closer look at some of the privacy-related browser extensions available today such as Privacy Badger, uBlock Origin, etc., as well as more in-depth look at the methods these tools use to boost privacy health. As I plan to develop my own privacy tool in the form of a browser extension for my FYP, I will take a closer look at the development process behind these extensions. This involves examining the languages commonly used in development as well as the tools and environments associated with browser extensions. I will then present a plan based on what I have learned and how I hope to proceed with development.

3.3 Existing Privacy Browser Extensions



Figure 15 - Privacy Badger Logo

3.3.1 Privacy Badger

Created by the Electronic Frontier Foundation (EFF) and first released in 2014, Privacy Badger is a free and open-source software that is available to download on multiple browsers (including Chrome, Mozilla Firefox, etc.) in the form of an extension/add-on. Privacy Badger prevents advertisers and other third-party trackers from tracking web users while they browse online. The approach that Privacy Badger uses to block trackers is through the use of algorithms. As a web user browses the web, the algorithms observe third-party trackers and if a tracker is found across multiple domains as the web user browses, it is then blocked.

Privacy Badger can discern trackers that are required for functionality through the use of the 'yellow list'. This is essentially a list of domains that are allowed access for functionalities sake, however they will still be unable to track web users. Privacy Badger also allows web users to adjust trackers to their liking through a slider. If the slider is red the domain is blocked completely, if it is yellow the domain is allowed but cookies from the domain are blocked and if it is green then both the domain and its cookies are allowed (Emery, 2020).

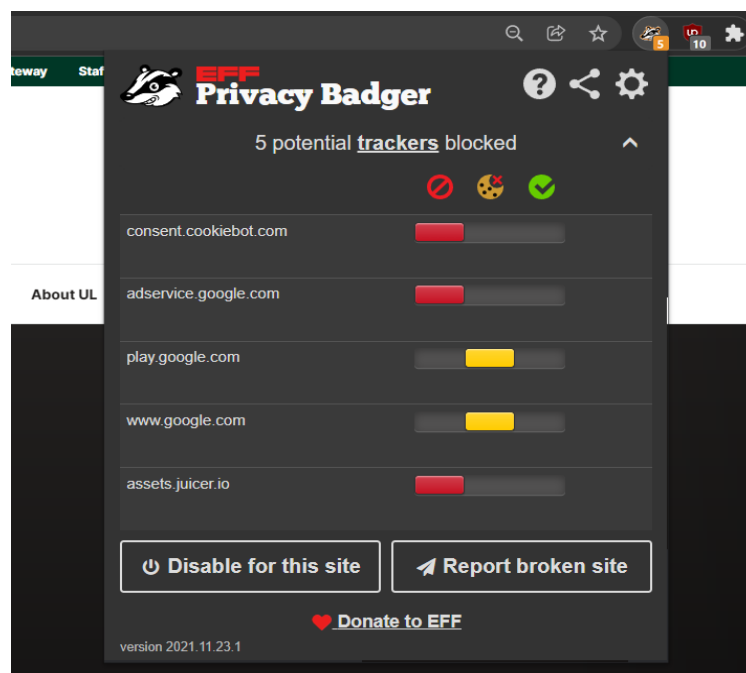


Figure 16 - Privacy Badger UI

3.3.2 uBlock Origin

In 2014, the first iteration of uBlock Origin was created by founder and lead developer Raymond Hill. It is a free and open-source browser extension that is compatible with the likes of Chrome, Mozilla Firefox and many other popular browsers. uBlock Origin has functionalities such as the ability to block ads and third-party trackers. This is done through the blacklist method. This is essentially a list of domains that are known to belong to various advertising entities as well as sources of malware (uBlock Origin, 2014).

With this list it can block these entities from tracking web users and appearing as ads while browsing. While uBlock Origin isn't always 100% effective due to its more simple approach to blocking trackers, it is highly compatible with other privacy-related browser extensions. As a result it works quite well alongside an extension like Privacy Badger (Emery, 2019).



Figure 17 - uBlock Origin Logo

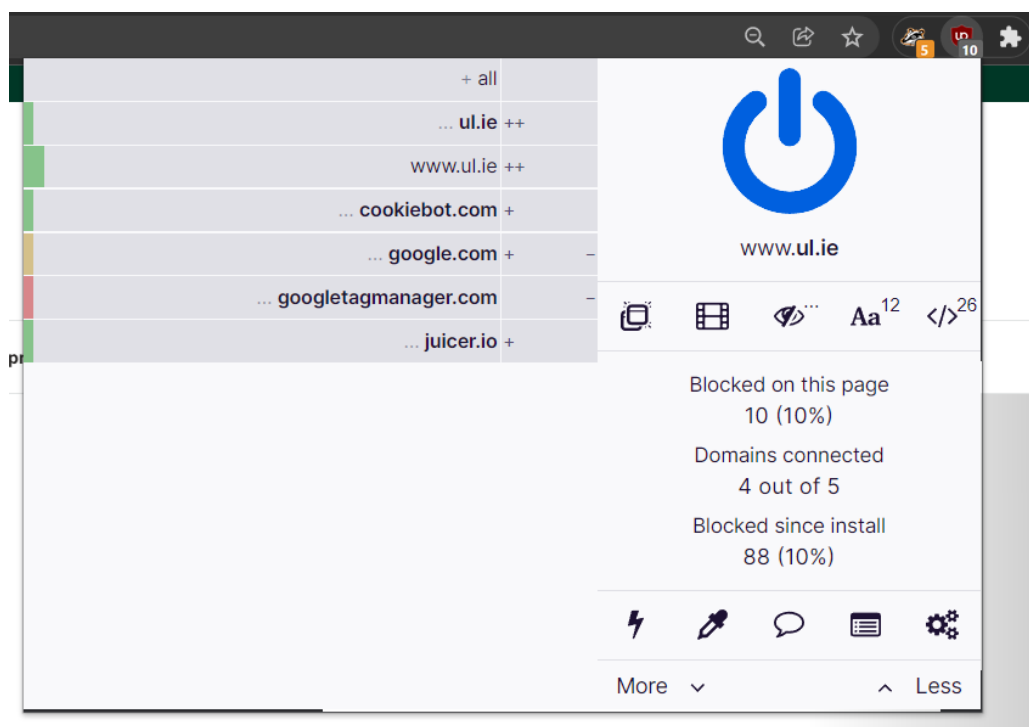


Figure 18 - uBlock Origin UI

3.3.3 Blacklist and Algorithm Methods

There are two primary techniques that privacy extensions utilize to block trackers and improve the privacy health of web users. One way to do this is through the blacklist method. The blacklist method involves blocking third-party trackers or domains when a web user comes into contact with them as they browse the web. Browser extensions like uBlock Origin use a variety of pre-made lists that are updated frequently.

