GDPR Cookie Monster Mash

Tiago André Carneiro Bessa
Master in Information Security
Department of Computer Science
Faculty of Sciences of the University of Porto
2025

GDPR Cookie Monster Mash

Tiago André Carneiro Bessa

Dissertation carried out as part of the Master in Information Security

Department of Computer Science

2025

Supervisor
Luís Antunes, Full Professor, Faculty of Sciences of the University of Porto

External Host Supervisor
Luís Valente, Area Leader Cybersecurity & Information Systems Audit, MC Sonae

Acknowledgements

I would like to thank Professor Lus Filipe Coelho Antunes for his support, clarification and guidance, which proved to be fundamental at the beginning of this dissertation. I would also like to thank Luís Valente for his continuous support throughout the dissertation.

Finally, I would like to thank my family who have always encouraged me to keep going.

Resumo

Os cookies desempenham um papel fundamental na manutenção de um registo do estado do website entre transações e ligações cliente-servidor. Podem melhorar a experiência do utilizador e ajudar os proprietários a compreender o comportamento do utilizador durante as visitas. Desde a sua introdução, os cookies tornaram-se parte integrante do funcionamento da Internet moderna e são utilizados em vários websites para diversos fins. No entanto, a ausência de um quadro normalizado que regule as informações que os cookies, incluindo os cookies de terceiros, podem recolher e para que fins, suscita preocupações de segurança significativas.

Esta tese tem como objetivo desenvolver uma ferramenta para analisar cookies de sites com base numa metodologia de segurança por desenho. A ferramenta executa várias tarefas, incluindo a verificação de banners de consentimento de cookies, a validação de opções de configuração de cookies e a avaliação da conformidade dos cookies com quadros legais, como o Regulamento Geral de Proteção de Dados (Regulamento (UE) 2016/679). A análise abrange áreas críticas, como a avaliação da funcionalidade e do comportamento dos banners de consentimento de cookies, a análise das páginas de configuração de cookies e a verificação da transferência de cookies para fora da UE.

Através deste projeto, o objetivo é dotar a organização com uma ferramenta capaz de testar a conformidade, garantindo a transparência e a segurança na gestão de cookies, com a atual Lei de Proteção de Dados Pessoais (Lei n.º 58/2019).

Abstract

Cookies play a key role in keeping track of the state of the website between transactions and client-server connections. They can improve the user experience and help owners understand user behaviour during visits. Since their introduction, cookies have become an integral part of how the modern Internet works and are used on many websites for various purposes. However, the absence of a standardised framework governing what information cookies, including third-party cookies, can collect and for what purposes, raises significant security concerns.

This thesis aims to develop a tool to analyse website cookies based on a security by design methodology. The tool performs various tasks, including checking cookie consent banners, validating cookie configuration options and assessing cookie compliance with legal frameworks such as the General Data Protection Regulation (Regulation (EU) 2016/679). The analysis covers critical areas such as assessing the functionality and behaviour of cookie consent banners, analysing cookie configuration pages and verifying the transfer of cookies outside the EU.

Through this project, the aim is to provide the organisation with a tool capable of testing compliance, ensuring transparency and security in the management of cookies, with the current Personal Data Protection Law (Law no. 58/2019).

Table of Contents

List of Tables

List of Charts

List of Figures

List of Abbreviations

UP	UNIVERSITY OF PORTO
EU	EUROPEAN UNION
EEA	EUROPEAN ECONOMIC AREA
GDPR	GENERAL DATA PROTECTION REGULATION
CCPA	CALIFORNIA CONSUMER PRIVACY ACT
ePD	EPRIVACY DIRECTIVE
ePR	EPRIVACY REGULATION
CMP	CONSENT MANAGEMENT PLATFORM
HTTP	HYPERTEXT TRANSFER PROTOCOL
HTTPS	HYPERTEXT TRANSFER PROTOCOL SECURE
RFC	REQUEST FOR COMMENTS
WWW	WORLD WIDE WEB
XSS	CROSS-SITE SCRIPTING
HSTS	HTTP STRICT TRANSPORT SECURITY
HTML	HYPERTEXT MARKUP LANGUAGE
CSRF	CROSS-SITE REQUEST FORGERY
SSO	SINGLE SIGN-ON
DPA	DATA PROTECTION AUTHORITIES
EDPB	EUROPEAN DATA PROTECTION BOARD
WCAG	WEB CONTENT ACCESSIBILITY GUIDELINES
TMS	TAG MANAGEMENT SYSTEMS
TCF	TRANSPARENCY AND CONSENT FRAMEWORK
DNT	DO NOT TRACK
GPC	GLOBAL PRIVACY CONTROL
LLM	LARGE LANGUAGE MODELS
SbD	SECURITY BY DESIGN
HLD	HIGH-LEVEL DESIGN
PDF	PORTABLE DOCUMENT FORMAT
PNG	PORTABLE NETWORK GRAPHICS
DOM	DOCUMENT OBJECT MODEL
NLP	NATURAL LANGUAGE PROCESSING
WCAG	WEB CONTENT ACCESSIBILITY GUIDELINES

1 Introduction

Cookies play a significant role in the functionality of modern websites[29][30]. They serve various purposes, such as maintaining a user's shopping basket during an online shopping session or enabling auto-login features. For certain websites, cookies and tracking mechanisms are indispensable, as they help retain a client’s status during a visit. Additionally, cookies are essential for authenticating users and providing access to restricted sub-pages and services. These functionalities exemplify the role of first-party cookies, which directly support the primary purpose of the website.

In contrast, third-party cookies serve entities other than the domain the user is visiting. These cookies are placed by external services, often for advertising purposes, which can then access user session data and browsing behaviour across multiple websites. This process of cross-site tracking, which can involve the collection of identifiable information without adequate consent, may violate European data protection regulations and constitutes a significant portion of the cookies present on many websites.

The management of cookies poses significant challenges for both website owners and users. Website owners, as data controllers, often struggle with managing the multitude of scripts required for a website's operation[31]. Visitors, on the other hand, frequently lack the ability to control which scripts run on their browsers. Furthermore, the implementation of cookie banners is often inadequate. Many banners fail to provide users with a clear and transparent choice to "opt-in" to certain types of cookies or do not offer this option at all. Even when consent mechanisms exist, they often lack transparency, failing to disclose the specific cookies and services in use or their purposes. Instead, they rely on broad categories for consent, which fall short of the GDPR's requirements for granular control and informed consent as outlined in Article 7 and the principles of Article 5, nº1, line c): “Personal data shall be adequate, relevant, and limited to what is necessary.”

While the GDPR provides a cornerstone for cookie-related regulations, it is not the only legal framework addressing this issue. The GDPR itself mentions cookies explicitly only once, in Recital 30, stating that “cookie identifiers” used to identify users qualify as personal data. Another pivotal regulation is the ePrivacy Directive (ePD), originally passed in 2002 and amended in 2009. The ePD focuses on the confidentiality of electronic communications and user tracking, playing a critical role in the widespread adoption of cookie consent banners across the web. Often referred to as the "cookie law," the ePD is complemented by the proposed ePrivacy Regulation (ePR), introduced

by the European Commission in 2017. The ePR aims to modernize the ePD by addressing emerging issues such as browser fingerprinting, new communication methods, and enhanced protections for metadata. Although the ePR was intended to be enacted alongside the GDPR, its adoption has been delayed.

Beyond the European Union, other jurisdictions have introduced similar regulations to address data privacy concerns. For example, the California Consumer Privacy Act (CCPA), enacted in the United States, provides a framework for managing user data within the state of California. While similar in intent to the GDPR, the CCPA differs in scope and definition. One notable distinction is that the CCPA excludes publicly available information from its definition of personal data, whereas the GDPR applies regardless of the data's source.

1.1 Proposed Solution

To establish a more transparent and compliant relationship between users and website owners, this thesis proposes the development of an application designed to address the core challenges of cookie management compliance. The proposed solution integrates cookie scanning, granular consent mechanisms, and detailed information disclosure into a cohesive application, aiming to bridge the gap between legal requirements and practical implementation, creating a balanced approach to data privacy.

One of the main concerns driving this proposal is the prevalent issue of unlawful data sharing, largely resulting from the irresponsible practices of organizations and website owners who allow third-party cookies to be set without proper oversight[32][33][34]. The deficiencies in how both public and private entities address these challenges highlight the need for a robust, user-centric solution. The following outlines the core functionalities of the proposed application.

The first interaction between the user and the website often begins with the cookie consent banner and the application will ensure the verification of the following:

- **Banner Dimensions and Text**: Validation of the banner's visibility, readability, and compliance with regulatory requirements.
- **Button Functionality**: Accurate behavior of "Accept All Cookies," "Set Cookies," and "Refuse All Cookies" buttons, ensuring users have clear and actionable choices.
- **Overlay Presentation**: Evaluation of the banner's overlay behavior to ensure it does not obstruct or force unwanted consent.

- **Cookie Consent Values**: Confirmation that the *CookieConsent* cookie stores appropriate values upon accepting or refusing all categories of cookies.

The cookies configuration page is a critical component for achieving granular consent and it will test and validate the following:

- **Window Launch and Closure**: Proper functionality for opening and closing the configuration page.
- **Title, Text, and Links**: Verification of clear and accessible text, including links to the cookies policy and detailed cookie information.
- **"View List of Cookies" Functionality**: Ensuring users can easily access a comprehensive list of cookies used on the website.
- **Cookie Consent Updates**: Verification of how the *CookieConsent* cookie is updated based on the user's selections.

The application will address the complexities of tracking user consent and the transfer of data outside the EU and key functionalities include:

- **Cross-Border Data Analysis**: Identification of cookies that facilitate data transfers outside the EU to ensure compliance with regulations.
- **Initial Consent States**: Verification of the initial state of the *CookieConsent* cookie before user interaction.
- **Consent Updates**: Tracking and validating how the *CookieConsent* values are modified based on specific user authorizations.

To further empower users, the application will integrate functionalities for managing browser settings alongside website-specific controls and these features include:

- **Browser Configuration Access**: Testing the opening of browser settings to modify or remove cookie permissions.
- **Toggle Button Functionality**: Examination of the toggle buttons to ensure they accurately display the status of cookie permissions.
- **Banner Reappearance**: Validation of the reappearance of the cookie consent banner after cookies are cleared through browser settings.

The proposal will combine existing tools, each optimized for specific tasks, with additional features developed to address shortcomings in current cookie management compliance

solutions. By providing a comprehensive suite of tools, the application aims to simplify compliance for website owners and improve transparency and control for users. This holistic approach ensures adherence to regulatory frameworks like the GDPR and ePrivacy Directive while fostering user trust through informed consent and actionable choices.

1.2 Contributions

The theme of this dissertation was proposed in partnership with MC Sonae, a leading company in the food retail sector, and it operates a diverse portfolio of businesses, offering customers a wide range of high-quality products and services at competitive prices. Hence, the cookie management compliance can strengthen its ability to maintain transparency and control over cookie usage, reinforcing customer trust and meeting regulatory standards.

In addition to the continuous improvement in services and products in production and to support the activity of the Data Protection Officer, and therefore, it seeks to develop new solutions, and it will later be implemented as a service. Furthermore, the proposed solution aims to be a valid contribution to the scientific community, who, contributed with open-source tools and information to the fulfilment of the present proposal.

1.3 Thesis Structure

This document is structured in the following order:

- **State Of The Art** - This chapter goes into detail about what constitutes a "cookie" and its uses. Elaborate on some of the most well-known and used market solutions of today that aim the management of cookies.
- **Design** - The chapter consists of the design process, describing the main concepts, specifying the requirements, and all the necessary components that comprise its inner workings.
- **Development** - This chapter details the steps taken in the development process focusing on explaining some of the major choices that were made and how they impact the features of the application. The chapter also explains how each feature utilizes all the technologies discussed in the Design chapter.
- **Conclusion** - This chapter summarizes the progress made during the time dedicated to the project and all the suggested future work that might help mature the developed solution.

2 State Of the Art

In the modern digital era, information is characterized by an unprecedented exchange, with the internet standing as the primary conduit for communication, commerce and content consumption. Central to the functioning of this complex ecosystem are *cookies*, small text files stored on a user's device by websites they visit. Initially conceived as tools for enhancing web usability, *cookies* have evolved into powerful instruments for data collection, user tracking, and behavioural advertising, creating a nuanced and often contentious landscape where utility clashes with fundamental privacy rights.

2.1 Brief History of Cookies

The foundation of the World Wide Web (WWW), Hypertext Transfer Protocol (HTTP), is inherently stateless. This means each request from a web browser to a server is treated as an independent transaction, unrelated to any previous requests. While this simplicity helped the web scale, it created challenges for interactive experiences, as a website could not remember if a user was logged in or what items they had added to a shopping cart.

To solve this problem, Lou Montulli[23], an engineer at Netscape Communications, invented the cookie in 1994. He derived the name from the Unix term "magic cookie," which refers to small data tokens that programs pass between each other. The system Montulli devised allowed a web server to send a small piece of data (the cookie) to a client's browser. The browser would then store this data and send it back to the server with subsequent requests, enabling the server to identify returning users and maintain a "stateful" session.

Netscape’s first use case for cookies was to check whether a visitor had previously accessed its site[22], but the technology quickly facilitated features like persistent logins, user preferences (e.g., language settings), and e-commerce functionality. Montulli filed a patent for this "method and apparatus for client-server transaction persistence" in 1995, which was granted in 1998 (US Patent 5,774,670)[23].

Microsoft quickly followed suit, adding cookie support to Internet Explorer 2 in 1995. However, public discussion of the technology began to surface soon after. A 1996 article in the *Financial Times* warned that cookies could be stolen or misused, which resonated with a public that was largely unaware the technology even existed.

Because early browsers implemented cookies differently, standardization was necessary. Netscape’s format became the reference model and was formalized in RFC

2109[26], which recommended disabling third-party cookies by default. A subsequent effort in 2000 to loosen these restrictions (RFC 2965) was not adopted by Internet Explorer or Netscape[24]. A decade later, the effort to resolve lingering inconsistencies culminated in RFC 6265[20], which was published in 2011 and remains the governing specification for cookies today.

2.2 Dichotomy of Cookies: Utility vs Privacy

Cookies let websites remember a visitor's choices across pages and sessions, supporting essentials like staying signed in, keeping items in a shopping cart, and tailoring content to prior behaviour. This so-called session or strictly necessary cookies tend to be viewed as helpful or at least innocuous because they stop functioning once the browser is closed or after a short period, often perceived as beneficial or, at least, unobtrusive by users.

The scope of cookie use broadened once persistent and, especially, third-party variants arrived, turning them into a linchpin of the online advertising economy[35][36][37][38]. Third-party cookies track a person from site to site, enabling advertisers and data brokers to assemble detailed profiles of browsing patterns, interests, and inferred demographics. Supporters argue that such profiling funds "free" web services by delivering ads presumed to match user preferences. Critics counter that the practice is opaque, where most people do not realize how extensively they are followed, which firms receive their data, or how those dossiers may be sold or merged with other information sets[35][36][37][38]. This lack of transparency and control lies at the heart of today's cookie-driven privacy debate, as technical studies such as research on cookie-synchronization methods continue to reveal the sophistication of the tracking infrastructure.

2.3 Cookies and Web Tracking: Methods and Privacy

The internet economy has increasingly shifted towards a data-driven model. User data, often collected through cookies and similar tracking technologies, has become a valuable commodity. This "surveillance capitalism", as some scholars term it [9][10][11], powers targeted advertising, personalized services, and data analytics, but also carries significant social implications. The aggregation of vast amounts of personal data can lead to discriminatory practices, manipulative marketing, and a chilling effect on free expression if users fear constant monitoring. The concern is not just about individual privacy but also about power imbalances between large tech companies and individual users. The irresponsible practices of organizations and website owners who allow third-

party cookies to be set without proper oversight are a direct consequence of this economic model. Research into "dark patterns"[39] - i.e., user-interface design choices that steer or manipulate users into unintended, less privacy-protective decisions (for example, by making the "accept" option more salient or the "reject" option harder to find) - demonstrates how website designs can exploit psychological biases to encourage data sharing, further exacerbating these power imbalances."[5]

In response to growing public and regulatory alarm, comprehensive data protection laws have been enacted, most notably the ePrivacy Directive and the General Data Protection Regulation (GDPR) in Europe, and the California Consumer Privacy Act (CCPA)/California Privacy Rights Act (CPRA) in the United States. These regulations mandate, among other things, transparency and user consent for the use of non-essential cookies. The introduction of these legal frameworks led to the proliferation of "cookie banners" or "consent pop-ups" on websites.