An advantage of the blacklist method would be that it is often more consistent than using algorithms due to the availability of multiple lists where a domain or URL occurs. This results in it being much quicker to take action as it does not have to observe and learn which trackers to block like in the case of the algorithm approach. It also seems easier to develop than the algorithm method as it is just checks for when a domain or URL occurs on the blacklist.

However, there are some disadvantages with the blacklist method. The lists that are used within the software have to be constantly updated to add new blacklisted domains. Also, in the case that a blacklisted entity changes their domain or URL, it can be missed by the software until the blacklist has been updated again. This results in the blacklist method having a reduced effectiveness against trackers and ads.

This can be remedied with the algorithm method however. In cases where a domain or URL has not been added to (or updated on) the blacklist, a browser extension that uses the algorithm method can observe domains that are tracking web users and block them. If a domain has tracked a web user across multiple sites, the software will learn from the tracker's behavior and will block it as a result. One disadvantage of the algorithm method though would be that it can result in the breaking of a website due to the blocking of some trackers being used for site functionality. Due to these false positives, websites could be negatively affected and not work properly for web users (Nibbeling, 2019).

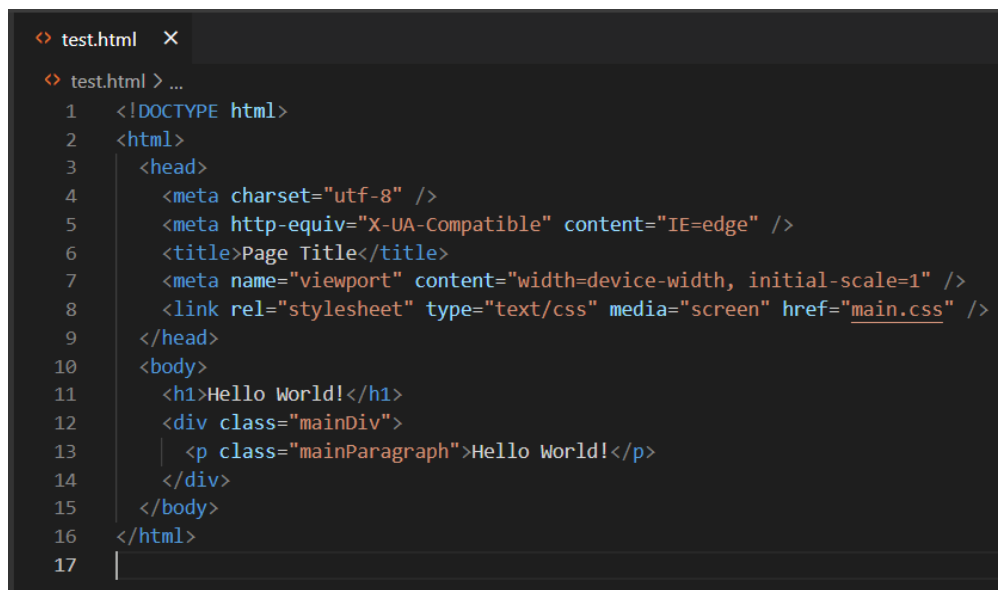
3.4 Languages Used In Development

The languages commonly used in the development of browser extensions are HTML, CSS and JavaScript. The development of a browser extension starts with three basic files to begin with. These are the manifest file, the background script, and the user interface (UI) file. The manifest file is in a .json format and it stores all the metadata information of the extension. This includes info such as the extension's name, version, description, etc. The background script is a JavaScript file that continuously runs and gives the extension functionality. The UI file is another HTML file which the web user directly interfaces with. The UI for browser extensions are commonly displayed as popups and can be styled with the use of CSS. Below is a more in-depth look at the three main languages used in the development of browser extensions (Chrome Developers, 2014).

3.4.1 HTML (HyperText Markup Language)

The first version of HTML was created in 1993 by Tim Berners-Lee. Since then there has been multiple iterations with the current version being HTML5. HTML is a markup language

that browsers can interpret to structure text, images, and more, to form the basic visual aspect of a web page. Markup languages use tags to enclose code in the file. Below is an example of HTML and the typical structure of tags in a file.

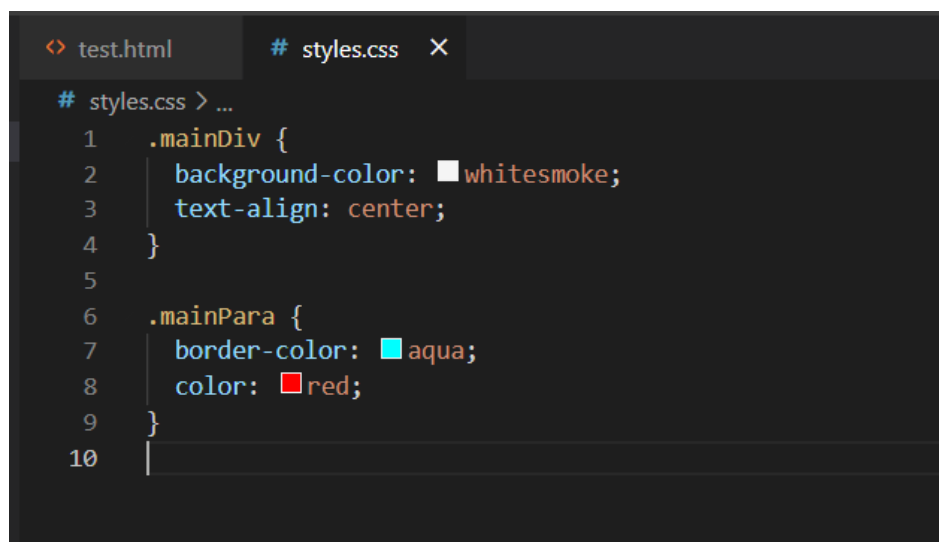


```
<> test.html X
<> test.html > ...
1  <!DOCTYPE html>
2  <html>
3    <head>
4      <meta charset="utf-8" />
5      <meta http-equiv="X-UA-Compatible" content="IE=edge" />
6      <title>Page Title</title>
7      <meta name="viewport" content="width=device-width, initial-scale=1" />
8      <link rel="stylesheet" type="text/css" media="screen" href="main.css" />
9    </head>
10   <body>
11     <h1>Hello World!</h1>
12     <div class="mainDiv">
13       <p class="mainParagraph">Hello World!</p>
14     </div>
15   </body>
16 </html>
17 |
```

Figure 19 - Example of HTML

3.4.2 CSS (Cascading Style Sheets)

The first iteration of CSS was released in 1996 and was proposed by Håkon Wium Lie. The current version that is used today is CSS3. CSS essentially defines the look and style of HTML documents which is then rendered by the browser. It allows for styles to be inherited to other HTML elements and can even overwrite ones that have been previously styled. This hierarchical nature that CSS has, allows for multiple stylesheets to influence a single HTML page (Hoffman, 2017).



```
<> test.html # styles.css X
# styles.css > ...
1  .mainDiv {
2    background-color: ■ whitesmoke;
3    text-align: center;
4  }
5
6  .mainPara {
7    border-color: ■ aqua;
8    color: ■ red;
9  }
10 |
```

Figure 20 - Example of CSS

3.4.3 JavaScript

JavaScript was created in the year 1995 by Brendan Eich. It is a scripting language that allows developers to add certain behaviors and functionalities to web pages. While the original version of JavaScript is ran on the client-side there are new frameworks like Node.js that run on the server-side. Other frameworks and libraries of note are Angular, React and Vue (DeGroat, 2019).

```
<script>
$(document).ready(function() {

    load_cart_data();

    function load_cart_data() {
        $.ajax({
            url: "fetch_cart.php",
            method: "POST",
            dataType: "json",
            success: function(data) {
                $('#cart_details').html(data.cart_details);
            }
        });
    }
})
</script>
```

Figure 21 - Example of JavaScript

3.5 Development Tools/Environments

As the primary languages used in the development of browser extensions are HTML, CSS and JavaScript, even a simple text editor would be a sufficient environment to code in. However, there are many powerful tools and editors available today that are more suited to the development of these languages. Examples of this include Visual Studio Code, Sublime Text and many more.

3.5.1 Visual Studio Code

VS Code is a code editor that accommodates a wide variety of programming languages. First released in 2015 and developed by Microsoft, VS Code is completely free and supports Windows, Linux and Mac OS. Similar to Visual Studio IDE, VS Code is a much more lightweight and streamlined code editor allowing for a simpler work environment. VS Code boasts high customizability with a plethora of extensions available to download. These extensions can add new features, languages and themes to the editor. While VS Code can be used as an editor for many different languages, it is well suited for developing browser extension due to its easy accessibility (Faulds, 2021).

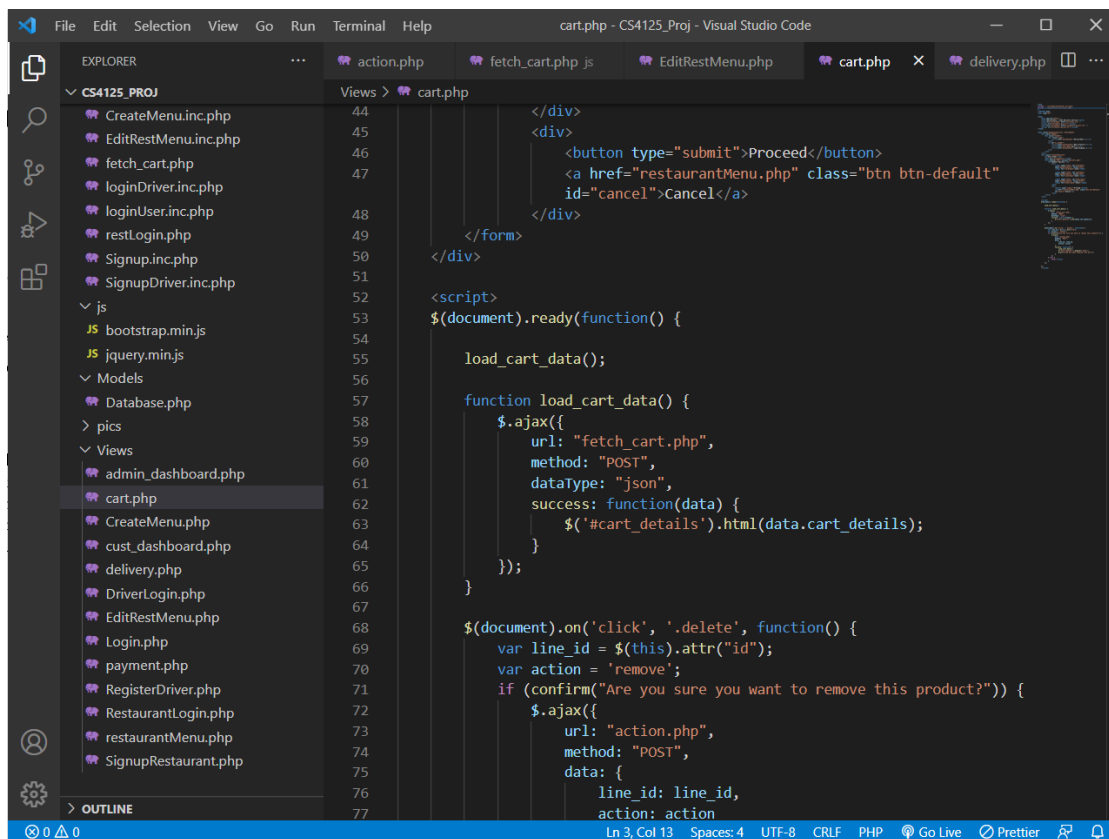


Figure 22 - Visual Studio Code

4.0 Privacy Tool Development

4.1 Initial Plan

In the second semester I planned to develop a browser extension that boosted privacy health much like the previously mentioned extensions - Privacy Badger and uBlock Origin. These two extensions take different approaches to blocking trackers, either through the blacklist method or the algorithm method.

From what I had researched the blacklist method seemed like a simpler approach compared to the algorithm method which seemed more difficult to implement. However, there were pros and cons to both. While the blacklist method can be more consistent at blocking third-party trackers, the use of algorithms can be much more effective in instances where the blacklist approach fails.

To flesh out more of what I had in mind for the privacy tool, I prepared a basic diagram to visualise how the tool would work and flow as well as some basic functional and non-functional requirements for the privacy tool. I also drew a mock-up of a potential UI for the browser extension.

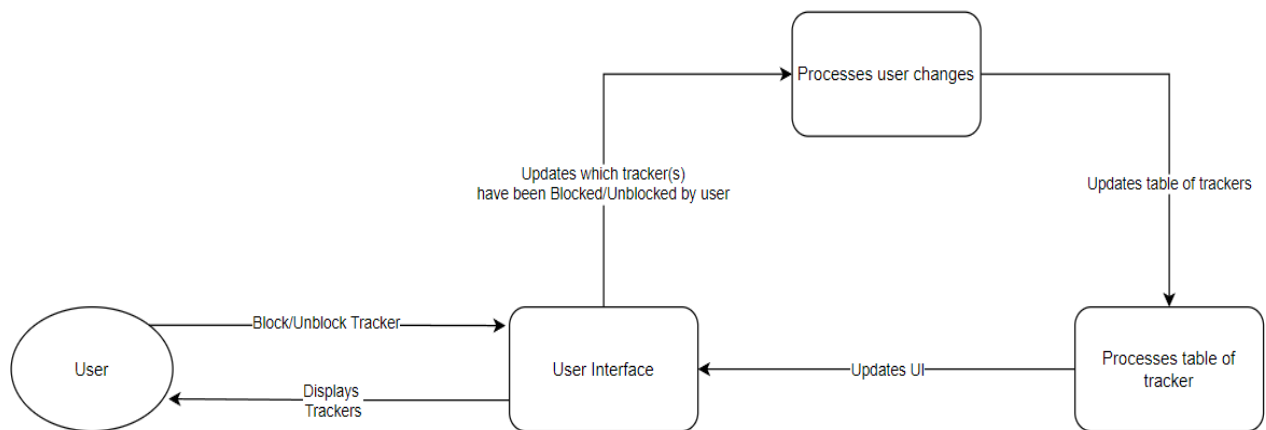


Figure 23 - Basic Diagram for Privacy Tool

Functional & Non-Functional Requirements

FRs

- Display the current trackers that are tracking the user in the browser.
- Allow the user to block/unblock any tracker through the use of a domain blacklist.
- Allow the user to delete listed trackers from their system.
- Update changes made by the user to the blocked/deleted trackers.
- Give a live reading of trackers currently being sent from a domain that the user is currently visiting.