However, the implementation of these consent mechanisms has been fraught with challenges, which numerous studies, including the extensive analyses by Bollinger[2][5] and the automated dynamic analysis of cookie banner compliance from ETH Zurich[8], reveal widespread non-compliance. Issues range from banners that do not offer a genuine choice (e.g., no clear "reject all" option), the use of pre-ticked boxes, the deployment of cookies before any user interaction, to overly complex interfaces that discourage users from exercising their rights. Furthermore, "dark patterns", as investigated in Bollinger's research[6], are frequently employed to nudge users towards accepting cookies, undermining the principle of freely given consent.

The utility and privacy implications of cookies largely depend on their context, particularly whether they are set by the website, the user intentionally visits (first party) or by a different domain (third-party).

First party cookies are set by the website domain displayed in the browser's address bar and their primary purpose is usually to enhance the user's direct experience on that specific site, which include[35]:

- **Session Management**: Keeping a user logged in as they navigate different pages of a site.
- **Personalization**: Remembering user preferences, such as display settings, language choices, or recently viewed items.
- **Shopping Carts**: Tracking items added to a cart before checkout.

- **Basic Analytics**: Allowing the website owner to understand how users interact with their site (e.g., popular pages, navigation paths), often for site improvement. While this involves data collection, it is typically confined to the context of the user's interaction with that single website.

Generally, first party cookies are less controversial from a privacy perspective, as users have a direct relationship with the website setting them and often perceive their function as beneficial or necessary for the site's operation. However, even first party cookies can collect significant data, and transparency about their use is still important.

Third-party cookies are set by a domain other than the one the user is currently visiting, and this typically occurs when a website incorporates content or scripts from external services, such as:

- **Advertisements**: Ad networks embed scripts and invisible pixels (web beacons) that place cookies on the user's browser. When the user visits other sites that also use the same ad network, this third-party cookie can be read, allowing the network to track the user's browsing activity across multiple unrelated websites.
- **Social Media Widgets**: "Like" or "Share" buttons from social media platforms can set third-party cookies, enabling these platforms to track users even if they don't interact with the buttons.
- **Analytics Services**: While some analytics are first party, many popular services like Google Analytics, when not configured for anonymity, operate in a third-party context, collecting data across a vast network of sites.

The primary purpose of third-party cookies is often cross-site tracking, which is fundamental to behavioural advertising, serving ads based on inferred user interests, and building user profiles. This practice is at the heart of most online privacy concerns related to cookies and the lack of a direct relationship between the user and these third parties, coupled with the often-invisible nature of this tracking, makes it difficult for users to understand or control how their data is being collected and used.

A critical mechanism for enhancing the power of third-party tracking is cookie synchronization or cookie matching. As detailed in research like "CookieMonster: Exploring the Ecosystem of Web-based Cookie Synchronization"[40] and Google Cookie Matching[41], this process allows different advertising technology (ad-tech) companies and data brokers to link their respective cookie identifiers for the same user.

Here's a simplified explanation:

1. A user visits Website A, which has an ad from Ad Network X. Ad Network X sets a cookie (CookieX) on the user's browser.
2. The user later visits Website B, which has an ad from Ad Network Y. Ad Network Y sets its cookie (CookieY).
3. If Ad Network X and Ad Network Y have a partnership, they can initiate a cookie sync. For example, when the user visits a page with elements from both networks, one network might redirect the user's browser, often invisibly and instantaneously, to a sync endpoint of the other network. During this process, both networks can read their own cookies and record the association between CookieX and CookieY for that user.

This allows different entities to pool their data about a user, creating much richer and more comprehensive profiles than any single entity could build alone[4]. CookieMonster likely investigates the scale and complexity of these synchronization networks, revealing how identifiers are exchanged between numerous parties, often without the user's explicit knowledge or consent. This massively expands the scope of tracking and makes it even harder for users to control who has their data.

As browsers and users have become more aware of cookies and started to implement blocking or deletion mechanisms, the industry has developed alternative tracking techniques:

- **Browser Fingerprinting**: This involves collecting a wide array of information about a user's browser configuration (e.g., browser type and version, operating system, installed fonts, plugins, screen resolution, language settings, user agent string, canvas rendering, WebGL parameters). The combination of these attributes can create a unique or highly distinctive "fingerprint" that can identify a user's device even if cookies are disabled or deleted. Bollinger's research [2][6], sometimes touches upon the broader landscape of tracking, and fingerprinting is a key component of this.
- **Supercookies (e.g., HSTS Supercookies, Evercookies)**: These are tracking mechanisms that store information in unconventional places (e.g., HTTP Strict Transport Security flags, Flash Local Stored Objects, HTML5 storage) and can be difficult for users to find and delete using standard browser cookie controls. They are often designed to respawn deleted cookies.
- **Tracking via Login IDs/Email Hashes**: When users log into services with consistent identifiers like an email address, often in hashed form, these

identifiers can be used for cross-device tracking and linking online behaviour with offline data.

This techniques creates a unique, persistent identifier by collecting a wide array of data points, or "signals," from a user's device and browser configuration. Because the combination of these attributes creates a high-entropy identifier, it can distinguish one user from millions of others, even if they delete cookies or use incognito mode. Research has confirmed that this method is actively being used for real-world ad tracking and targeting.

Cookies are not monolithic, and they have several attributes that control their behaviour and have privacy implications:

- **Domain and Path**: These attributes specify which website(s) and which parts of a website the cookie should be sent to. Third-party cookies have a domain attribute different from the site being visited.
- **Expires / Max-Age**: This determines the cookie's lifespan. Session cookies expire when the browser closes. Persistent cookies can last for days, months, or even years, enabling long-term tracking.
- **HttpOnly**: If set, this attribute prevents client-side scripts, like JavaScript, from accessing the cookie. This is primarily a security measure against cross-site scripting (XSS) attacks but can also make certain types of tracking slightly harder for malicious scripts.
- **Secure**: This attribute ensures the cookie is only transmitted over HTTPS, protecting it from interception during transit.
- **SameSite (Lax, Strict, None)**: This attribute is a browser security feature that gives website owners control over when cookies are sent with requests that originate from other domains. Its primary purpose is to mitigate Cross-Site Request Forgery (CSRF) attacks, which trick an authenticated user's browser into performing unwanted actions on a trusted site by leveraging the user's session cookies. By explicitly defining how cookies should be handled in cross-site contexts, it’s also enhanced user privacy and aligns with modern browser security policies.

  Strict is the most restrictive setting and it will only be sent if the request originates from the exact same site that issued the cookie. It is blocked in all cross-site contexts, even when a user clicks a regular link to navigate to the site from an external page (e.g., an email or another website). While this offers the strongest protection against CSRF, it can harm user experience by, for

example, requiring a user to log in again after following a link to the site. This setting is most appropriate for high-security applications, such as banking websites, that do not intend for transactional pages to be accessed from external links.

Lax provides a balance between security and usability and is the default value in many modern browsers if no attribute is specified and the cookie is not sent on cross-site sub-resource requests, like those for images or scripts, but is sent when a user performs a top-level navigation, such as clicking a link to go to another site. This allows a user to remain logged in when navigating to a site from an external source while still blocking most CSRF-prone requests, such as those made using the POST method.

None effectively disables SameSite restrictions, allowing the cookie to be sent with all cross-site requests. This is the setting required for third-party cookies to function correctly, enabling uses like embedded content, cross-site analytics, and single sign-on (SSO) systems. For security, browsers require that any cookie with that setting must also have the Secure attribute, ensuring it is only transmitted over an encrypted HTTPS connection.

The details of these attributes are often opaque to average users but are fundamental to how cookies operate and how they can be used or abused for tracking. An application aiming for detailed information disclosure, would ideally provide users with insights into these attributes for the cookies a website uses. The ETH Zurich cookie proposal[3] may also touch upon the technical specifications relevant for analysis.

The evolution from simple state managers to complex cogs in a vast tracking machinery underpins the regulatory and user concerns. The technical mechanisms, especially those facilitating cross-site tracking and data aggregation as explored by research like[4], create a challenging environment for achieving genuine user privacy and control.

2.4 Cookie Governance: Regulation and Consent

The ePrivacy Directive aimed to protect fundamental rights and freedoms, particularly the right to privacy, with respect to the processing of personal data in the electronic communication sector. While it covered various aspects like confidentiality of communications and unsolicited marketing, Article 5(3) became particularly well-known for its provisions on cookies.

The original 2002 version of Article 5(3)[12] stated that storing information or gaining access to information stored in the terminal equipment of a subscriber or user was only

allowed on condition that the subscriber or user concerned was provided with clear and comprehensive information, inter alia, about captivated the purposes of the processing, and was offered the right to refuse such processing. This was often interpreted as an opt-out regime.

The 2009 amendment significantly changed Article 5(3)[13] and it stipulated that the storing of information, or the gaining of access to information already stored, in the terminal equipment of a subscriber or user is only allowed on condition that the subscriber or user concerned has given his or her consent, having been provided with clear and comprehensive information, in accordance with Directive 95/46/EC[14] (the Data Protection Directive, GDPR's predecessor), inter alia, about the purposes of the processing.

This shift towards requiring consent, often interpreted as opt-in, though implementations varied, for most cookies except those "strictly necessary" for a service requested by the user, led to the first wave of cookie banners appearing on websites targeting EU users. The communication and understanding of EU regulations, potentially touched upon in broader contexts like Stoica et al.[7], became crucial for website operators globally.

Despite its intentions, the implementation of the ePrivacy Directive was hindered by several significant challenges. A primary issue was inconsistent enforcement, which arose because, as a directive, it had to be transposed into national law by each EU member state, leading to different interpretations and applications across the Union. This legal fragmentation contributed to problems with how consent was obtained.

The quality of consent mechanisms was often poor, with many early cookie banners relying on implied agreement through notices like "by continuing to use this site, you accept cookies", frequently without offering a clear way to refuse. The subsequent proliferation of these often-rudimentary banners led to widespread banner fatigue, causing users to click accept indiscriminately without understanding the consequences. Underlying these practical flaws was the fact that the directive's standard for consent was tied to the older Data Protection Directive 95/46/EC, which was far less stringent than the definition later established by the GDPR. These limitations, coupled with the evolving technological landscape and the broader need for a modernized data protection framework, paved the way for the General Data Protection Regulation. The ePrivacy Directive is still in force, but its provisions on consent for cookies are now read in conjunction with the stricter definition of consent provided by the GDPR.

The GDPR, which came into effect on May 25, 2018, represents the most significant overhaul of data protection law in over two decades. While not exclusively about cookies, its principles and requirements have profound implications for how cookie consent is obtained and managed.

GDPR is built on core principles such as lawfulness, fairness and transparency; purpose limitation; data minimization; accuracy; storage limitation; integrity and confidentiality; and accountability. For cookies that process personal data, which many do, especially tracking cookies, these principles fully apply.

GDPR (Article 4(1))[15] defines personal data broadly as "any information relating to an identified or identifiable natural person (data subject)." Recital 30 explicitly states that online identifiers, such as cookie identifiers, can be considered personal data if they can be used to single out individuals. This brought most advertising and analytics cookies firmly under GDPR's scope.

GDPR (Article 4(11))[15] defines consent as "any freely given, specific, informed and unambiguous indication of the data subject's wishes by which he or she, by a statement or by a clear affirmative action, signifies agreement to the processing of personal data relating to him or her."

To address the weaknesses of previous regulations and ensure that user consent is a genuine and meaningful choice, the GDPR established a far more rigorous definition, where consent is only considered valid if it is simultaneously:

- **Freely given**: Consent is not valid if the user feels compelled to consent or will suffer negative consequences for not consenting. Bundling consent for different processing purposes or making access to services conditional on consent for non-essential cookies is problematic.
- **Specific**: Consent must be obtained for specific processing purposes. Vague or blanket consent is not acceptable. Users should be able to consent to some purposes but not others (granularity).
- **Informed**: Users must be provided with clear, concise, and easily understandable information before consent is given. This includes the identity of the controller, the purposes of processing, the types of data collected, the existence of the right to withdraw consent, and information about data transfers. The "detailed information disclosure" functionality of the proposed thesis application is critical here.

- **Unambiguous & Affirmative Action**: Consent requires a clear affirmative act, such as ticking an unticked opt-in box or clicking an "Accept" button. Pre-ticked boxes or silence/inactivity do not constitute valid consent. This is a key area where research by Bollinger[2][6] and the ETH Zurich team[8] finds frequent non-compliance.

Besides the definitions, Article 7[15] lays down strict conditions for consent, including the controller's obligation to demonstrate that consent was obtained and the user's right to withdraw consent at any time as easily as it was given.

Each EU member state has one or more DPAs responsible for monitoring and enforcing GDPR. DPAs[16] can issue warnings, reprimands, orders, and impose significant fines, up to €20 million or 4% of global annual turnover, whichever is higher, for non-compliance. Several DPAs have issued specific guidance on cookie consent and have taken enforcement actions against non-compliant websites.

GDPR provides several legal bases for processing personal data (Article 6)[15]. While consent is the most relevant for tracking cookies, legitimate interest is another. Some website operators have attempted to rely on legitimate interest for deploying analytics or even advertising cookies. However, European DPAs and the European Data Protection Board (EDPB) have generally taken a restrictive view, stating that for cookies that track users or involve intrusive profiling, especially for advertising, consent is typically the only appropriate legal basis because the processing is unlikely to meet the balancing test required for legitimate interest (i.e., the interests of the controller are overridden by the fundamental rights and freedoms of the data subject).

GDPR has extraterritorial reach (Article 3)[15] and it applies to the processing of personal data of data subjects in the EU by a controller or processor not established in the EU, where the processing activities are related to the offering of goods or services to such data subjects or the monitoring of their behaviour as far as their behaviour takes place within the EU. This means many international websites with EU visitors must comply with GDPR's cookie consent rules.

The study "Dark Patterns after the GDPR: Vanishing, Thriving, or Hybridized?"[5] directly examines how website behaviours have adapted or failed to adapt in response to GDPR.

While the EU took the lead with GDPR, California enacted the CCPA, effective January 1, 2020, significantly enhancing privacy rights for California consumers. The CCPA was subsequently amended and expanded by the California Privacy Rights Act (CPRA), with most provisions becoming effective on January 1, 2023.

The California Consumer Privacy Act (CCPA), significantly expanded by the California Privacy Rights Act (CPRA)[17], grants California consumers a robust set of rights to control their personal information. These rights give individuals transparency and control over their data, starting with the Right to Know what personal information a business collects, its sources, the purposes for its collection, and the categories of third parties with whom it is shared.

Building on this, consumers have the Right to Delete their personal information held by a business, subject to certain exceptions, and the Right to Correct inaccurate information, a right specifically added by the CPRA.

Crucially for online tracking and advertising, the law provides a powerful Right to Opt-Out. The original CCPA gave consumers the right to opt-out of the sale of their data, but the CPRA broadened this to include the sharing of personal information, a change aimed directly at cross-context behavioural advertising. This right is typically operationalized through a "Do Not Sell or Share My Personal Information" link on a company's website.

The CPRA also introduced a new category for particularly vulnerable data, establishing the Right to Limit the Use and Disclosure of Sensitive Personal Information. To ensure consumers can exercise these rights freely, the law includes a Right to Non-Discrimination, which prohibits businesses from penalizing users for asserting their privacy rights.