NFRs

- Clear and easily to understand UI.
- UI must be easy to navigate through.
- Extensibility for new features.
- Aesthetically pleasing.
- Fast loading times (if any).

Current Trackers	
Tracker 1	<input type="checkbox"/> Blocked
Tracker 2	<input type="checkbox"/> Blocked
Tracker 3	<input type="checkbox"/> Blocked
Tracker 4	<input checked="" type="checkbox"/> Unblocked

Figure 24 – Original Browser Extension UI Mock-Up

Upon starting the development process on the browser extension, I found that learning JavaScript to be far more difficult to learn than I had initially expected. So ultimately I decided to switch the development to a Windows application written in Python as I had more experience in the language and I felt confident that I would be able to produce a tool that would be of a much better quality.

The Windows application would have the same functional and non-functional requirements as the browser extension however I decided to rework the UI as I now had more options and freedom to design something better.

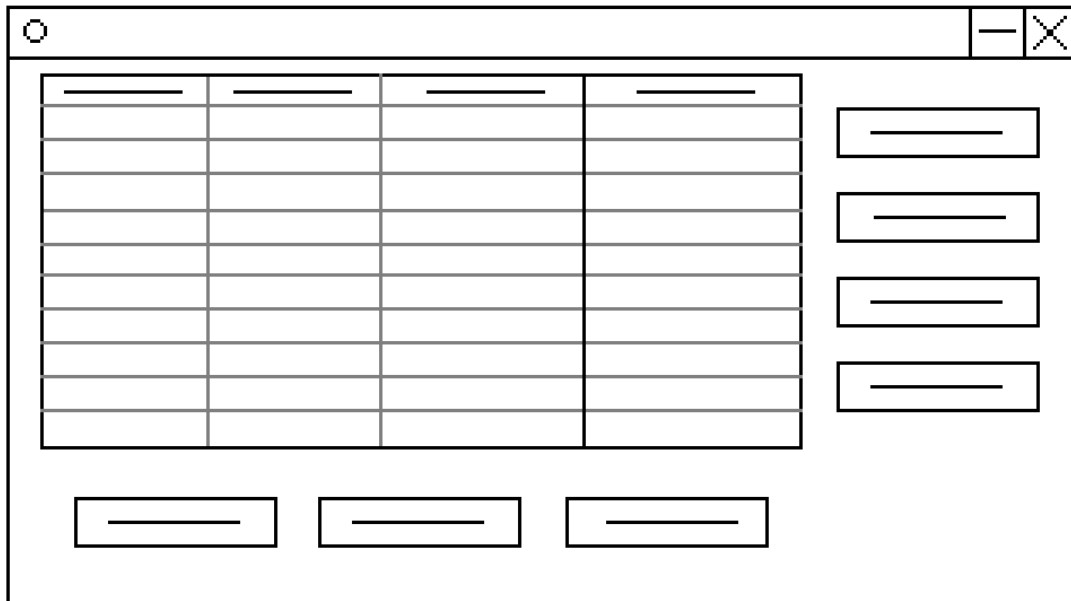


Figure 25 - Windows Application UI Mock-up

4.2 Development Process

For the Privacy Tool, I decided to use the Tkinter GUI framework for the application's user interface. Tkinter is a standard GUI library that is already built-in to Python. I found it fairly easy to pick up and was able to design a decent user interface with it. Below are some screenshots of some Python code utilising the Tkinter framework.

```

128 #####
129 # Tkinter GUI
130
131
132 window = tkinter.Tk()
133 window.title('Browser Privacy Tool')
134 window.geometry('1218x575')
135 window.resizable(False, False)
136 window.configure(bg='snow3')
137 window.iconbitmap('images/cookie.ico')
138
139
140 #####
141 # Tkinter Treeview Table Frame
142
143
144 table_frame = LabelFrame(window, text="All Cookies")
145 table_frame.grid(row=0, column=0, sticky=NW, padx=(10, 0), pady=(2, 0))
146 table_frame.configure(bg='snow3')
147
148 cookie_table = ttk.Treeview(table_frame, height=20)
149 cookie_table.grid(row=0, column=0, padx=5, pady=(0, 5))
150 cookie_table['columns'] = ('row_no', 'domain_name',
151 | | | | | 'cookieID', 'isHTTPOnly', 'creation', 'expiry')
152
153 vertSB = Scrollbar(table_frame, command=cookie_table.yview)
154 vertSB.place(x=937, y=1, height=424)
155 cookie_table.config(yscrollcommand=vertSB.set)
156

```

Figure 26 - Tkinter GUI Code Example

```

284 #####
285 # Sidebar Menu Buttons
286
287 menu_Frame = LabelFrame(window, text="Menu")
288 menu_Frame.configure(bg='snow3')
289 menu_Frame.grid(row=0, column=0, padx=(90, 5), pady=(2, 0), sticky=NE)
290
291
292 cImage = Image.open('images/cookie.png')
293 resizedCImage = cImage.resize((30, 30), Image.ANTIALIAS)
294 cookieImage = ImageTk.PhotoImage(resizedCImage)
295
296 allCookiesButton = Button(
297     menu_Frame, text="View All Cookies", image=cookieImage, compound="left", width=160, padx=20, pady=20, bg='steel blue').pack(padx=15, pady=(70, 15))
298
299
300 sImage = Image.open('images/search.png')
301 resizedSImage = sImage.resize((30, 30), Image.ANTIALIAS)
302 searchImage = ImageTk.PhotoImage(resizedSImage)
303
304 currentCookiesButton = Button(
305     menu_Frame, text="Search for Trackers", command=goToCurrentCookiesWindow, image=searchImage, compound="left", bg='light steel blue', width=160, padx=20, pady=20).pack(padx=15,
306     pady=15)
307
308
309 bImage = Image.open('images/blocklist.png')
310 resizedBImage = bImage.resize((30, 30), Image.ANTIALIAS)
311 blocklistImage = ImageTk.PhotoImage(resizedBImage)
312
313 blocklistsButton = Button(
314     menu_Frame, text="Domain Blocklist", command=goToBlocklistWindow, image=blocklistImage, compound="left", bg='light steel blue', width=160, padx=20, pady=20).pack(padx=15, pady=(15,
315     73))
316
317
318 #####
319 # Window Mainloop
320 window.after(10, refresh_program)
321 window.mainloop()
322
323

```

Figure 27 - Tkinter GUI Code Example 2

After I became more accustomed to Tkinter I was able to produce a decent GUI for the Privacy Tool. Below are some screenshots of the final version of the GUI.

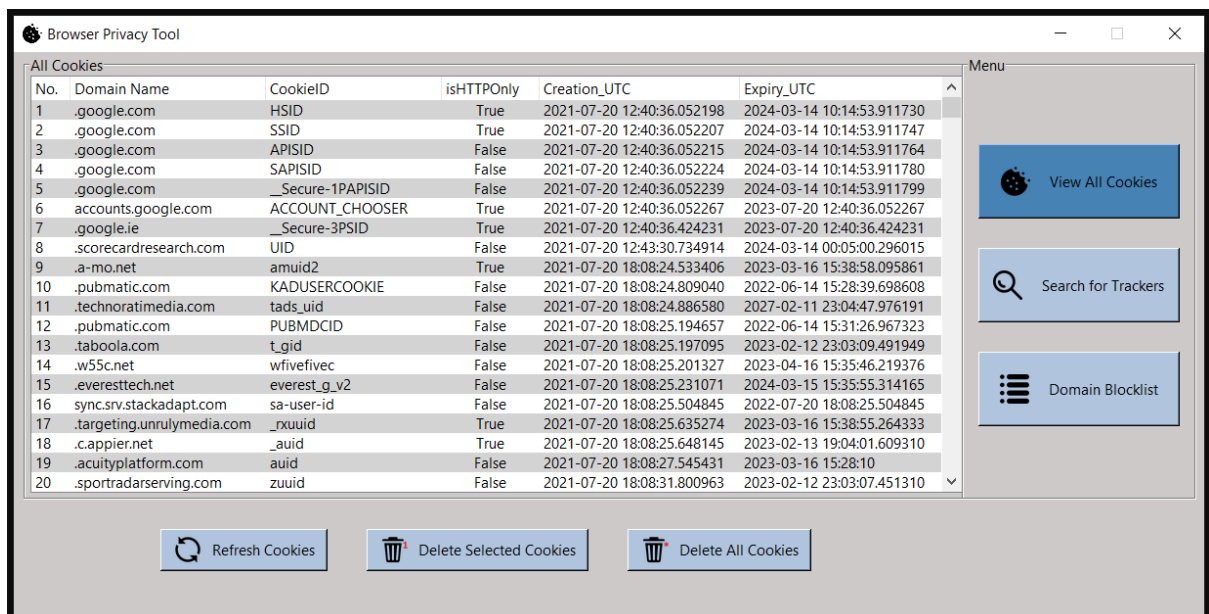


Figure 28 - View All Cookies window

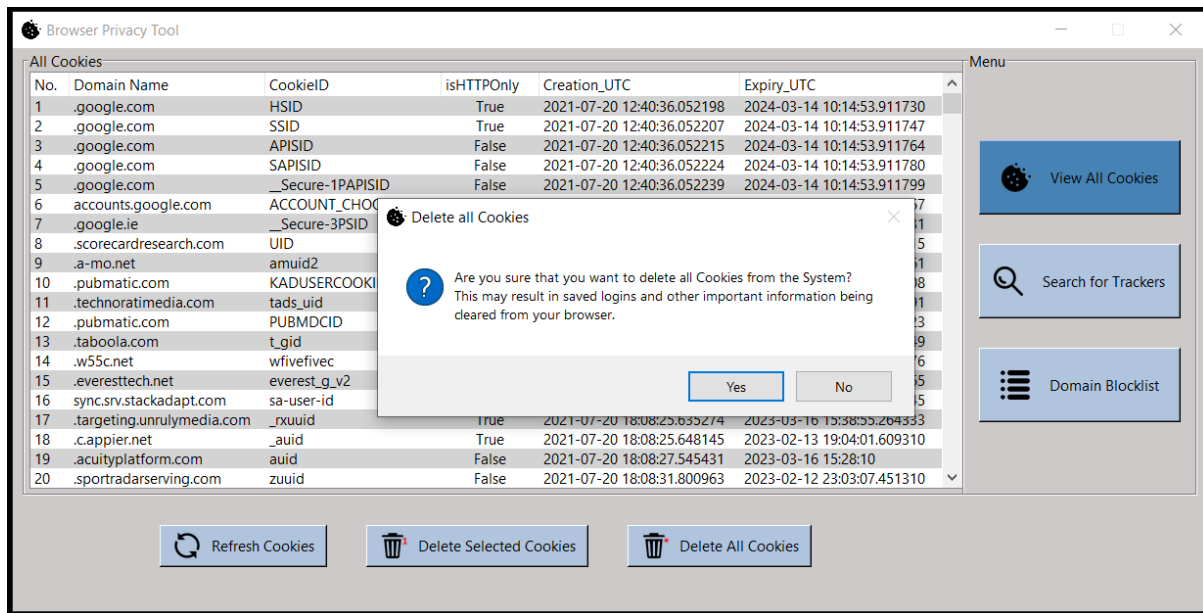


Figure 29 - View All Cookies window w/ Message pop-up

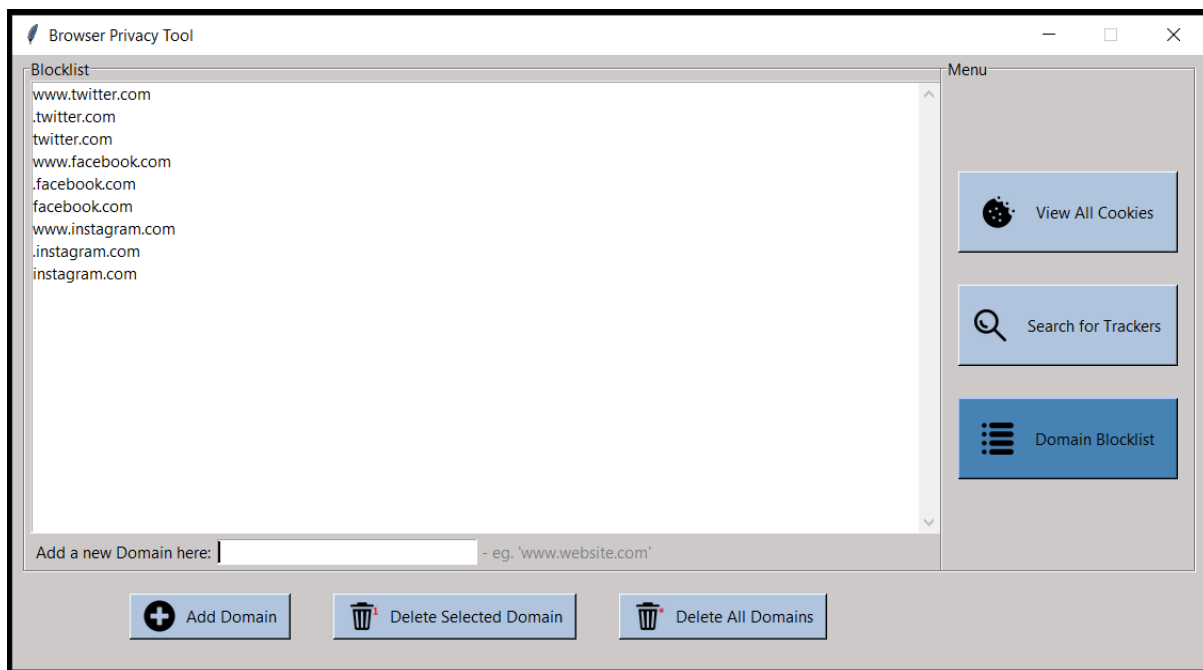


Figure 30 - Domain Blocklist window

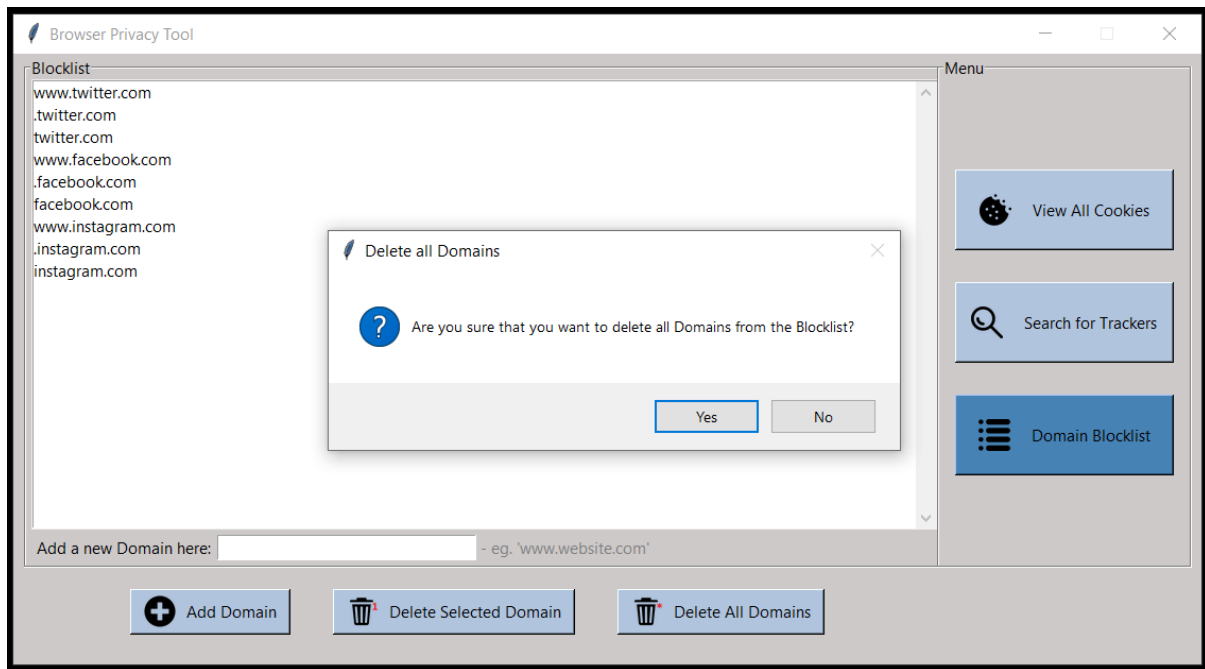


Figure 31 - Domain Blocklist window w/ Message pop-up