Under CCPA[17], "sale" is broadly defined and includes the exchange of personal information for monetary or other valuable consideration. Many interpretations conclude that the use of third-party advertising cookies, where website owners receive advertising services or revenue in exchange for allowing trackers to collect user data, constitutes a "sale." The CPRA added the concept of "sharing," defined as including the transfer of personal information to a third party for cross-context behavioural advertising, whether for monetary or other valuable consideration. This clarification further solidifies that many ad-tech cookie practices fall under the opt-out right.

A key difference between GDPR and CCPA/CPRA regarding cookies is the consent model. GDPR generally requires an opt-in approach for non-essential cookies, users must affirmatively agree. CCPA/CPRA, for the "sale" or "sharing" of data via cookies, primarily establishes an opt-out system, data collection can occur until the user exercises their right to opt-out. However, for minors, CCPA has opt-in requirements and despite these differences, both frameworks aim for greater transparency and user control. The

cookie proposal from ETH Zurich[3] might consider such comparative aspects when discussing the design of compliant systems.

The CPRA established the California Privacy Protection Agency (CPPA) to implement and enforce California's privacy laws, taking over from the California Attorney General. This agency has rulemaking authority and can conduct investigations and impose administrative fines.

The existence of multiple, sometimes differing, regulatory frameworks create significant harmonization challenges for businesses operating globally. The EU itself is working on an ePrivacy Regulation, intended to replace the ePrivacy Directive and sit alongside GDPR as lex specialis for electronic communications data, including more specific rules for cookies and tracking technologies. However, its adoption has been long-delayed due to disagreements among member states on key provisions, such as the balance between privacy and innovation, and how consent should be managed (e.g., via browser settings vs. website banners).

The ideal scenario for many would be a globally interoperable privacy framework, but this remains elusive. In the interim, tools and approaches that can help navigate the existing complexities are crucial.

Despite the clear legal mandates established by regulations like the GDPR and ePrivacy Directive, and the increasing public awareness of online privacy issues, the practical reality of cookie consent management remains fraught with challenges. Website operators face difficulties in implementing compliant and user-friendly systems, while users often encounter confusing, manipulative, or simply ineffective consent mechanisms.

A significant body of research consistently demonstrates that a large proportion of websites, including those utilizing Consent Management Platforms (CMPs), fail to fully comply with the stringent requirements of modern data protection laws. This non-compliance undermines the very purpose of these regulations, leaving users vulnerable to unwanted tracking and data processing.

Numerous academic studies and technical audits have systematically investigated the state of cookie consent compliance across the web, painting a concerning picture, like the extensive research by Dino Bollinger, as presented in his dissertation[2] and related publications such as the USENIX Security paper[6], provides robust empirical evidence of widespread issues. His work often involves large-scale automated analysis of websites and CMP implementations, revealing high rates of non-compliance with GDPR

principles. For instance, these studies may detail the percentage of websites that drop cookies before consent, fail to offer a clear "reject" option, or use pre-ticked boxes.

The project on "Automated Dynamic Analysis of Cookie Banner Compliance" from ETH Zurich[8] further substantiates these findings. By developing tools to automatically interact with cookie banners and observe the resulting cookie-setting behaviour, such research can systematically identify discrepancies between stated policies, user choices, and actual website actions. This reference likely details methodologies for such automated testing and the prevalence of different types of violations uncovered.

These studies often find that even when consent banners are present, their configuration or the underlying logic does not correctly capture or respect user preferences. For example, a user might click "Reject All," but tracking cookies are still placed, or the consent signal is not properly communicated to third-party vendors integrated into the site.

The observed forms of non-compliance with cookie regulations are varied, impacting every stage of the consent process. A fundamental violation occurs when websites place non-essential cookies, especially for tracking and advertising, on a user's device before any valid consent is obtained. This practice, often happening immediately upon page load, renders any subsequent consent interaction moot for those initial cookies.

Many consent banners also fail to provide a genuine choice by lacking a clear and equally prominent option to refuse non-essential cookies. While an "Accept" button is highly visible, rejecting often requires navigating through complex sub-menus, a design that contradicts the GDPR's mandate that refusing consent must be as easy as giving it. This manipulation of user choice is frequently compounded using pre-ticked opt-in boxes, a dark pattern designed to nudge users into acceptance that violates the GDPR's requirement for an unambiguous and affirmative action. Similarly, outdated banners that rely on implied consent using phrases like "by continuing to browse, you accept cookies" are no longer compliant because they lack a clear affirmative act from the user.

Beyond flawed design, direct technical failures are also common and, in some cases, "Reject" or "Save Settings" buttons in preference centers may not work correctly, meaning user choices are not actually stored or respected. A more subtle but equally serious issue is incorrect consent signal propagation, where a user's choice, even if correctly recorded by the website's consent management platform, is not properly communicated to or respected by all third-party vendors operating on the site.

Many consent notices are deficient due to insufficient information. They often fail to provide the clear, comprehensive, and easily accessible details about the types of cookies used, their purposes, duration, and the third parties involved, as required by the GDPR's informed consent criteria.

Despite the authority granted to DPAs, enforcing cookie consent rules across millions of websites remains a monumental task. A primary obstacle is significant resource constraints, as DPAs often lack the necessary funding and personnel to proactively monitor and investigate the vast digital landscape. This scarcity of resources forces them to operate reactively, relying heavily on complaints filed by individuals or consumer organizations, which means non-compliance can persist undetected for long periods.

Compounding this issue is the sheer technical complexity of the work. Assessing compliance requires deep technical expertise to analyse intricate cookie behaviours, consent management platform configurations, and backend data flows, a skillset that is challenging to apply at scale and while automated analysis methodologies aim to assist in this, their broad adoption by all DPAs is an ongoing process.

Furthermore, enforcement is often complicated by cross-border issues. Even with the GDPR's extraterritorial scope, pursuing action against a non-compliant website operator can be difficult when the company is based outside the jurisdiction of the DPA handling a complaint.

This enforcement gap means that non-compliant practices can persist, highlighting the need for tools that can empower users and organizations to assess and demand compliance.

2.5 Consent UX and Transparency

Beyond outright non-compliance, the design and implementation of cookie consent mechanisms frequently suffer from poor usability, leading to user frustration, misunderstanding, and an inability to exercise meaningful control.

The constant bombardment of cookie banners as users navigate the web leads to a well-documented phenomenon known as banner fatigue or consent fatigue. Overwhelmed by these persistent interruptions, many users resort to indiscriminately clicking "Accept All" or the most prominent positive option simply to dismiss the banner and access content, an action that undermines the entire goal of informed consent.

This behaviour is a significant contributor to the "privacy paradox", a disconnect observed where users express strong privacy concerns in principle but then engage in

actions that seem to contradict those beliefs. Beyond compromising meaningful consent, the intrusive and often poorly designed banners degrade the overall website experience, leading to user annoyance and potentially causing them to abandon a site altogether.

A significant challenge to meaningful consent is the use of "dark patterns", user interface design choices deliberately crafted to trick or nudge users into actions they might not otherwise take, often benefiting the service provider at the user's expense. The study "Dark Patterns after the GDPR: Vanishing, Thriving, or Hybridized?"[5] specifically investigates how these manipulative designs are employed within cookie consent banners.

These tactics manifest in several ways. A common example is obstructed choice or forced action, where rejecting cookies is made significantly harder than accepting them, such as by pairing a bright, prominent "Accept" button with a small, grayed-out "Settings" link. Designers also use misleading wording to imply that rejecting consent will unnecessarily degrade the service or deploy confirm shaming to guilt users with phrases like, "No, I don't want amazing offers."

Other manipulative techniques include the pre-selection of opt-in boxes for non-essential cookies and interface interference, where banners cover a large portion of the screen and block website content until a choice is made. As the referenced study suggests with its title, these manipulative designs have not vanished but have instead adapted or hybridized in response to regulation, indicating that legal frameworks alone have not been sufficient to eradicate them.

A core requirement for valid consent is that it must be informed, yet many cookie notices fail to provide the clear, concise, and easily understandable information necessary to achieve this. Often, privacy policies and cookie descriptions are composed of lengthy, dense legal jargon that is inaccessible to the average person. This issue is frequently compounded by information overload, as some banners present users with excessively long lists of cookies or vendors without proper categorization or explanation, a practice that overwhelms rather than enlightens.

Furthermore, the stated purposes for data collection are often vague and unhelpful, using broad terms like "to improve user experience" or "for operational purposes" that provide no meaningful insight into how data will be used.

Like any other web interface element, cookie banners must be accessible to users with disabilities, adhering to established standards like the Web Content Accessibility Guidelines (WCAG). This requires that users who cannot use a mouse are able to

navigate and interact with all banner options using only a keyboard, and that the banner's content is correctly interpreted by screen readers for visually impaired users.

Furthermore, the design must ensure legibility through sufficient text size and adequate contrast against the background. Ultimately, failures in accessibility can effectively prevent an entire segment of users from exercising their consent choices, rendering the process invalid for them.

True user control hinges on transparency regarding data practices and the ability to make granular choices. These are areas where current cookie consent mechanisms frequently fall short.

While cookie banners may state that cookies are used for general purposes like "advertising" or "analytics," they often fail to provide the detailed explanations necessary for users to give truly informed consent. They frequently lack transparency about the specific operations involved, such as what analytics entails, which precise data points are collected, and for what analyses they will be used.

Furthermore, the identities of the specific third-party companies that will receive data are often obscured providing a link to a policy with hundreds of potential vendors does not constitute clear disclosure. A critical piece of information that is also frequently omitted is whether the data collected via these cookies will be combined with other information the company or third parties already hold about the user. This collective lack of detailed transparency makes it nearly impossible for users to make a genuinely informed decision.

A key principle of the GDPR is granular consent, which requires that users can consent to some processing purposes, like essential site functionality, while rejecting others such as personalized advertising. However, this principle is frequently violated in practice and any banners still present a binary, "all-or-nothing" choice, forcing users to either "Accept All" or "Reject All," assuming a reject option is even clearly available.

Even when websites do offer granular controls, they are often buried within complex, multi-layered preference centers that are difficult to navigate and understand. The sliders or toggles provided in these menus may not clearly explain the implications of each choice, preventing an informed decision. Furthermore, the concept of true granularity, which would allow users to choose which specific third-party vendors are acceptable, is a level of control that is almost never offered, further highlighting the gap between regulatory requirements and real-world implementation.

Consent is not a one-time event but an ongoing process that must be managed throughout its lifecycle. A key aspect of this is the duration of consent, which is often unclear to the user, although best practices suggest it should be re-obtained periodically rather than being considered permanent.

Equally critical is the ease of withdrawal, where GDPR mandates that withdrawing consent must be as easy as giving it, yet many websites make it difficult for users to find the necessary settings to change or revoke their choices after the initial interaction.

Finally, the lifecycle includes a significant administrative burden on controllers, who must maintain robust record-keeping. They are required to demonstrate that valid consent was obtained for each user, which involves logging the specific choices made and the version of the privacy information that was provided at the time of consent.

For websites with an international audience, data collected via cookies is often transferred outside the user's jurisdiction. When this data moves outside the EU/EEA, it is subject to the GDPR's strict rules, which require legal safeguards like adequacy decisions or Standard Contractual Clauses. Despite these requirements, users are rarely informed explicitly and clearly when their data will be transferred internationally or what protections are in place.

This critical transparency gap is particularly important considering landmark legal rulings like Schrems II[18], which invalidated the EU-US Privacy Shield and placed transatlantic data flows under much greater scrutiny, making robust compliance verification more essential than ever.

Even with well-designed interfaces and clear information, technical challenges can prevent user consent choices from being effectively enforced. Cookie management involves both client-side (browser) and server-side actions.

- **Client-Side Scripting**: Many cookies are set and managed by JavaScript running in the user's browser ensuring that these scripts correctly interpret consent signals and only fire trackers when authorized can be complex, especially with numerous third-party scripts on a page.
- **Server-Side Integration**: User preferences might also need to be communicated to and respected by server-side applications.
- **Tag Management Systems**: Websites often use TMS to manage the deployment of various tracking scripts tags. Configuring these TMS solutions to be fully consent-aware requires careful implementation. Incorrect configurations are a common source of non-compliance.

The immense complexity of the online advertising ecosystem, with its web of intermediaries like ad exchanges, data brokers, and various platforms, presents a formidable challenge to managing user consent. To standardize this process, frameworks like the IAB Europe's Transparency and Consent Framework (TCF)[19] were developed to provide a common method for capturing user choices and communicating them throughout the ad-tech chain via a consent string.

However, the effectiveness and compliance of the TCF are highly contested. The framework has faced significant legal challenges, notably a ruling by the Belgian Data Protection Authority[2][6] which found that TCF consent strings constitute personal data and that aspects of the framework violate GDPR. Beyond legal scrutiny, ensuring that every vendor in the long ad-tech chain correctly interprets and adheres to a user's consent signal is a major technical and governance challenge, with research likely auditing the TCF's practical implementation and shortcomings.

This problem is further exacerbated by the process of cookie synchronization, which can effectively bypass a user's choice. As detailed in the "CookieMonster"[4] research, this technique allows a cookie ID for which consent was given to be matched with another ID from a party for whom consent was not given, undermining the entire consent mechanism.

Modern browsers are increasingly taking on the role of privacy gatekeepers through native features like Safari's Intelligent Tracking Prevention, Firefox's Enhanced Tracking Protection, and Chrome's upcoming deprecation of third-party cookies. This shift introduces new layers of complexity, as these browser-level protections can interact unpredictably with website-based CMPs or even block legitimate first-party functionality. These dynamic fuels an ongoing debate over where consent should be managed, at the browser level, through signals like Global Privacy Control, or on a site-by-site basis. The proposed application acknowledges this critical interplay through its features for testing browser settings and validating banner reappearance, recognizing that consent operates at the intersection of website design and browser controls.

Addressing these multifaceted challenges, ranging from blatant non-compliance and manipulative designs to deep technical complexities, requires a concerted effort.

2.6 Consent Tools and Platforms

Consent Management Platforms (CMPs) have emerged as the most common commercial solution adopted by website publishers to address the requirements of cookie consent regulations, particularly GDPR.

A CMP is a software solution that helps website operators and businesses systematically collect, store, and manage user consent for data processing, which is essential for complying with privacy regulations like the GDPR. These platforms act as a centralized hub, automating many of the most labour-intensive compliance tasks.

A CMP's first and most visible function is to display a customizable cookie banner or notice when a user first visits a site. This initial interface informs visitors about the website's data collection practices and provides clear options to provide consent. Through this banner, the platform facilitates the collection of user consent, offering interfaces with buttons, toggles, or checkboxes that allow users to accept, reject, or customize their preferences for different categories of cookies, such as those for analytics or marketing.

Once a user makes a choice, the CMP securely stores a record of that consent in a centralized repository, creating an audit-ready log that includes a timestamp and other contextual information. This record-keeping is critical for website owners to demonstrate compliance with their legal obligations under the GDPR's accountability principle.

Beyond just recording the choice, the platform's most crucial technical function is to signal the user's consent status to other technologies operating on the site. The CMP dynamically blocks or allows scripts and third-party tags to fire based on the specific permissions the user has granted, ensuring that data is only processed with valid consent.

Finally, consent management is an ongoing process. A comprehensive CMP provides users with access to a persistent preference center, often accessible via a link in the website's footer or privacy policy. This allows users to easily review and change their consent settings at any time, fulfilling the legal requirement that withdrawing consent must be as simple as giving it.

CMPs aim to simplify the complex task of consent management for website owners, offering pre-built templates and integrations. Many CMPs also try to align with industry frameworks like the IAB TCF.

Despite their widespread adoption, the effectiveness and compliance of CMPs have been extensively questioned by academic research, which reveals a significant gap between perceived and actual compliance.

Systematic, large-scale audits, such as those conducted by Bollinger[2][5] and researchers at ETH Zurich[8], consistently uncover widespread issues. One landmark

study found potential GDPR violations in a surprising 94.7% of analysed websites. These violations are not minor; research shows that over 70% of websites activate cookies before the user interacts with the banner, and more than 20% deploy cookies that a user has specifically rejected. Many CMP implementations still exhibit dark patterns that make rejecting consent harder than accepting it, and the information provided in the interfaces can be incomplete or misleading.

Crucially, these failures are not always the fault of the individual website owner. Research suggests that manipulative dark patterns often result from the default configurations of the CMPs themselves, implicating even major, established providers in enabling these practices. Furthermore, there can be a critical breakdown between the user's choice and the technology's execution. A technical analysis found that 37% of CMP consent signals routinely fail to be enforced, meaning the user's preferences are not communicated or respected by downstream technologies.

These findings demonstrate that simply installing a CMP does not guarantee compliance. The specific configuration, the integrity of the provider, and how the platform integrates with a website's other technologies are all critical factors.

The CMP market includes a range of providers, from large, specialized companies to smaller players, and some website platforms offer built-in CMP functionalities. Business models typically involve subscription fees based on website traffic or features. The competition in this market can lead to innovation but also potentially to CMPs prioritizing features that maximize consent rates for publishers, sometimes at the expense of genuine user choice, rather than strictly adhering to the spirit of the law.

Parallel to website-centric solutions like CMPs, a variety of tools operating at the browser level have been developed to give users more direct control over tracking and to assist researchers in analysing web privacy.

Millions of users are actively enhancing their online privacy by employing a diverse array of browser extensions and native browser features. A primary line of defense comes from popular ad blockers like AdBlock Plus and uBlock Origin. While their main purpose is to remove advertisements, they often block third-party tracking cookies and scripts as a side effect, since ad delivery and user tracking are so closely intertwined.

Beyond ad blocking, a dedicated category of anti-tracking extensions like Privacy Badger and Ghostery exists specifically to combat online surveillance. These tools are designed to identify and neutralize various tracking methods, from cookies to more advanced fingerprinting scripts.

These tools can provide a significant layer of protection, but their effectiveness can vary, and they can sometimes break website functionality if they are too aggressive or if websites retaliate against their use.

While users have access to privacy tools, a more technical suite exists for developers and researchers. Browsers come with built-in developer tools that allow for the inspection and deletion of cookies, but for more in-depth analysis, specialized solutions are required. These range from advanced cookie inspection extensions to comprehensive web privacy measurement platforms like OpenWPM, which researchers use to conduct large-scale, automated website crawls to measure tracking and fingerprinting. At an even deeper level, network traffic analyzers like Wireshark and browser network inspectors allow for the examination of raw HTTP requests and responses, revealing the precise moment tracking cookies are set. While these tools are invaluable for academic research and debugging, they generally require a high degree of technical expertise to be used effectively.

This steep learning curve highlights the broader limitations of relying on browser-based privacy tools. Many average users are unaware of their existence or lack the technical knowledge to install and configure them optimally. Furthermore, their effectiveness is challenged by a constant arms race between anti-tracking tools and the surveillance industry, which constantly develops new methods to evade detection. Tools that rely on blocklists, for instance, require continuous updates to keep pace with new tracking domains. This aggressive blocking can also have a negative side effect, as it can interfere with legitimate website functionality, leading frustrated users to disable the very tools meant to protect them.

While existing solutions offer certain benefits, they also come with inherent limitations that prevent them from fully addressing the compliance deficit.

Consent Management Platforms, for instance, are primarily tools, and their effectiveness is heavily dependent on correct configuration by the website owner. Misconfigurations can easily lead to non-compliance despite the platform's presence. For users and external auditors, the internal workings of a CMP can be a black box, making it difficult to verify how consent is enforced. This opacity is concerning, as some CMPs may even have a vendor bias, employing subtle dark patterns or default settings designed to maximize consent rates rather than align with the spirit of privacy regulations. Ultimately, their primary function is to collect consent, and they do not inherently offer robust, independent verification that the consent is being technically respected by all elements on the page.

Similarly, browser-based tools and extensions like ad blockers and anti-trackers have a different focus. Their main purpose is to block trackers, not to analyse or verify the website's consent mechanism itself. An aggressive ad blocker treats a compliant and a non-compliant website the same, providing little insight into the consent logic, such as whether a website is correctly storing a user's choices. While empowering for individuals, these tools are user-side protections, not auditor-side solutions designed for systematically checking the nuances of a consent interface.

Finally, academic research tools and large-scale scanners are generally not suitable for widespread use. Their complexity and resource requirements make them inaccessible to individual users or smaller organizations. Their strength lies in identifying broad trends across thousands of websites, not in providing the granular, step-by-step diagnostic capabilities needed for auditing a specific site. Furthermore, these large scans offer only a snapshot in time, whereas a website's compliance status can change quickly, necessitating more frequent and targeted checks that an accessible tool could provide.

2.7 Compliance Evidence and Automation

A rich and growing body of academic research has significantly contributed to our understanding of the cookie ecosystem, driving the development of better tools and policy recommendations by systematically uncovering compliance issues and analyzing user interactions with consent mechanisms.

To assess compliance at scale, academics have pioneered sophisticated methodologies. Central to this effort is the use of automated systems, or bots, that can crawl thousands of websites, identify cookie banners, and dynamically interact with consent options to observe the website's actual cookie-setting behavior. Work from institutions like ETH Zurich[8] and researchers like Bollinger[2][6] exemplifies this approach, which often combines static analysis of website code with dynamic analysis of its runtime behavior. These studies yield detailed taxonomies of common compliance violations and dark patterns, and projects like "CookieMonster"[4] have developed specific methods to map complex data-sharing relationships, such as cookie synchronization.

Parallel to this technical auditing, another significant stream of research focuses on the user's experience. Through usability studies employing eye-tracking and think-aloud protocols, researchers analyze how users comprehend banner designs and make decisions. A major focus is on identifying dark patterns in consent interfaces and measuring their manipulative effect, which consistently shows that these designs coerce

users into higher rates of consent. This user-focused research also delves into people's fundamental mental models, or common misunderstandings, about what cookies are, how tracking works, and what their consent choices mean.

Beyond technical compliance and user interaction, academics also study the broader socio-economic impacts of cookie regulations. This includes analyzing the effects of stricter consent rules on publishers' advertising revenues, exploring the economic valuation of privacy, and examining the profound power imbalances between large technology platforms and individual users within the data ecosystem.

Recognizing the limitations of current banner-by-banner consent and the burden it places on users, several proposals aim for more automated or user-centric approaches to consent management.

- **Do Not Track Signal**: An early attempt was the "Do Not Track" HTTP header, where users could express a general preference not to be tracked. However, DNT largely failed due to a lack of industry adoption and no legal enforceability.
- **Global Privacy Control**: GPC is a more recent initiative where users can express their privacy preferences (e.g., to opt-out of the sale/sharing of their data under CCPA/CPRA) through a signal sent by their browser or extension. Some jurisdictions, like California, are beginning to recognize GPC as a valid method for users to exercise their rights.
- **Browser-Level Settings**: The idea is that users could set their privacy preferences once in their browser, and websites would automatically respect these signals, reducing the need for individual banner interactions. The ePrivacy Regulation proposal in the EU has also debated the role of browser settings in managing consent.

More advanced proposals envision using Artificial Intelligence, particularly Large Language Models (LLMs), to fundamentally change how users manage online privacy. These AI systems could be trained to automatically parse and comprehend the complex; legalistic language found in privacy policies and cookie banners. Based on that understanding, an intelligent agent could then compare a website's data practices against a user's predefined privacy profile, allowing it to either automatically configure the consent choices on their behalf or provide a simple, actionable recommendation. Furthermore, this technology could be specifically trained to detect deceptive language and the subtle dark patterns embedded in consent interfaces, offering a proactive layer of protection against manipulation.

These approaches are still largely in the research and development phase but hold promise for reducing user burden and making consent more meaningful.

An AI-driven consent automator, aligns with the user-centric spirit by aiming to provide users (and auditors/developers) with clearer insights into how specific websites are handling consent, thus empowering more informed decisions and actions. It serves as a verification tool that can complement both existing CMP-based approaches and future automated consent systems.

Despite years of regulation and the proliferation of consent tools, a persistent compliance deficit remains, where widespread non-compliance and a lack of genuine user control are the norm. This gap between regulatory intent and real-world practice is a recurring theme in academic research, including large-scale analyses that reveal how even websites using CMPs can be flawed. This situation highlights a clear need for reliable, independent verification tools that can audit whether a website's stated consent practices match its actual technical behaviour.

Current compliance checks often stop at the surface, focusing only on the initial appearance of a cookie banner. However, true compliance requires deeper scrutiny into the functionality of second-layer preference centers, the accuracy of how consent is stored, and whether a user's choices prevent cookies from being deployed. There is a specific need for tools that can test if the granular consent mandated by regulations is implemented effectively, rather than being presented through obtuse interfaces or all-or-nothing choices.

Furthermore, a significant gap exists in providing users with clear and specific information. Users often lack transparency about what specific cookies are active, their precise purposes, their duration, and, critically, whether their data is being transferred outside their legal jurisdiction, a key issue the proposed thesis solution aims to address by analysing cross-border data flows.

Finally, there is a need to bridge the two primary ways users manage their privacy: website banners and their own browser settings. It is essential to understand and verify the interplay between these two levels of control. This points to the overarching need for user-centric tools that are actionable. While powerful research tools exist, they are often inaccessible to average users or even less technical website auditors, leaving a critical need for solutions that provide clear, understandable, and actionable insights into a website's true compliance status.

The cookie proposal from ETH Zurich[3] likely identified some of these early needs, while research into dark patterns[5] and cookie synchronization[4] provides crucial context for the types of sophisticated issues that current solutions may not adequately address and that the proposed application aims to tackle.

2.8 Recapitulation of Key Findings

Cookies have evolved from simple session management tools to sophisticated instruments for cross-site tracking and user profiling, largely driven by the online advertising economy. This evolution, particularly the rise of third-party cookies and advanced tracking techniques like cookie synchronization[4], is at the heart of privacy concerns.

Regulations like the EU's GDPR and ePrivacy Directive, and California's CCPA/CPRA, have established stringent requirements for transparency and consent. However, their practical implementation is widely inconsistent, with frequent non-compliance documented by numerous studies[2][6][8].

A significant portion of websites, even those using Consent Management Platforms, exhibit various forms of non-compliance, including lack of genuine choice, pre-ticked boxes, cookies set before consent, and the use of dark patterns[5] designed to manipulate user choices.

Many cookie consent interfaces suffer from poor usability, banner fatigue, unclear information, and a lack of true granularity, hindering users' ability to make informed and meaningful decisions about their data.

While CMPs offer tools for consent collection and browser extensions provide user-side blocking, significant gaps remain in providing accessible, independent, and comprehensive verification of a website's actual compliance with both the letter and spirit of privacy laws. Existing research tools, while powerful, are often not geared towards individual website audits or use by less technical stakeholders.

The deficiencies highlighted throughout this review, underscore an urgent need for more robust, user-centric, and integrated solutions. The current environment often places an undue burden on users to navigate complex and sometimes deceptive interfaces, while website owners may lack accessible tools to accurately assess and rectify their own compliance shortcomings. There is a clear demand for mechanisms that can effectively bridge the "gap between legal requirements and practical implementation," fostering a more equitable and trustworthy relationship between users and website operators.

2.9 Thesis Contribution

The application proposed in this Master Thesis is conceived to directly address these persistent gaps and unmet needs. By focusing on the systematic verification of key functional elements of the cookie consent lifecycle, the proposed solution offers a distinct contribution:

- It provides an integrated approach to examining initial banner interactions, the usability and functionality of cookie configuration pages, the accuracy of consent state storage, the identification of potential cross-border data transfers linked to cookies, and the interplay with browser-level cookie management.
- It aims to deliver actionable insights by not just identifying problems but by pinpointing specific failures in the consent mechanism's design and technical implementation (e.g., non-functional buttons, incorrect consent value storage, misleading overlay behaviour).
- It seeks to enhance transparency for users and auditors by providing a clearer view of a website's cookie practices and consent handling, moving beyond superficial checks.
- It endeavours to empower users and support compliance efforts by offering a tool that can be used to assess whether websites are genuinely respecting privacy choices and adhering to regulatory mandates.

While not a panacea for all challenges in the digital privacy sphere, the proposed application represents a pragmatic step towards improving accountability and fostering better practices in cookie management. It complements existing research, such as the foundational work[2][3][6][8] by translating analytical insights into a focused diagnostic tool.

3 Design

Security by Design (SbD) represents a fundamental paradigm shift in software engineering, advocating for the integration of security as a core, non-negotiable component throughout the entire software development lifecycle. In contrast to traditional models where security is often treated as a reactive measure or a feature to be added post-development, SbD mandates a proactive stance. This philosophy requires that security considerations, from threat modeling to secure coding practices, are ingrained in the project's foundation, beginning with initial conception and planning and extending through design, implementation, deployment, and ongoing maintenance.

The development of the compliance auditing tools presented in this thesis was fundamentally guided by this philosophy. Auditing cookie consent mechanisms against the General Data Protection Regulation (GDPR) is not merely a matter of legal compliance, and it is an essential security function centered on protecting user data and privacy. Therefore, applying an SbD approach was critical to ensure that the resulting system would be inherently robust, reliable, and capable of producing trustworthy and actionable insights.

This principle-driven approach culminated in the creation of two distinct yet complementary software artifacts. The first is the cookie monster compliance suite, a broader framework designed to audit and validate cookie consent mechanisms. This suite is composed of three primary scripts: *CookieMonster.py*, *CookiebotValidator.py*, and *ComplianceReport.py*. The second artifact is the automated compliance validator, a more specialized tool engineered specifically to audit the GDPR compliance of websites that utilize the Cookiebot Consent Management Platform (CMP). Cookiebot is a commercial service that provides the interactive cookie consent banner and is responsible for managing and storing user privacy choices. While both tools share a common goal, the validator provides a focused and deeper analysis for a specific implementation.

3.1 Concepts

The foundational design concept of the validator is automation for accuracy. Manual compliance checks are time-consuming, prone to human error, and difficult to repeat consistently. This tool was conceived to replace that process with an automated, scriptable workflow that executes a comprehensive suite of tests with precision and generates verifiable evidence.

A second core concept is defense in depth for validation. The system does not rely on a single check and instead, it performs a multi-layered audit. It validates the user interface, the technical implementation of consent storage in browser cookies, the actual network behaviour resulting from that consent, and the accuracy of the cookie declarations against an external source of truth. This layered approach ensures that a failure in one area does not mask a potential compliance gap in another.

The final key concept is evidentiary clarity. The ultimate output of the tool is not just a pass or fail status but a detailed report. This report is designed to be a standalone piece of evidence, presenting its findings in a clear, human-readable format that is suitable for both technical teams and non-technical stakeholders, such as legal or compliance officers. The design prioritizes clean, understandable error messages over raw technical logs in the final output, making the results immediately actionable.

3.2 Elements

The validator is comprised of three primary, decoupled elements, each with a distinct responsibility. This modular separation is a deliberate design choice that enhances maintainability, testability, and clarity.

The first element is the orchestrator (*CookieMonster.py*). This script serves as the system's main entry point and configuration hub. It is responsible for defining the scope of the audit, such as the list of target websites. Its role is to initialize the core validation engine and the reporting module, execute the test suite, and then pass the results from the validator to the reporter. This simple, focused design keeps the high-level control logic separate from the complex implementation details.

The second and most critical element is the validation engine (*CookiebotValidator.py*). This class contains the entirety of the testing logic and can be considered the heart of the system. It is responsible for launching and controlling a headless web browser to simulate real user interactions with the target website's cookie banner. It systematically executes a series of validation scenarios, from accepting or rejecting all cookies to testing granular consent choices. Within this engine are sub-components for verifying the state of the *CookieConsent* cookie, analyzing network traffic for cross-border data transfers, and scraping the website's cookie declaration panel.

The third element is the reporting module (*ComplianceReport.py*). This component's sole function is to receive the structured data collected by the validation engine and transform it into a PDF document. It is designed to handle complex page layouts, including a unique cover page, a clean table of contents, and content pages with headers and footers. It

dynamically styles the report's content, using color-coding to draw attention to compliance mismatches or technical errors, thereby translating raw data into actionable insights.