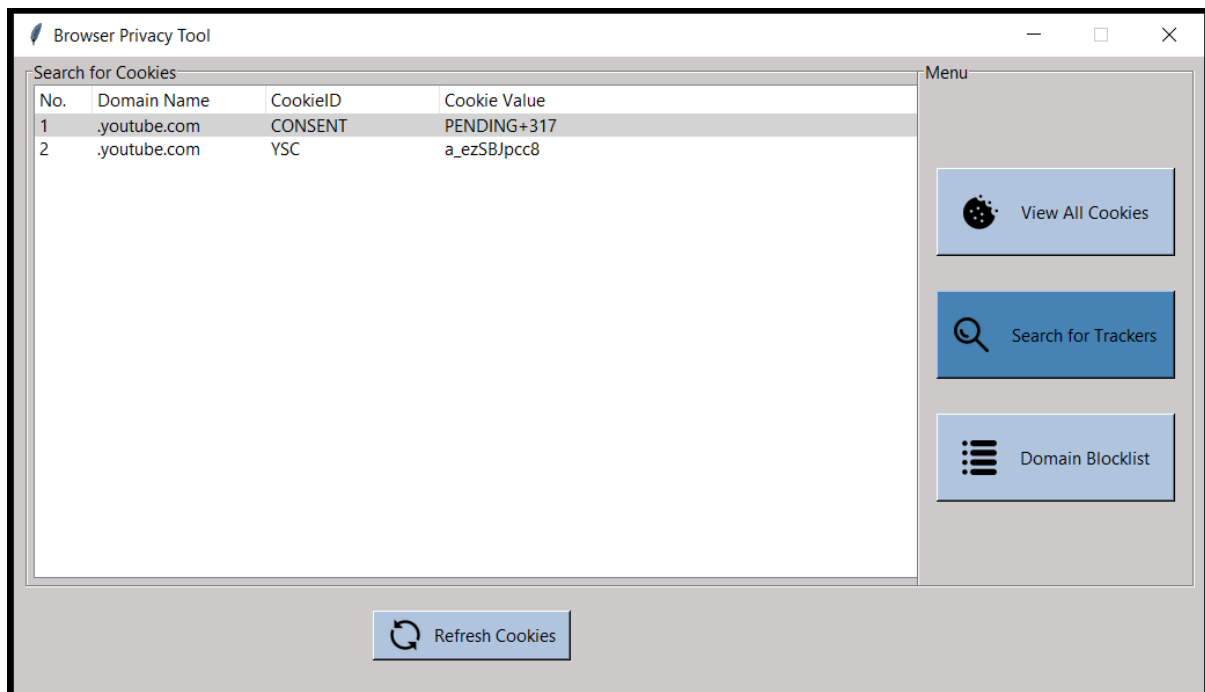


Figure 32 - Search for Trackers window

4.3 Software Testing

To fully test that the Privacy Tool is working as intended I thoroughly checked the code to make sure all methods and GUI elements were working as intended. For the GUI I tested to make sure the correct data was being displayed in the Tkinter tree view table for each window.