Figure 1 - Overview of the elements

3.3 Technologies

The technology stack for this project was selected to ensure robustness, modern web compatibility, and performance. Python serves as the foundational programming language due to its powerful libraries and suitability for automation and data processing.

For browser automation and user simulation, Playwright was chosen. As a modern automation library, it provides superior reliability in handling dynamic, JavaScript-heavy websites compared to older tools. Its ability to wait intelligently for elements to become available and its precise locators are critical for interacting with the cookie banner's components without failure.

To validate cookie declarations against an external source, the system uses a combination of technologies. The Requests library is employed to perform HTTP queries against the Cookiepedia website. Cookiepedia functions as an open, public-facing database that maintains a comprehensive directory of web cookies and their purposes, serving as an impartial reference for this analysis. Since the target website is not a structured API, the BeautifulSoup library is used to parse the resulting HTML, intelligently scraping the necessary information about cookie descriptions and categories from the page's structure.

To enhance performance and act as a responsible internet citizen, the system incorporates a local caching mechanism using Python's built-in SQLite3 library. This creates a simple, file-based database to store the results of Cookiepedia lookups. This design drastically reduces the number of external network requests on subsequent runs and helps avoid rate-limiting, making the tool faster and more efficient.

Finally, the ReportLab library is used to generate the final PDF report. It was selected for its flexibility in programmatically creating complex, high-quality documents with precise control over layout, styling, and content.

3.4 Overview

The system operates in a clear, sequential workflow. The process begins when the Orchestrator (*CookieMonster.py*) is executed. It defines the target websites and instantiates the *CookiebotValidator* class. The validator's *run_all_tests* method is then invoked, which starts a headless browser instance.

For each target website, the validator navigates to the URL and begins its test sequence. It first checks the banner's initial state, then validates its content. It proceeds to test the accept all and refuse all button functionalities, strictly enforcing that the refuse option is available on the first layer. After each interaction, it verifies the technical state of the *CookieConsent* cookie to confirm the user's choice was correctly recorded. It then systematically tests granular consent options by navigating the customization menu.

Once user interaction tests are complete, the validator scrapes the cookie declaration from the banner to build an inventory. For each unique cookie found, it checks its local SQLite cache. If the cookie information is missing or incomplete, it queries Cookiepedia, scrapes the details, and updates the local cache for future use. The entire time, it logs all network requests to analyze them for potential cross-border data transfers.

Upon completion, the structured results, including any clearly explained failures, are returned to the Orchestrator. The Orchestrator then passes this data to the Reporting Module (*ComplianceReport.py*), which builds the PDF document section by section, applies styling, and saves the final *Compliance Report.pdf*.

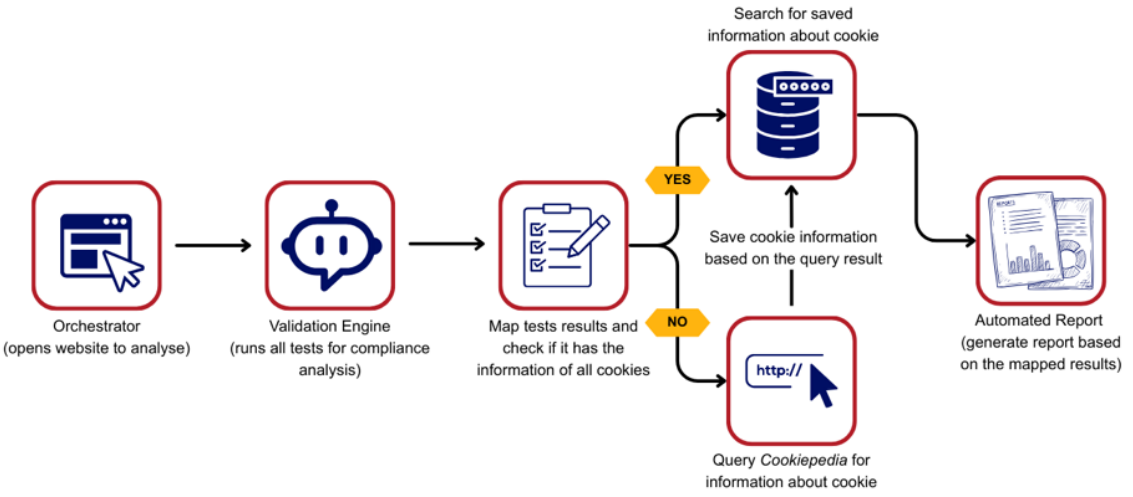


Figure 2 - HLD flowchart of the developed tool

3.5 Configuration and Scanner Modules

Within the context of this scripted tool, the concepts of a configuration page and a scanner page are not graphical user interfaces but rather logical modules. The *CookieMonster.py* script functions as the configuration module. It is here that an operator can easily modify the *websites_to_test* list to change the scope of an audit without needing to alter the core validation logic.

The *CookiebotValidator.py* class represents the scanner module. It is the self-contained engine that performs all the active scanning, interaction, and validation tasks. Its design as a class allows it to be instantiated with different configurations and run as a discrete, reusable component.

3.6 Requirements

The design of the automated compliance validator was driven by a strict set of functional and security requirements.

Functionally, the system is required to automatically validate the presence and content of a cookie banner, verify the correct operation of both accept all and refuse all buttons, and confirm that granular consent choices are accurately stored. A key compliance requirement is that the test for the refuse all button must fail if the button is not present on the first layer of the banner, enforcing the GDPR's equal ease principle. The system must also programmatically scrape all declared cookies, validate their stated purpose against Cookiepedia, and analyse network traffic for high-risk data transfers. The final functional requirement is the generation of a clean, shareable PDF report detailing all findings.

From a design and security perspective, the requirements mandate a modular architecture to ensure the system is maintainable and extensible. It must be robust enough to handle timeouts and variations in website structure. To ensure efficiency and responsible use of external resources, the system is required to implement a local caching strategy. Most importantly, it is required that all test failures are reported with clear, human-readable explanations in the final PDF, ensuring that the output is always actionable and its meaning unambiguous.

4 Development

The system operates based on a clear, linear execution flow orchestrated by the main script. The *CookieMonster.py* script acts as the conductor, initiating the process by instantiating the primary *CookiebotValidator* class. This class is the heart of the system, encapsulating all the logic required to perform the compliance tests on a given list of websites.

Upon invocation of its main execution method, *run_all_tests()*, the *CookiebotValidator* launches a headless browser instance and systematically navigates to each target URL. For each website, it executes a predefined suite of tests, which includes checking the initial state of the consent banner, validating its content, simulating various user consent choices (e.g., accepting all, refusing all, customising choices), and verifying that the website's cookie-setting behavior aligns with the user's expressed preferences. Throughout this process, it collects detailed results, logs network activity, and scrapes cookie information directly from the banner's declaration.

Once the test suite for all websites is complete, the *CookiebotValidator* returns a single dictionary containing all the collected data. This data structure is then passed by *CookieMonster.py* to the *generate_pdf_report* function within the *ComplianceReport.py* module. This final module parses the results dictionary and uses the ReportLab library to dynamically construct a formal PDF document, translating the raw test data into styled paragraphs, tables, and formatted text. This architectural separation ensures that the logic for testing is completely decoupled from the logic for presentation, making the system more modular and easier to maintain.

4.1 The Orchestrator

The *CookieMonster.py* script serves as the high-level entry point for the entire validation suite. Its design is intentionally minimalistic, as its primary function is to initialize the necessary components and orchestrate the main sequence of operations. It is the script that a user would execute to launch a full compliance audit.

The script begins by importing the necessary components from the other two modules: the *CookiebotValidator* class, which contains the testing logic, and the *generate_pdf_report* function, which handles the final output.

A global list, *websites_to_test*, is defined to hold the target URLs for the audit. This design allows for easy modification of the test targets without altering the core logic of the program. For this proof-of-concept, the list is intentionally focused on a single URL

to demonstrate in detail how the system behaves when analysing one of the organization's most significant digital assets. However, the application is fully capable of handling an extensive list of multiple URLs, allowing for a broad and scalable analysis in a production environment.

The core logic is encapsulated within a standard Python *if __name__ == "__main__":* block. This is a conventional practice ensuring that the code inside this block only runs when the script is executed directly, rather than when it is imported as a module into another script.

An instance of the *CookiebotValidator* class is created, passing the list of target websites to its constructor. This prepares the validator object with its primary targets.

The *run_all_tests()* method of the validator object is called. This is the most significant call in the script, as it triggers the entire suite of automated browser tests. This method can take a considerable amount of time to complete, depending on the number of websites and the complexity of the tests. The method is designed to return a dictionary containing the results of all tests performed.

After the tests are complete, the script checks if the *final_results* dictionary is not empty. This check ensures that a report is only generated if the test suite ran successfully and returned data.

If results are present, the *generate_pdf_report()* function is called, with the *final_results* dictionary passed as its argument. This initiates the creation of the final PDF document.

```python
from CookiebotValidator import CookiebotValidator
from ComplianceReport import generate_pdf_report

'''
The list websites_to_test is for the case study only.
The original version reads data from a CSV file containing
the domain name, priority level, and last analysis date,
and selects entries from highest to lowest priority.
'''
websites_to_test = ["https://www.continente.pt/"]

if __name__ == "__main__":
    print("Initializing Cookie Monster validator...")
    validator = CookiebotValidator(websites=websites_to_test)

    print("\n================================================")
    print("            STARTING COMPLIANCE TEST SUITE")
    print("================================================\n")

    # Run all the tests and collect the results into a single dictionary
    final_results = validator.run_all_tests()

    print("\n================================================")
    print("              TEST SUITE COMPLETE")
    print("================================================\n")

    # If tests ran successfully, generate the report
    if final_results:
        print("Generating final compliance report...")
        generate_pdf_report(final_results)
    else:
        print("Could not generate report because the test suite failed to return data.")
```

Figure 3 - Code snapshot of the CookieMonster.py

4.2 The Core Validation Engine

This module is the technical heart of the system, where the entirety of the compliance validation logic resides. The *CookiebotValidator* class is a sophisticated component that uses browser automation to simulate real user behaviour, inspect web elements, intercept network requests, and verify compliance against a set of predefined rules. It integrates several key libraries, including Playwright for robust browser control, BeautifulSoup for HTML parsing, and SQLite for data caching.

The class is initialized with a list of websites to test and an optional set of consent scenarios. If no scenarios are provided, a default list is used, ensuring comprehensive testing across different consent paths: accepting all, refusing all, and selecting specific categories (preferences, statistics, marketing), as well as a minimal consent scenario.

Upon initialization, the constructor sets up several instance variables to manage the state throughout the test run, including references to the Playwright instance, the browser, and the current page. A *report_data* dictionary is created to accumulate the results from all tests, and a *network_logs* list is used to store the URLs of all captured network requests.

A crucial step in the initialization process is the call to *_init_cookie_db()*, a private method that sets up a local SQLite database.

```
11  class CookiebotValidator:
12      """
13      Validates the functionality of a Cookiebot CMP on a list of websites,
14      supporting multiple consent scenarios parameterized in-code.
15      All errors in each check are reported, not fatal.
16      The results of all tests are returned for report generation.
17      """
18      def __init__(self, websites, consent_scenarios=None):
19          self.websites = websites
20          self.consent_scenarios = consent_scenarios or [
21              'accept_all',
22              'refuse_all',
23              'custom_statistics',
24              'custom_preferences',
25              'custom_marketing',
26              'custom_minimal'
27          ]
28          self.playwright = None
29          self.browser = None
30          self.page = None
31          self.network_logs = []
32          self.report_data = {}
33          self._init_cookie_db()
```

Figure 4 - Code snapshot of the CookiebotValidator class

The purpose of the database is to act as a cache for results from Cookiepedia, an external online database of cookies. Since querying this external service for every cookie on every run would be inefficient and could lead to rate-limiting issues, the system caches the results locally. The *_init_cookie_db* method ensures the database file and the required table exist, creating them if necessary. The *_get_cookie_from_db* and *_save_cookie_to_db* methods provide the interfaces for reading from and writing to this cache, significantly speeding up subsequent test runs and this caching mechanism is a critical design feature for performance and reliability.

```
76      # --- Database Helper Methods ---
77      def _init_cookie_db(self, db_path="cookie_cache.db"):
78          print(f"   - Initializing or connecting to local cookie cache: {db_path}")
79          self.db_conn = sqlite3.connect(db_path)
80          cursor = self.db_conn.cursor()
81          cursor.execute('''
82              CREATE TABLE IF NOT EXISTS cookiepedia_cache (
83                  name TEXT PRIMARY KEY, category TEXT NOT NULL, description TEXT, last_checked TIMESTAMP
84              )''')
85          self.db_conn.commit()
86
87      def _get_cookie_from_db(self, cookie_name):
88          cursor = self.db_conn.cursor()
89          cursor.execute("SELECT category, description FROM cookiepedia_cache WHERE name = ?", (cookie_name,))
90          return cursor.fetchone()
91
92      def _save_cookie_to_db(self, name, category, description):
93          cursor = self.db_conn.cursor()
94          cursor.execute('''
95              INSERT OR REPLACE INTO cookiepedia_cache (name, category, description, last_checked)
96              VALUES (?, ?, ?, ?)
97          ''', (name, category, description, datetime.now()))
98          self.db_conn.commit()
```

Figure 5 - Code snapshot of the database helper methods

The method *run_all_tests()* is the central control unit for the entire testing process. It methodically executes each step of the validation workflow for every website specified during initialization. It uses a *with sync_playwright()* as *self.playwright:* block, which is the standard way to manage the Playwright lifecycle, ensuring that all resources are properly handled and shut down.

The method iterates through each URL in *self.websites*. For each URL, it creates a new *browser_context*. This is an important step for test isolation; a new context is akin to a new, clean browser profile with no pre-existing cookies, cache, or local storage. This ensures that the tests for one website do not interfere with the tests for another.

A try, except and finally block wraps the entire test execution for a given URL. This makes the system resilient if a fatal, unrecoverable error occurs during the testing of one site, the program will log the error, clean up the resources for that site in the finally block (by clearing network logs and closing the context), and then proceed to the next site in the list, rather than crashing entirely.

The sequence of test execution is logical and systematic:

1. The browser navigates to the URL.
2. *_check_initial_state()* is called to verify the banner presence and no consent cookie is set.
3. *_validate_banner_dimensions_and_text()* checks the visibility and content of the banner itself.
4. The method then iterates through the defined *consent_scenarios*. For each scenario, it clears the cookies and reloads the page to ensure a fresh start. It then calls either *_validate_button_functionality()* (for "accept all" or "refuse all") or *_validate_customization_toggles()* (for granular choices) to simulate the user action and verify the outcome. The results are accumulated in local lists.
5. After all scenarios are tested, *_check_banner_reappearance()* confirms the banner shows up again after cookies are cleared.
6. *_analyze_cross_border_data()* inspects the collected network logs for potential GDPR compliance risks.
7. *_extract_and_validate_cookies()* performs the detailed scraping and validation of the cookie declaration.

```
35      # --- Master Test Runner ---
36      def run_all_tests(self):
37          with sync_playwright() as self.playwright:
38              self.browser = self.playwright.chromium.launch(headless=True)
39              for url in self.websites:
40                  print(f"\n--- Starting tests for: {url} ---")
41                  context = self.browser.new_context()
42                  self.page = context.new_page()
43                  self.page.on("request", self._log_request)
44                  try:
45                      self.page.goto(url, wait_until="domcontentloaded", timeout=60000)
46
47                      self.report_data['initial_state'] = self._check_initial_state()
48                      self.report_data['banner_content'] = self._validate_banner_dimensions_and_text()
49
50                      button_results, custom_results = [], []
51                      for scenario in self.consent_scenarios:
52                          print(f"\n--- Running Consent Scenario: {scenario} ---")
53                          self.page.context.clear_cookies()
54                          self.page.reload(wait_until="domcontentloaded")
55                          if scenario in ['accept_all', 'refuse_all']:
56                              button_results.extend(self._validate_button_functionality(accept=(scenario ==
                                'accept_all')))
57                          else:
58                              custom_results.extend(self._validate_customization_toggles(scenario))
59
60                      self.report_data['button_functionality'] = button_results
61                      self.report_data['customization_toggles'] = custom_results
62
63                      self.report_data['banner_reappearance'] = self._check_banner_reappearance()
64                      self.report_data['cross_border_data'] = self._analyze_cross_border_data()
65                      self.report_data['cookie_inventory'] = self._extract_and_validate_cookies()
66
67                      print(f"--- All tests for {url} completed successfully. ---")
```

```
68                  except Exception as e:
69                      print(f"[FATAL ERROR] An error stopped the test suite for {url}: {e}")
70                  finally:
71                      self.network_logs.clear()
72                      context.close()
73              self.browser.close()
74          return self.report_data
```

Figure 6 - Code snapshot of the master test runner

Finally, all collected results are stored in the *self.report_data* dictionary, which is returned at the end of the method.

Each private method prefixed with *_check_*, *_validate_*, or *_analyze_* corresponds to a specific compliance test.

For a user's initial encounter with the website, the system first verifies its pristine state. This involves confirming the visibility of the Cookiebot banner and the absence of any pre-existing CookieConsent cookie, a crucial step to ensure that consent is actively sought before any non-essential cookies are deployed, aligning with GDPR principles (*_check_initial_state()*).

Subsequently, the system rigorously tests the core functionalities of the “Accept All” and “Refuse All” buttons. It dynamically identifies these buttons, accommodating various language localizations, before simulating a click. The validation then extends to confirming the banner's disappearance and the accurate population of the CookieConsent cookie, reflecting either full acceptance or the rejection of all non-essential categories. The particular emphasis on the “Refuse All” button's accessibility on the initial layer underscores its importance in upholding the GDPR's “equal ease” principle, ensuring that declining cookies is as straightforward as accepting them (*_validate_button_functionality(accept=True)*).

Moving to more granular control, the system simulates user customization of cookie preferences. This involves navigating to the second layer of the banner by clicking the “Customize” or “Details” button. Depending on the specified scenario, the system programmatically manipulates individual category toggles (e.g., Preferences, Statistics, Marketing) to achieve a desired configuration. Finally, it clicks the “Allow selection” button and verifies that the resulting CookieConsent cookie precisely mirrors these specific, user-defined choices, confirming the website's adherence to granular consent (*_validate_customization_toggles(scenario)*).

In addition to managing cookies, the system executes a multi-stage analysis to detect and confirm potential cross-border data transfers.

The process begins with an initial screening of all network requests logged during the scan. The domain of each request is compared against an extensive list of non-EU indicators. This list is composed of country-code top-level domains (ccTLDs) such as *.us* (United States) and *.cn* (China), as well as the domains of numerous major international technology, analytics, and advertising companies known to operate outside of the European Union.

Any domain that matches an indicator is flagged as a potential cross-border transfer and proceeds to a more thorough verification phase. In this second stage, the system attempts to resolve the domain's IP address. If successful, it queries a geolocation service to identify the country where the server associated with that IP is physically located and the results are then classified into one of three categories:

- **Confirmed Non-EU Transfers**: Domains whose IP addresses are geolocated to a country outside the EU.
- **Confirmed EU Transfers**: Domains whose servers are verified to be within an EU member state.
- **Manual Review Required**: Domains that could not be verified, either because their IP address could not be resolved or because the geolocation lookup service failed to return a conclusive country location.

This layered approach provides a more precise and evidence-based assessment of where data is being sent, moving beyond simple URL-based flags to offer a clearer picture for data privacy compliance (*_analyze_cross_border_data()*).

Arguably the most sophisticated component of this validation engine, its dual purpose is to meticulously extract the complete roster of cookies declared within the website's Cookiebot banner and, crucially, to cross-reference the website's classification of each cookie against the authoritative information provided by the external Cookiepedia database.

The operational flow of this function is remarkably detailed, starting with its interaction with the banner. It automates the navigation to the cookie details section, subsequently clicking on each declared category, such as “Necessary” or “Statistics”, to ensure all associated cookie lists become visible.

Following this, the system proceeds with data scraping. For every category, it meticulously identifies all declared cookies and their respective providers. A key feature

here is its ability to account for instances where a single cookie name might represent multiple cookies (e.g., “_ga [x2]”), parsing this information to extract the cookie's name and its specified category. To ensure consistency, a helper function, *_normalize_category_key*, is then employed to standardize category names, reconciling linguistic or phrasing variations (e.g., “Analíticos” and “Statistics” are uniformly mapped to “statistics”), which is vital for accurate comparisons later in the process.

A crucial external validation loop then begins, iterating through a compiled list of all unique cookie names discovered. Before initiating a network request, the system first checks a local SQLite cache via *_get_cookie_from_db()*. If a complete and valid entry for a cookie is found in the cache, it utilizes this cached data, thereby conserving time and resources by bypassing redundant network calls.

Should a cookie not be found in the cache, or if the cached data is incomplete, the system constructs a URL for the cookie's page on Cookiepedia and dispatches an HTTP GET request. A deliberate 15-second delay (*time.sleep(15)*) is incorporated between these requests to prevent overwhelming the external server and to mitigate the risk of rate-limiting or blocking.

Upon receiving the HTTP response, BeautifulSoup is employed for HTML parsing. The system specifically searches for the cookie's "Type" (its category) and its description, with the parsing logic incorporating several fallbacks to accommodate potential variations in page layout. The newly retrieved category and description are then saved to the local database using *_save_cookie_to_db()*, ensuring that subsequent validations will benefit from this cached information, eliminating the need for repeated network lookups.

The system performs a critical comparison, where the category declared by the website is juxtaposed against the category retrieved from Cookiepedia. A status is then generated for each cookie: “OK” if the categories align, “MISMATCH” if they differ, and “ERROR” if the Cookiepedia lookup failed to yield a result.

This detailed information is structured into a dictionary containing the total number of declarations, the count of unique cookies, and a list of dictionaries, where each inner dictionary represents a cookie and its validation status. This structured output is perfectly suited for conversion into a table in the final report.

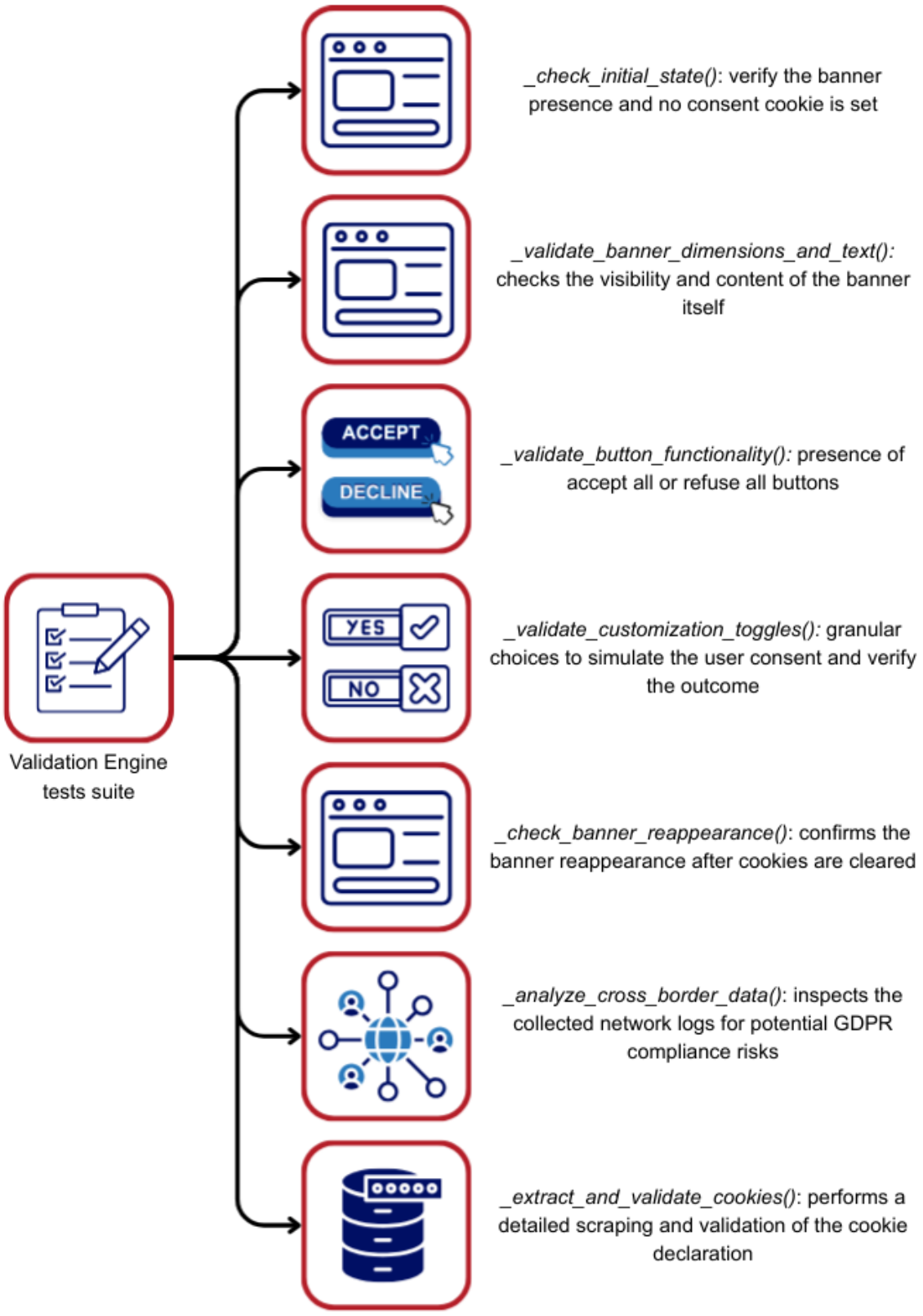


Figure 7 – HLD of the validation engine tests suite

4.3 Automated Report Generation

The final piece of the system is the *ComplianceReport.py* module, which is responsible for transforming the raw, structured data collected by the *CookiebotValidator* into a professional and readable PDF document. It leverages the open-source ReportLab library, which provides a programmatic way to construct PDF files from basic building blocks called flowables (e.g., paragraphs, tables, images).

To ensure the consistent styling and layout of the final document, the module leverages several key helper functions. For textual formatting, the *SectionStyle* function is employed to generate styled Paragraph objects specifically for section titles. This is achieved by defining a custom *ParagraphStyle* that specifies a Helvetica-Bold font, a precise size, and a distinct color (#000F63). This approach guarantees that all section headings throughout the report maintain a uniform and professional appearance, adhering to the organization's established report template.

Beyond text styling, the module utilizes a pair of callback functions, *draw_first_page* and *draw_later_pages*, which are essential for creating static page elements within the ReportLab framework. The *draw_first_page* function is dedicated to constructing a full-page cover. It accomplishes this by attempting to render a PNG image as a background, over which the primary title, Compliance Report, is centrally overlaid. To ensure robustness, this process is encapsulated within a try and except block, which gracefully manages any potential errors that might arise if the image file is not found.

For all subsequent pages, the *draw_later_pages* function is applied. This function is responsible for adding a consistent header and footer, thereby giving the report a polished and cohesive layout from the second page onward. The header contains a designated PNG logo, while the footer incorporates a horizontal line and a page number, completing the professional presentation of the *document.Report* in the center. A try and except block gracefully handle cases where the image file is missing.

```
7   def SectionStyle(section_number_text, section_name):
8       """Creates a styled Paragraph for a section title."""
9       section_style = ParagraphStyle(name="CustomTitleStyle", parent=getSampleStyleSheet()["Heading2"])
10      section_style.textColor = colors.HexColor("#000F63")
11      section_style.fontName = 'Helvetica-Bold'
12      section_text = f"{section_number_text}. {section_name}"
13      return Paragraph(section_text, section_style)
14
15  def draw_first_page(canvas, doc):
16      """Draws the full-page cover image on Page 1."""
17      canvas.saveState()
18      try:
19          canvas.drawImage('Cover.png', 0, 0, width=A4[0], height=A4[1])
20      except Exception as e:
21          print(f"Warning: Could not draw Cover.png. Error: {e}")
22          canvas.setFont('Helvetica-Bold', 24)
23          canvas.drawCentredString(A4[0]/2.0, A4[1]/2.0, "Compliance Report")
24      canvas.restoreState()
25
26  def draw_later_pages(canvas, doc):
27      """Draws elements on all pages after the first page."""
28      page_num = doc.page
29      canvas.saveState()
```

```
30      if page_num > 1:
31          try:
32              canvas.drawImage('MC_logo.png', -2, A4[1] - 110, width=2000/3.35, height=370/3.35)
33          except:
34              pass
35
36          page_text = f"Page {page_num + 1}"
37          canvas.setStrokeColorRGB(0, 0, 0)
38          canvas.setLineWidth(0.5)
39          canvas.line(66, 78, A4[0] - 66, 78)
40          canvas.setFont('Helvetica', 9)
41          canvas.drawString(A4[0]-128, 65, page_text)
42
43      canvas.restoreState()
```

Figure 8 - Code snapshot of the ComplianceReport

The central function of the reporting module is *generate_pdf_report()*, which orchestrates the complete construction of the PDF document from a *report_data* dictionary. The process commences with the initialization of a *SimpleDocTemplate* object, which serves as the foundational representation of the PDF file. Concurrently, an empty list, referred to as the story, is created. Within the ReportLab framework, the story acts as a sequential container for all content elements, such as paragraphs, tables, and spacers, that are to be methodically flowed into the document.

Initially, the function focuses on constructing a Table of Contents. A key design choice is the implementation of a *section_mapping* dictionary, which elegantly maps each section number to its corresponding title and the relevant key within the *report_data* dictionary. This structure enhances the code's clarity, readability, and extensibility, simplifying the future addition of new report sections and by iterating through this mapping, the function generates a simple list of section titles that form the Table of Contents page.

Following the creation of the Table of Contents, the function proceeds to build the main body of the report by iterating through the *section_mapping* dictionary a second time. For each section, it adds a styled title to the story using the *SectionStyle* helper function, ensuring visual consistency. A particularly powerful feature of this function is its capacity for conditional data handling, allowing it to adapt to various data structures present in the *report_data* dictionary. If the data for a given section is none, a message is appended to the story indicating that the corresponding test was not executed. If the data is presented as a list, the function iterates through it, adding each item as an individual paragraph, a method well-suited for reporting a series of pass/fail messages. When the data is a dictionary containing the specific key *report_data*, it is recognized as the unique data structure from the cookie inventory test, which in turn triggers the table generation logic.

For the cookie inventory section, the function dynamically constructs a table object. It assembles a list of lists, *table_data*, to populate the table's content, including its headers. A comprehensive *TableStyle* is defined to meticulously control every visual aspect of the table, from the header's background color and text color to alignment, font, padding, and grid lines. A crucial element of this process is a loop that iterates through the generated table data, examining the status column of each row. If a row's status is MISMATCH, a style command is applied to color the text red, if the status is ERROR, the text is colored orange and if the status is OK, the style applied is green. This conditional formatting provides an immediate and intuitive visual cue that highlights the most critical findings in the report.

Finally, the function culminates in the call to *doc.build(story, ...)* which passes the complete story list along with references to the *draw_first_page* and *draw_later_pages* callback functions. At this point, the ReportLab engine takes control, expertly flowing all the content into the document, applying the predefined styles, drawing the consistent headers and footers, and ultimately saving the final PDF file to disk.

This module successfully abstracts the complexities of PDF generation, providing a robust mechanism to translate technical test results into a polished, professional, and highly informative final deliverable.

4.4 Demonstration

To illustrate the practical application and effectiveness of the developed software, this section provides a demonstration of the *Cookie Monster Validator* performing a full compliance audit on a live website. The objective of this demonstration is to showcase the end-to-end workflow, from the initial command execution to the generation of the final, detailed PDF report, thereby validating the successful integration of all system components. The target website for this demonstration is continente[.]pt, as specified in the *CookieMonster.py* orchestrator script.