All Cookies					
No.	Domain Name	CookieID	isHTTPOnly	Creation_UTC	Expiry_UTC
1	.google.com	HSID	True	2021-07-20 12:40:36.052198	2024-03-14 10:14:53.911730
2	.google.com	SSID	True	2021-07-20 12:40:36.052207	2024-03-14 10:14:53.911747
3	.google.com	APISID	False	2021-07-20 12:40:36.052215	2024-03-14 10:14:53.911764
4	.google.com	SAPISID	False	2021-07-20 12:40:36.052224	2024-03-14 10:14:53.911780
5	.google.com	__Secure-1PAPISID	False	2021-07-20 12:40:36.052239	2024-03-14 10:14:53.911799
6	accounts.google.com	ACCOUNT_CHOOSER	True	2021-07-20 12:40:36.052267	2023-07-20 12:40:36.052267
7	.google.ie	__Secure-3PSID	True	2021-07-20 12:40:36.424231	2023-07-20 12:40:36.424231

Figure 33 - All Cookies Table

Search for Cookies			
No.	Domain Name	CookieID	Cookie Value
1	.facebook.com	fr	0qPxv8Mo6x49yi7kf..BiRcEd.GI.AAA.0.0.BiRcEd.AWXYoH
2	.facebook.com	sb	HcFFYgErC4ie4ovMqKbBakWs

Figure 34 - Search for Cookies Table

Blocklist	
www.twitter.com	
.twitter.com	
twitter.com	
www.facebook.com	
.facebook.com	
facebook.com	

Figure 35 - Blocklist Table

I also tested each of the button used within each window and made sure that the changing of windows when a button was pressed displayed correctly and that no visual bugs were present in the tree view tables.

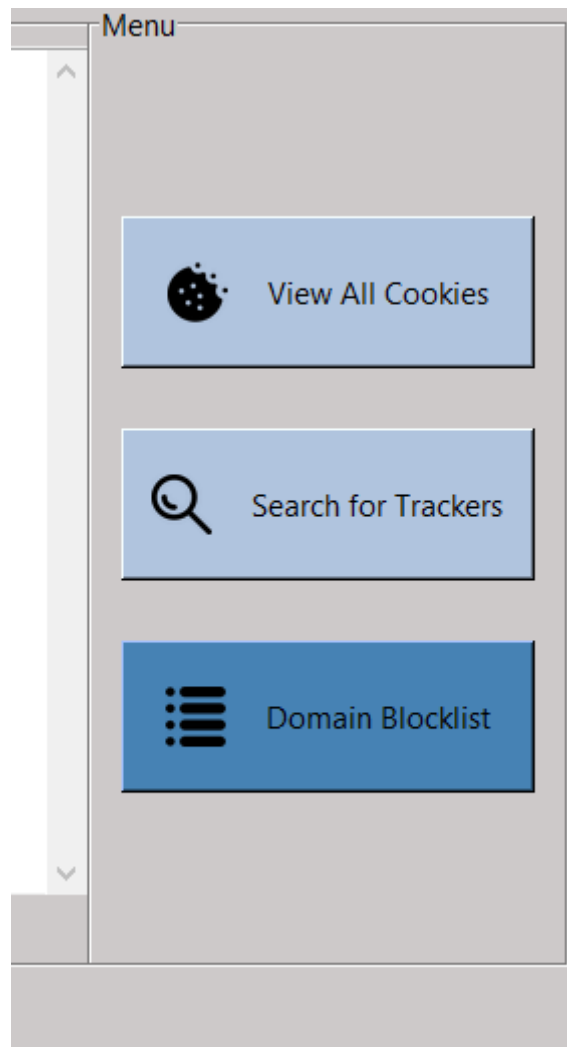


Figure 36 - Menu Buttons

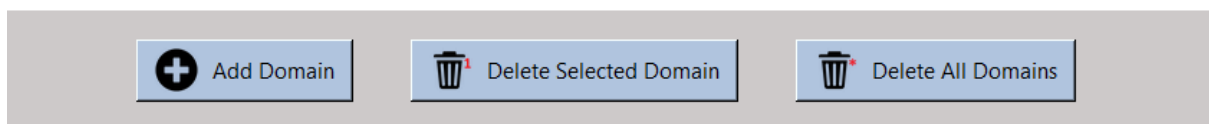


Figure 37 - Table Buttons

Below is a more detailed look at the testing that was done for the browser privacy tool:

VAC – View All Cookies window

SFT – Search For Trackers window

DB – Domain Blocklist window

Test Case	Description	Test Step	Expected Result	Status
Refresh Cookies Button (VAC)	Table should refresh to show new Cookies on the system when button is clicked.	Click Refresh Cookies Button.	Table will refresh to show new Cookies, if there are any.	Pass
Delete Selected Cookies Button (VAC)	Selected Cookies (one or many) should be deleted when button is clicked.	Click Delete Selected Cookies Button.	Deleted Cookie(s) will disappear from the table.	Pass
Delete All Cookies Button (VAC)	All Cookies should be deleted when the button is clicked.	Click Delete All Cookies Button.	All Cookies will be deleted from the table.	Pass
Search For Trackers Button (VAC)	The Search for Trackers window should open and the View all Cookies window should close when button is clicked.	Click Search For Trackers Button.	The Search for Trackers window will open and the View all Cookies window will close.	Pass
Domain Blocklist Button (VAC)	The Domain Blocklist window should open and the View all Cookies window should close when button is clicked.	Click Domain Blocklist Button.	The Domain Blocklist window will open and the View all Cookies window will close.	Pass
Refresh Cookies Button (SFT)	Table should refresh to show the live Cookies being sent from the website open in the current Chrome tab.	Click Refresh Cookies Button.	Table will refresh to show the live cookies being sent from a websites server.	Pass
View All Cookies Button (SFT)	The View all Cookies window should open and the Search for Trackers window should close when button is clicked.	Click View all Cookies Button.	The View all Cookies window will open and the Search for Trackers window will close.	Pass
Domain Blocklist Button (SFT)	The Domain Blocklist window should open and the Search for Trackers window should close when button is clicked.	Click Domain Blocklist Button.	The Domain Blocklist window will open and the Search for Trackers window will close.	Pass