Before executing the validation script, it is essential to properly set up the Python environment to ensure that all required dependencies are installed. This setup streamlines the process of running the compliance tests and guarantees consistent results across different systems. The following shell script automates the entire setup workflow: creating and activating a dedicated Python virtual environment, installing all necessary packages as specified in the requirements, and preparing the Playwright browsers for automated testing. This approach isolates the project dependencies, preventing conflicts with other Python projects and simplifying maintenance and

deployment. Users will need to run this script once before running the main compliance validator and subsequently activate the virtual environment as needed to perform audits.

Once the environment is ready and the virtual environment is activated, the audit is initiated by a user from the command line by simply running the main orchestrator script.

Upon execution, the application immediately provides feedback in the console, indicating that the process has started. The *CookieMonster.py* script first initializes the *CookiebotValidator* class, which in turn establishes a connection to the local *cookie_cache.db* SQLite database. The script then announces the start of the full test suite and launches a headless Chromium browser instance managed by Playwright.

```
#!/bin/bash

set -e

echo "Creating Python virtual environment in ./.venv ..."
python3 -m venv ./.venv

echo "Activating virtual environment..."
source .venv/bin/activate

echo "Installing Playwright Python package in venv..."
pip install --upgrade pip
pip install -r requirements.txt

echo "Installing Playwright browsers in venv..."
playwright install

echo "Playwright and browser installation complete! Environment is ready."
echo "To activate this environment again later, run: source .venv/bin/activate"
```

Figure 9 - Code snapshot of the setup script

The console output provides a real-time narrative of the validator's progress, making the automated process transparent to the user. A condensed representation of this output would appear as follows:

```
Initializing Cookie Monster validator...
 - Initializing or connecting to local cookie cache: cookie_cache.db

================================================
 STARTING COMPLIANCE TEST SUITE
================================================

--- Starting tests for: https://www.continente.pt/ ---

1. Checking Initial State...
2. Validating Banner Display and Content...

--- Running Consent Scenario: accept_all ---
3. Validating Button Functionality: Accept All

--- Running Consent Scenario: refuse_all ---
3. Validating Button Functionality: Refuse All

--- Running Consent Scenario: custom_statistics ---
4. Validating Customization Toggles: Custom Statistics
... (and so on for all other scenarios) ...

5. Checking Banner Reappearance...
6. Analyzing Network Requests for Cross-Border Data...
7. Cookie Inventory & Validation (with DB Cache)...
 - Processing category button: Necessário
 - Mapped cookies from category 'Necessário'.
 - Processing category button: Preferências
 - Mapped cookies from category 'Preferências'.
 ...
 [DEBUG] Starting validation for 124 unique cookies...
 [DEBUG] [1/124] Processing cookie: 'AWSALB'
  -> [DEBUG] Found complete data in local cache. Skipping network call.
 [DEBUG] [2/124] Processing cookie: 'weird_get_top_level_domain'
  -> [DEBUG] Not in cache or data is incomplete. Preparing to query Cookiepedia...
  -> [DEBUG] Pausing for 15 seconds to avoid rate-limiting...
--- All tests for https://www.continente.pt/ completed successfully. ---

================================================
 TEST SUITE COMPLETE
================================================

Generating final compliance report...
Building PDF report...
Report 'Compliance Report.pdf' created successfully.
```

Figure 10 - Example of output from the Cookie Monster validator command line

This console log demonstrates the systematic execution of each validation step, and it shows the validator iterating through the different consent scenarios, from accept all to granular selections. A particularly insightful part of the log is the output from the cookie inventory check, which shows the script scraping cookies from each category and then validating each unique cookie, leveraging the cache for efficiency and making external network calls only when necessary.

The primary artifact of this entire process is the generated file, *Compliance Report.pdf*. This multi-page document is professionally formatted with a cover page, a table of contents, and consistently styled headers and footers on each page. The most impactful section of the report is the final table, which presents the results of the cookie inventory validation. This table clearly lists each cookie, its category as declared by the website, its category according to the independent Cookiepedia database, and a final status. The conditional formatting, which colors mismatches in red and errors in orange, provides an immediate visual summary of potential compliance issues.

The following presents the results from the scan of the specified URL, followed by a brief analysis of the findings.

Cookie Name	Declared Category	DB/Cookiepedia Category	Status
AWSALB	Necessary	Necessary	OK
AWSALBCORS	Necessary	Necessary	OK
AWSALBTG	Necessary	Necessary	OK
AWSALBTGCORS	Necessary	Necessary	OK
test_cookie	Necessary	Marketing	MISMATCH
rc::a	Necessary	Marketing	MISMATCH
rc::c	Necessary	Marketing	MISMATCH
JSESSIONID	Necessary	Necessary	OK
c	Necessary	Necessary	OK
ts	Necessary	Functional	MISMATCH
weird_get_top_level_domain	Necessary	No Info	MISMATCH
byside_cookie_service_level	Necessary	No Info	MISMATCH
rbzid	Necessary	Unknown	MISMATCH
realUserVerifier	Necessary	No Info	MISMATCH
ar_debug	Necessary	Unknown	MISMATCH
CookieConsent	Necessary	Necessary	OK
ARRAffinity	Necessary	Necessary	OK
ARRAffinitySameSite	Necessary	Necessary	OK
ASLBSA	Necessary	Necessary	OK

ASLBSACORS	Necessary	Necessary	OK
BrowserId	Necessary	Functional	MISMATCH
CookieConsentPolicy	Necessary	Necessary	OK
LSKey-c$CookieConsentPolicy	Necessary	Necessary	OK
__anact	Necessary	Functional	MISMATCH
__cq_dnt	Necessary	Functional	MISMATCH
__cqact	Necessary	Functional	MISMATCH
__dc_env	Necessary	No Info	MISMATCH
__dcact	Necessary	No Info	MISMATCH
cqcid	Necessary	Unknown	MISMATCH
cquid	Necessary	Unknown	MISMATCH
dw_dnt	Necessary	Necessary	OK
dwac_#	Necessary	Marketing	MISMATCH
dwanonymous_UID{32}	Necessary	Marketing	MISMATCH
dwsid	Necessary	Necessary	OK
MailingAnonID	Necessary	No Info	MISMATCH
sid	Necessary	Necessary	OK
ssoSession	Necessary	Unknown	MISMATCH
tabHistory_#	Necessary	No Info	MISMATCH
g	Functional	Marketing	MISMATCH
_ga	Statistics	Statistics	OK
_ga_#	Statistics	Unknown	MISMATCH
_gcl_au	Statistics	Marketing	MISMATCH
CLID	Statistics	Functional	MISMATCH
NRBA_SESSION	Statistics	No Info	MISMATCH
cq.recoUUID	Statistics	No Info	MISMATCH
cq.viewCategory	Statistics	No Info	MISMATCH
cq.viewReco	Statistics	No Info	MISMATCH
__cq_uuid	Statistics	Unknown	MISMATCH
_tt_enable_cookie	Statistics	Marketing	MISMATCH
eu01/v.gif	Statistics	No Info	MISMATCH
vwo_apm_sent	Statistics	No Info	MISMATCH
_vwo_uuid_v2	Statistics	No Info	MISMATCH
undefined	Statistics	Marketing	MISMATCH
log/error	Marketing	Unknown	MISMATCH
_fbp	Marketing	Marketing	OK
C	Marketing	Necessary	MISMATCH
uid	Marketing	Unknown	MISMATCH
uuid2	Marketing	Marketing	OK
XANDR_PANID	Marketing	No Info	MISMATCH

_gat	Marketing	Statistics	MISMATCH
_gid	Marketing	Statistics	MISMATCH
IDE	Marketing	Marketing	OK
NID	Marketing	Marketing	OK
pagead/1p-conversion/#/	Marketing	No Info	MISMATCH
pagead/1p-user-list/#	Marketing	No Info	MISMATCH
pagead/gen_204	Marketing	No Info	MISMATCH
pagead/gen_204/	Marketing	No Info	MISMATCH
csi	Marketing	Unknown	MISMATCH
__rtbh.lid	Marketing	Marketing	OK
__rtbh.uid	Marketing	Marketing	OK
cq.anchor	Marketing	No Info	MISMATCH
__cq_bc	Marketing	Unknown	MISMATCH
__cq_seg	Marketing	Functional	MISMATCH
uuid	Marketing	Marketing	OK
tt_appInfo	Marketing	No Info	MISMATCH
tt_pixel_session_index	Marketing	No Info	MISMATCH
tt_sessionId	Marketing	No Info	MISMATCH
_ttp	Marketing	Marketing	OK
#-#	Marketing	No Info	MISMATCH
iU5q-!O9@[#COOKIETABLE_ADVERTISING#]nbsp;	Marketing	No Info	MISMATCH
LAST_RESULT_ENTRY_KEY	Marketing	Unknown	MISMATCH
nextId	Marketing	Unknown	MISMATCH
requests	Marketing	Unknown	MISMATCH
yt.innertube::nextId	Marketing	Unknown	MISMATCH
yt.innertube::requests	Marketing	Unknown	MISMATCH
ytidb::LAST_RESULT_ENTRY_KEY	Marketing	No Info	MISMATCH
YtIdbMeta#databases	Marketing	No Info	MISMATCH
yt-remote-cast-available	Marketing	Unknown	MISMATCH
yt-remote-cast-installed	Marketing	Unknown	MISMATCH
yt-remote-connected-devices	Marketing	Unknown	MISMATCH
yt-remote-device-id	Marketing	Unknown	MISMATCH
yt-remote-fast-check-period	Marketing	Unknown	MISMATCH
yt-remote-session-app	Marketing	Unknown	MISMATCH

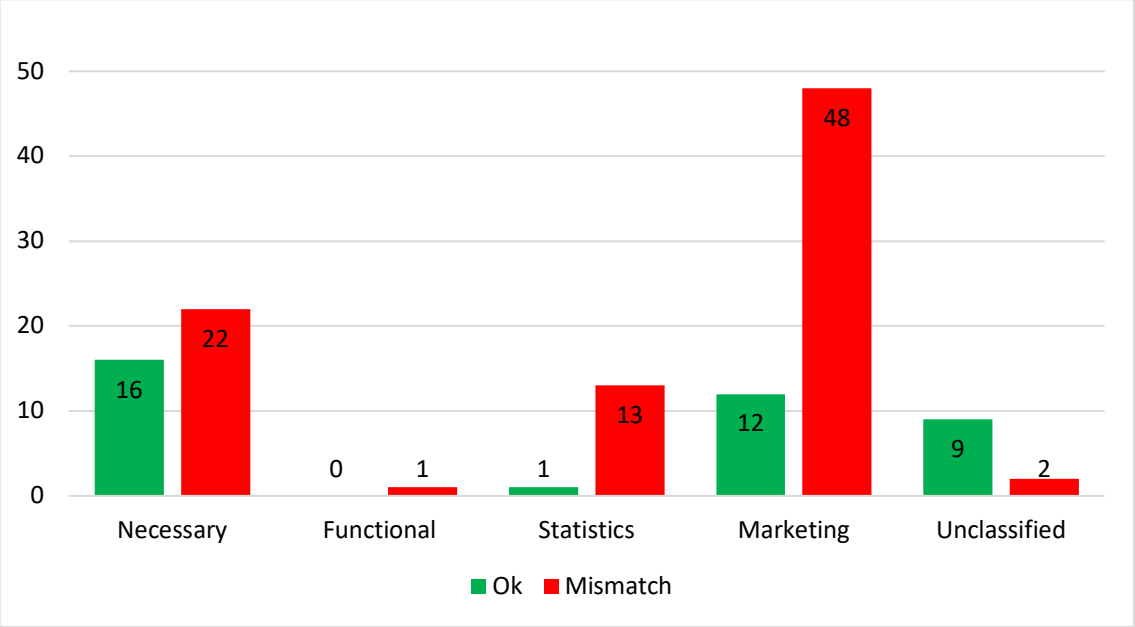
yt-remote-session-name	Marketing	Unknown	MISMATCH
__Secure-ROLLOUT_TOKEN	Marketing	No Info	MISMATCH
__Secure-YEC	Marketing	Unknown	MISMATCH
LogsDatabaseV2:V#||LogsRequestsStore	Marketing	No Info	MISMATCH
remote_sid	Marketing	Unknown	MISMATCH
ServiceWorkerLogsDatabase#SWHealthLog	Marketing	No Info	MISMATCH
TESTCOOKIESENABLED	Marketing	Necessary	MISMATCH
VISITOR_INFO1_LIVE	Marketing	Marketing	OK
YSC	Marketing	Unknown	MISMATCH
byside_webcare_session_tid	Marketing	Functional	MISMATCH
byside_webcare_tuid	Marketing	Marketing	OK
lastExternalReferrer	Marketing	Unknown	MISMATCH
lastExternalReferrerTime	Marketing	No Info	MISMATCH
topicsLastReferenceTime	Marketing	No Info	MISMATCH
sp_ssid	Marketing	Marketing	OK
spid	Marketing	Marketing	OK
syndi_pageid_timestamps	Marketing	No Info	MISMATCH
event/a.gif	Marketing	Unknown	MISMATCH
event/p.gif	Marketing	Unknown	MISMATCH
_gcl_ls	Marketing	No Info	MISMATCH
v1/tiles	Unclassified	Unknown	OK
ttcsid	Unclassified	No Info	OK
ttcsid_CUVG0JJC77U9HMCID3V0	Unclassified	No Info	OK
automaise-botchat-es5	Unclassified	No Info	OK
automaise-botchat-es5-version	Unclassified	No Info	OK
_taggstar_cfg_contin entept	Unclassified	No Info	OK
_taggstar_exps	Unclassified	No Info	OK
_taggstar_ses	Unclassified	Marketing	MISMATCH
_taggstar_vid	Unclassified	Necessary	MISMATCH
waap_id	Unclassified	No Info	OK
event/h.gif	Unclassified	Unknown	OK

Table 1 - Cookie categorization: website declaration vs database or Cookiepedia

To complement the detailed tabular data, a graphical summary provides an invaluable tool for rapid analysis. A grouped bar chart is generated to present a clear and insightful overview of the findings. This visualization effectively segregates the cookie counts within each category, distinguishing between those that were accurately declared (OK) and those whose categorization conflicts with the external Cookiepedia database (MISMATCH).

From a total of 183 cookie declarations found, 124 unique cookies were identified for validation. This detailed visualization immediately reveals critical insights that a simpler chart might obscure. It becomes instantly apparent that the most significant compliance issues lie within the Marketing category, which contains 48 mismatched cookies, vastly outnumbering the 12 that were correctly classified. The Necessary and Statistics categories also show a substantial number of mismatches (22 and 13, respectively), indicating a systemic issue with cookie classification on the site. Conversely, the Functional and Unclassified categories show minimal discrepancies.

The analysis also surfaced a critical failure in the user consent mechanism. During the Refuse All test scenario, the scan determined that the Refuse All button was not present or immediately visible on the first layer of the cookie banner. This finding is of particular concern as it suggests a potential violation of the GDPR's “equal ease” principle, which mandates that rejecting cookies must be as simple for the user as accepting them.



Graphic 1 - Cookie validation by category

In conclusion, this demonstration confirms that the developed application functions as a cohesive and effective system. It successfully integrates the orchestration logic of *CookieMonster.py*, the complex browser automation and validation engine of *CookiebotValidator.py*, and the sophisticated document generation capabilities of *ComplianceReport.py*. The tool autonomously performs a series of complex compliance checks that would otherwise require hours of meticulous manual effort, and it presents its findings in a clear, actionable, and professional format, thereby fulfilling the core objectives of this thesis.

5 Conclusion

This final chapter serves to consolidate the findings and outcomes of the research project, reflecting on the journey from the initial proposal to the final implemented solution. It will first summarize the work accomplished, evaluating the extent to which the developed application, the *Cookie Monster Validator*, successfully met the objectives outlined in the thesis proposal. Following this, the chapter will explore potential avenues for future development, outlining enhancements that could build upon the current foundation to create an even more powerful and comprehensive compliance tool, with a particular focus on the integration of Artificial Intelligence.