Test Case	Description	Test Step	Expected Result	Status
Add Domain Button (DB)	Once a domain is entered into the input box, the Add Domain button should enter the domain into the list box when the button is clicked.	Click Add Domain Button.	The inputted domain will be entered into the list box when the button is clicked	Pass
Delete Selected Domain Button (DB)	Selected Domains (one or many) should be deleted when button is clicked.	Click Delete Selected Domain Button.	Deleted Domain(s) will disappear from the list box.	Pass
Delete All Cookies Button (DB)	All Domains should be deleted when the button is clicked.	Click Delete All Domains Button.	All Domains will be deleted from the list box.	Pass
View All Cookies Button (DB)	The View all Cookies window should open and the Domain Blocklist window should close when button is clicked.	Click View all Cookies Button.	The View all Cookies window will open and the Domain Blocklist window will close.	Pass
Search For Trackers Button (DB)	The Search For Trackers window should open and the Domain Blocklist window should close when button is clicked.	Click Search For Trackers Button.	The Search for Trackers window will open and the Domain Blocklist window will close.	Pass
Domain Entry Box (DB)	When a domain is entered in the style 'www.website.com' and the Add Domain button is clicked, the domain along with two other variations of the domain should be entered into the Blocklist.	Type new domain into entry box and click the Add Domain Button.	When a domain is typed in the entry box and the Add Domain button is clicked, the domain and two other variations of it will be added to the Blocklist.	Pass
Window change stress test	When the window is changed multiple times by navigating through the menu to the VAC, SFT or DB windows, they should transition smoothly.	Change windows to VAC, SFT or DB windows.	When the window is changing multiple times to the VAC, SFT or DB windows, they should transition smoothly	Fail

Figure 38 - Software Testing Table

In my software testing I found the application to be mostly running as intended with all the buttons correctly functioning as well as all tables and list boxes displaying correctly. One bug

however that I was unable to fix due to my own skill level and time constraints, was that when navigating through the menu and changing windows multiple times to either the View All Cookies, Search for Trackers or Domain Blocklist windows, I found that the program would crash and close out, resulting in the application having to be run again.

On closer inspection of this issue I figured it had something to do with the methods I set up for changing the Tkinter windows, as I would use `window.destroy()` to get rid of the previous window and then import the class containing the next window to be opened. My guess was that destroying a window more than once was causing the program to crash. To fix this I researched another method of doing this which involved an object-oriented approach, however due to my own inexperience with Python and the time constraints I was ultimately unable to fix this issue.

5.0 Evaluation / Conclusion

Looking back on the time I spent researching the various types of Web Tracking Technologies and developing the Browser Privacy Tool, I am very happy with what I was able to produce.

The aims of my final year project were to research commonly used tracking techniques that are utilised by many websites today. I found there to be two main types of web trackers; Stateful and Stateless Trackers. With Stateful referring to trackers that are stored on the client side. Examples of this being HTTP Cookies, Flash Cookies, Evercookies, etc. Whereas Stateless is a form of tracking that doesn't require storing on the client side, i.e. Fingerprinting.

Another aim I had was to research ways of measuring privacy through utilising privacy protection tools. Methods such as browser extensions, VPNs, Private Browsers, etc., allowed for the ability to monitor and boost privacy health. When researching more into browser extensions, I found that the two main methods that are commonly used in the development of these privacy tools are the blacklist and the algorithm method. With the blacklist method involving a domain blacklist that blocks specified domains and the algorithm method involving a range of regular expressions and algorithms to boost privacy health.

I also aimed to develop a privacy tool similar to those that I had researched. I originally planned to develop a browser extension similar to Privacy Badger or UBlock Origin but due to my very little experience with Javascript, I decided to switch to developing a windows application in Python. This resulted in a tool that allows someone to measure their own privacy by giving them the ability to monitor and control HTTP Cookies that are currently on their system. With the use of the Blacklist method that I had previously researched, I implemented a blacklist feature that allows a user to block specified domains. I was also able to implement a 'Search for Trackers' feature that allowed for the analysis of Cookies from a domain that was currently being visited on Google Chrome. This essentially allowed for a live viewing of the Cookies being sent from the domain's server.

6.0 References

- Alper, 2019. *Running Penetration Tests for your Website with OWASP ZAP*. s.l.:s.n.
- Arrieta, A., Cyphers, B., Miagkov, A. & Barnett, D., 2020. *Privacy Badger Is Changing to Protect You Better*. s.l.:s.n.
- Basques, K., 2015. *View, edit, and delete cookies*. s.l.:s.n.
- Bielova, N., 2017. *Web Tracking Technologies and Protection Mechanisms*. s.l.:s.n.
- Bujlow, T., Carela-Espanol, V., SolÉ-Pareta, J. & Barlet-Ros, P., 2017. *A Survey on Web Tracking: Mechanisms, Implications, and Defenses*. s.l.:s.n.
- Chrome Developers, 2014. *Learn about developing extensions for Chrome.*, s.l.: s.n.
- Crawford, E., 2020. *Website Tracking: Why and How Do Websites Track You?*. s.l.:s.n.
- DeGroat, T., 2019. *The History of JavaScript: Everything You Need to Know*, s.l.: s.n.
- Emery, J., 2019. *Review: Ad blocking with uBlock Origin*. s.l.:s.n.
- Emery, J., 2020. *Review: Privacy Badger Browser Extension*. s.l.:s.n.
- Faulds, J., 2021. *Microsoft VS Code review*, s.l.: s.n.
- Hoffman, J., 2017. *A Look Back at the History of CSS*, s.l.: s.n.
- Internet Health Report, 2018. *The good, the bad and the ugly sides of data tracking*, s.l.: s.n.
- Joseph, C., 2020. *How Browser FINGERPRINTING Works (and How to STOP It)*. s.l.:s.n.
- Kamkar, S., 2010. *Evercookie*. s.l.:s.n.
- Kim, D., 2014. *Poster: Detection and Prevention of Web-based*. s.l.:s.n.
- Kristol, D. M., 2001. *HTTP Cookies: Standards, Privacy, and Politics*. s.l.:s.n.
- Nibbeling, N., 2019. *Comparing privacy plugins*. s.l.:s.n.
- Rafter, D., 2021. *How to protect your privacy online*. s.l.:s.n.
- Richard, 2010. *Browser Plugins vs Extensions – the difference*. s.l.:Stealth.
- Sanchez-Rola, I., Ugarte-Perdrero, X., Santos, I. & Bringas, P. G., 2017. *The web is watching you: A comprehensive review of web-tracking techniques and countermeasures*. s.l.:s.n.
- Schmidt, J., 2020. *Does the dark side still have (ever)cookies?*. s.l.:s.n.
- Sikorski, R. & Peters, R., 1997. *Cookie monster?*. s.l.:s.n.
- Sipior, J. C., Ward, B. T. & Mendoza, R. A., 2011. *Online Privacy Concerns Associated with Cookies, Flash Cookies, and Web Beacons*. s.l.:s.n.
- Soltani, A. et al., 2009. *Flash Cookies and Privacy*. s.l.:s.n.
- uBlock Origin, 2014. *uBlock Origin - Free, open-source ad content blocker.*. s.l.:s.n.
- VPN Accounts, 2020. *Understanding VPN & Tracking Cookies*. s.l.:s.n.

Whittaker, Z., 2018. *Cybersecurity 101: How to browse the web securely and privately*. s.l.:s.n.