5.1 Present Work

The central ambition of this thesis, undertaken in a strategic partnership with MC Sonae, was to confront the persistent and critical gap between the stringent legal requirements of modern cookie consent regulations, such as the GDPR, and their often-flawed practical implementation on websites. The proposal articulated the vision for a cohesive application designed to automate the multifaceted process of cookie management compliance, with a sharp focus on validating banner interactions, ensuring the integrity of granular consent mechanisms, analyzing cross-border data flows, and delivering transparent, actionable reports. The developed software solution successfully materializes this vision, delivering a robust and systematic auditing tool specifically engineered to validate Cookiebot-powered consent management platforms.

The implemented system, architecturally designed as three interconnected Python modules, directly and comprehensively addresses the core functionalities outlined in the proposal. In response to the proposal's requirement to validate banner visibility and button functionality, the *CookiebotValidator.py* module delivers precise verification. Its functions methodically assert the initial presence and correct dimensions of the consent banner, and, crucially, validate the behavior of primary user choices. The script does not merely click the Accept All and Refuse All buttons, it proceeds to interrogate the browser's state to confirm that the CookieConsent cookie is created with the correct boolean values for each category. This provides empirical proof of compliance. Significantly, the test for the immediate visibility of a Refuse All button on the first layer of the banner directly assesses adherence to the GDPR's "equal ease" principle, which mandates that rejecting consent should be as straightforward as granting it.

Furthermore, a central pillar of the proposal was the validation of granular consent, acknowledging that true user choice lies in the ability to select specific data processing

purposes. The *_validate_customization_toggles* function was engineered specifically to meet this challenge. It automates the complex user journey of navigating to the configuration panel, programmatically interacting with the individual toggles for categories like Preferences, Statistics, and Marketing, and then verifying that the resulting CookieConsent cookie precisely mirrors the user's fine-grained selections. This function provides automated assurance that a website is truly respecting the specific and informed preferences of its users.

The proposal also underscored the profound privacy risks associated with unlawful data sharing and the need to monitor consent states rigorously. The *_analyze_cross_border_data* function serves as a direct and powerful response, programmatically scanning all captured network traffic for requests directed to domains associated with non-EU jurisdictions or major international technology corporations. While this function provides a preliminary risk indicator rather than a definitive legal judgment, it acts as an invaluable automated first-pass screening tool for data protection officers. Complementing this, the entire test suite is built upon a foundation of checking and re-checking the CookieConsent cookie's state, ensuring that it does not exist on initial load and is updated correctly after every simulated user interaction, thereby confirming that consent is being tracked accurately.

Finally, the thesis aimed to simplify the complexity of compliance for website owners and enhance transparency for all stakeholders. The *ComplianceReport.py* module is the embodiment of this objective. Moving far beyond the raw output of typical scripts, it transforms the collected data into a professional, multi-page PDF document. The report is structured with a table of contents, clearly demarcated sections, and a detailed, dynamically color-coded table of findings. A significant contribution that extends beyond the initial proposal is the functionality of *_extract_and_validate_cookies*. This function does not merely list the cookies a website declares; it performs a data audit by programmatically cross-referencing the website's categorization of each cookie against the independent, external Cookiepedia database. This validation step directly confronts the issue of irresponsible or inaccurate cookie declarations, providing a deeper layer of scrutiny and reinforcing the tool's value as a genuine compliance asset for entities like MC Sonae, who can leverage it to support the activities of their Data Protection Officer and strengthen customer trust.

5.2 Future Work

While the current implementation represents a robust and comprehensive foundation, the landscape of data privacy and web technology is in a state of perpetual evolution. To

maintain its relevance and expand its impact, the *Cookie Monster Validator* can be enhanced through several key avenues of future work. These potential developments would significantly augment the tool's capabilities, improve its accuracy, and broaden its scope, transforming it from a platform-specific validator into a more universal and intelligent auditing system.

A primary path for expansion would be to evolve the tool's architecture to support a wider array of Consent Management Platforms (CMPs). The current validator is expertly tailored to the specific DOM selectors and behavioral patterns of the Cookiebot CMP. A significant and valuable future project would involve refactoring the core logic to achieve platform independence. This could be realized by designing an abstract base class for a CMP adapter, which defines a standard interface for actions like finding the accept button, interacting with customization toggles, and retrieving cookie declarations. Subsequently, concrete implementations could be developed for other market-leading CMPs. Such a modular architecture would render the tool universally applicable, exponentially increasing its utility for a global audience of developers, auditors, and regulators.

Building upon this foundation of broader compatibility, the next logical step is to enhance the tool's analytical intelligence for handling unclassified or novel cookies. The current system's reliance on the Cookiepedia database is effective but inherently limited by the database's existing knowledge. A more advanced version of the validator could incorporate a dynamic analysis module. This system would work by sandboxing a browser session after a specific consent level is granted, actively monitoring for the placement of any new cookies not declared in the CMP. For each such cookie, the tool could then initiate an automated investigation, performing WHOIS lookups on its domain, analyzing associated script behaviors, and using other heuristics to make an educated inference about its provider and purpose. This would empower the tool to flag suspicious or unclassified trackers for manual review, providing a crucial defense against emerging privacy threats.

A further enhancement, directly inspired by the initial proposal, would be a deeper and more realistic integration with the browser's native settings. While the current tool clears cookies programmatically via API calls for test isolation, a future iteration could use automation frameworks to interact directly with the browser's user interface. This would enable the simulation of more authentic user behaviors, such as navigating to the Chrome or Firefox settings page, manually deleting site data, or enabling the block all third-party cookies option. Observing how a website and its CMP respond to these native

browser actions would provide a more holistic test of compliance, assessing its resilience and behavior from the perspective of a privacy-conscious but not necessarily technical user.

The most transformative and powerful direction for future work, however, lies in the integration of Artificial Intelligence and Machine Learning. This would elevate the validator from a deterministic, rule-based script to an intelligent system capable of understanding nuance and context. An AI-powered Natural Language Processing (NLP) model could be trained to automatically fetch, parse, and analyze a website's privacy and cookie policy documents. This model would go beyond simple keyword searching, using semantic understanding to verify that the cookies technically identified in the CMP are also accurately and clearly described in the legal text. It could be fine-tuned to detect ambiguous, misleading, or non-compliant language, flagging policy sections that fail to adequately inform users about data processing activities, thus bridging the critical gap between technical implementation and legal disclosure.

In parallel, a sophisticated Machine Learning model could be employed for behavioral anomaly detection in network traffic. By training a model on the network logs from thousands of sessions on websites known to be fully compliant, the system could learn the signature of a normal, privacy-respecting interaction for any given consent level. When auditing a new website, the validator could then use this model to identify deviations from this baseline. This behavior-based approach could flag suspicious activities, such as data being sent to obscure domains, unusually large data payloads, or requests firing at unexpected times, that would be missed by the current static, list-based analysis. This would represent a paradigm shift from a blacklist approach to a dynamic, self-learning system capable of identifying novel tracking techniques.

Finally, the power of computer vision could be harnessed to analyze the user experience of the consent banner itself. A trained model could process a screenshot of the banner to assess compliance with more subjective principles. It could programmatically verify if the color contrast between text and background meets accessibility standards (WCAG), ensuring readability for all users. More importantly, it could locate the Accept and Reject buttons, measure their size, and analyze their color prominence and placement. This would provide an objective, data-driven assessment of whether a design constitutes a dark pattern by making it visually or cognitively more difficult to refuse consent than to grant it. By pursuing these AI-driven enhancements, the validator could evolve into a truly intelligent compliance auditing system, offering unparalleled depth and insight in the ongoing mission to foster a more transparent and trustworthy digital ecosystem.

References

1. MC Sonae. "About Us". URL: https://mc.sonae.pt/en/about-us/
2. Bollinger, Dino. "Automated Detection and Analysis of Cookie Banners". Doctoral Thesis, ETH Zurich, 2021. URL: https://www.research-collection.ethz.ch/bitstream/handle/20.500.11850/477333/Bollinger_Dino.pdf
3. Carlos, C. "Cookie Proposal". Online Document, ETH Zurich. URL: https://people.inf.ethz.ch/ccarlos/assets/proposals/cookie_proposal.pdf
4. Matte, C., Le Chen, B., and K. Raschke. "Dark Patterns in TCF cookie banners: a case study of the top 100 German news websites". In ACM International Conference Proceeding Series, July 2021. URL: https://dl.acm.org/doi/abs/10.1145/3466722
5. Luguri, J., and L. Strahilevitz. "Dark Patterns and the Fictions of Consent". SSRN Electronic Journal, January 2021. URL: https://papers.ssrn.com/sol3/papers.cfm?abstract_id=3761967
6. Bollinger, Dino, et al. "Automated Detection of Cookie Banner Issues". In USENIX Security Symposium, August 2022. URL: https://www.usenix.org/conference/usenixsecurity22/presentation/bollinger
7. Stoica, M., and C. F. Stoica. "Land Grabbing in Romania and Interlinkages with EU's Common Agricultural Policy". Communicating the EU, 2017. URL: https://d1wqtxts1xzle7.cloudfront.net/63441147/Communicating_EU_-_M_Stoica_et_al20200527-15405-k0cw62-libre.pdf
8. Fetic, Arnesa. "Web Privacy and the Impact of the GDPR". Master's Thesis, ETH Zurich, 2024. URL: https://www.research-collection.ethz.ch/handle/20.500.11850/662039
9. Zuboff, Shoshana. "The Age of Surveillance Capitalism: The Fight for a Human Future at the New Frontier of Power". Harvard Business School, January 2019. URL: https://www.hbs.edu/faculty/Pages/item.aspx?num=56791
10. ScienceDirect. "Surveillance Capitalism - an overview". ScienceDirect Topics. URL: https://www.sciencedirect.com/topics/social-sciences/surveillance-capitalism
11. Walsh, Colleen. "'Surveillance capitalism' is undermining democracy". The Harvard Gazette, March 2019. URL: https://news.harvard.edu/gazette/story/2019/03/harvard-professor-says-surveillance-capitalism-is-undermining-democracy/

12. The European Parliament and the Council of the European Union. "Directive 2002/58/EC (ePrivacy Directive)". Official Journal of the European Union, July 2002. URL: https://eur-lex.europa.eu/eli/dir/2002/58/oj/eng
13. The European Parliament and the Council of the European Union. "Directive 2009/136/EC". Official Journal of the European Union, December 2009. URL: https://eur-lex.europa.eu/eli/dir/2009/136/oj/eng
14. The European Parliament and the Council of the European Union. "Directive 1995/46/EC (Data Protection Directive)". Official Journal of the European Union, November 1995. URL: https://eur-lex.europa.eu/eli/dir/1995/46/oj/eng
15. The European Parliament and the Council of the European Union. "Regulation (EU) 2016/679 (General Data Protection Regulation)". Official Journal of the European Union, April 2016. URL: https://eur-lex.europa.eu/eli/reg/2016/679/oj/eng
16. European Data Protection Board (EDPB). "The data protection authority and you". SME Data Protection Guide. URL: https://www.edpb.europa.eu/sme-data-protection-guide/data-protection-authority-and-you_en
17. State of California Department of Justice. "California Consumer Privacy Act (CCPA)". Office of the Attorney General. URL: https://oag.ca.gov/privacy/ccpa
18. GDPR Summary. "Schrems II". GDPRSummary.com. URL: https://www.gdprsummary.com/schrems-ii/
19. IAB Europe. "Transparency & Consent Framework (TCF)". IAB Europe. URL: https://iabeurope.eu/transparency-consent-framework/
20. Adam Barth. HTTP State Management Mechanism. RFC 6265, April 2011. URL: https://rfc-editor.org/rfc/rfc6265.txt
21. Sandi Hardmeier. The history of internet explorer. URL: https://web.archive.org/web/20051001113951/http://www.microsoft.com/windows/IE/community/columns/historyofie.mspx, 2005
22. David M. Kristol. Http cookies: Standards, privacy, and politics. ACM Trans. Internet Technol., 1(2):151–198, November 2001. doi: 10.1145/502152.502153
23. Lou Montulli. Persistent client state in a hypertext transfer protocol-based client-server system, Jun 1998
24. Lou Montulli and David M. Kristol. HTTP State Management Mechanism. RFC 2965, October 2000. URL: https://rfc-editor.org/rfc/rfc2965.txt
25. Andrew Stuart. Where cookie comes from. URL: http://dominopower.com/article/where-cookie-comes-from/, 2002

26. Lou Montulli and David M. Kristol. HTTP State Management Mechanism. RFC 2109, February 1997. URL https://rfc-editor.org/rfc/rfc2109.txt
27. Usercentrics A/S. "Cookiebot CMP by Usercentrics". Online, 2025. URL: https://www.cookiebot.com
28. OneTrust, LLC. "Cookiepedia". Online, 2025. URL: https://cookiepedia.co.uk/
29. Mozilla Developer Network (MDN). “Using HTTP cookies - HTTP”. Documentation page, MDN Web Docs. URL: https://developer.mozilla.org/en-US/docs/Web/HTTP/Guides/Cookies
30. Harvard University (Enterprise Architecture). “Use of HTTP Cookies”. URL: https://enterprisearchitecture.harvard.edu/use-http-cookies
31. Bollinger, Dino; Kubicek, Karel; Cotrini, Carlos; Basin, David. “Automating Cookie Consent and GDPR Violation Detection”. Conference paper, 31st USENIX Security Symposium (USENIX Security ’22), 2022. URL: https://www.usenix.org/system/files/sec22-bollinger.pdf
32. “Economic consequences of online tracking restrictions”. Journal article (ScienceDirect). URL: https://www.sciencedirect.com/science/article/pii/S0167811623000708
33. “Cookiescanner: An Automated Tool for Detecting and Evaluating GDPR Consent Notices on Websites”. URL: https://arxiv.org/pdf/2309.06196
34. Bouhoula, Ahmed (ETH Zurich) et al. “Automated Large-Scale Analysis of Cookie Notice Compliance”. ETH Research Collection. URL: https://www.research-collection.ethz.ch/server/api/core/bitstreams/fa4a066f-4fba-484a-a4bc-d3c4f5afa4b9/content
35. Mozilla Developer Network (MDN). “Third-party cookies - Privacy on the web”. Documentation page, MDN Web Docs. URL: https://developer.mozilla.org/en-US/docs/Web/Privacy/Guides/Third-party_cookies
36. “How Can and Would People Protect From Online Tracking?”. PoPETs, 2022. URL: https://petsymposium.org/popets/2022/popets-2022-0006.pdf
37. “”I’m Not for Sale” – Perceptions and Limited Awareness of Privacy Risks by Digital Natives About Location Data”. URL: https://arxiv.org/html/2502.11658v3
38. “Transparency, Awareness, and Privacy Cynicism”. Journal article (Taylor & Francis Online). URL: https://www.tandfonline.com/doi/full/10.1080/15252019.2025.2546338#abstract
39. European Data Protection Board. “Guidelines 3/2022 on Dark patterns in social media platform interfaces: How to recognise and avoid them (Version 1.0, adopted on 14 March 2022)”. EDPB, 2022. URL:

https://www.edpb.europa.eu/system/files/2022-03/edpb_03-2022_guidelines_on_dark_patterns_in_social_media_platform_interfaces_en.pdf

40. Google. “Cookie Matching | Real-time Bidding | Google for Developers”. Developer documentation page. URL: https://developers.google.com/authorized-buyers/rtb/cookie-guide
41. Nikiforakis, N. et al. “Cookieless Monster: Exploring the Ecosystem of Web-Based Device Fingerprinting”. IEEE Symposium on Security and Privacy, 2013. URL: https://www.researchgate.net/publication/261310314_Cookieless_Monster_Exploring_the_Ecosystem_of_Web-Based_Device_Fingerprinting

Attachments

Figure 11 - Illustrative cover page of the report

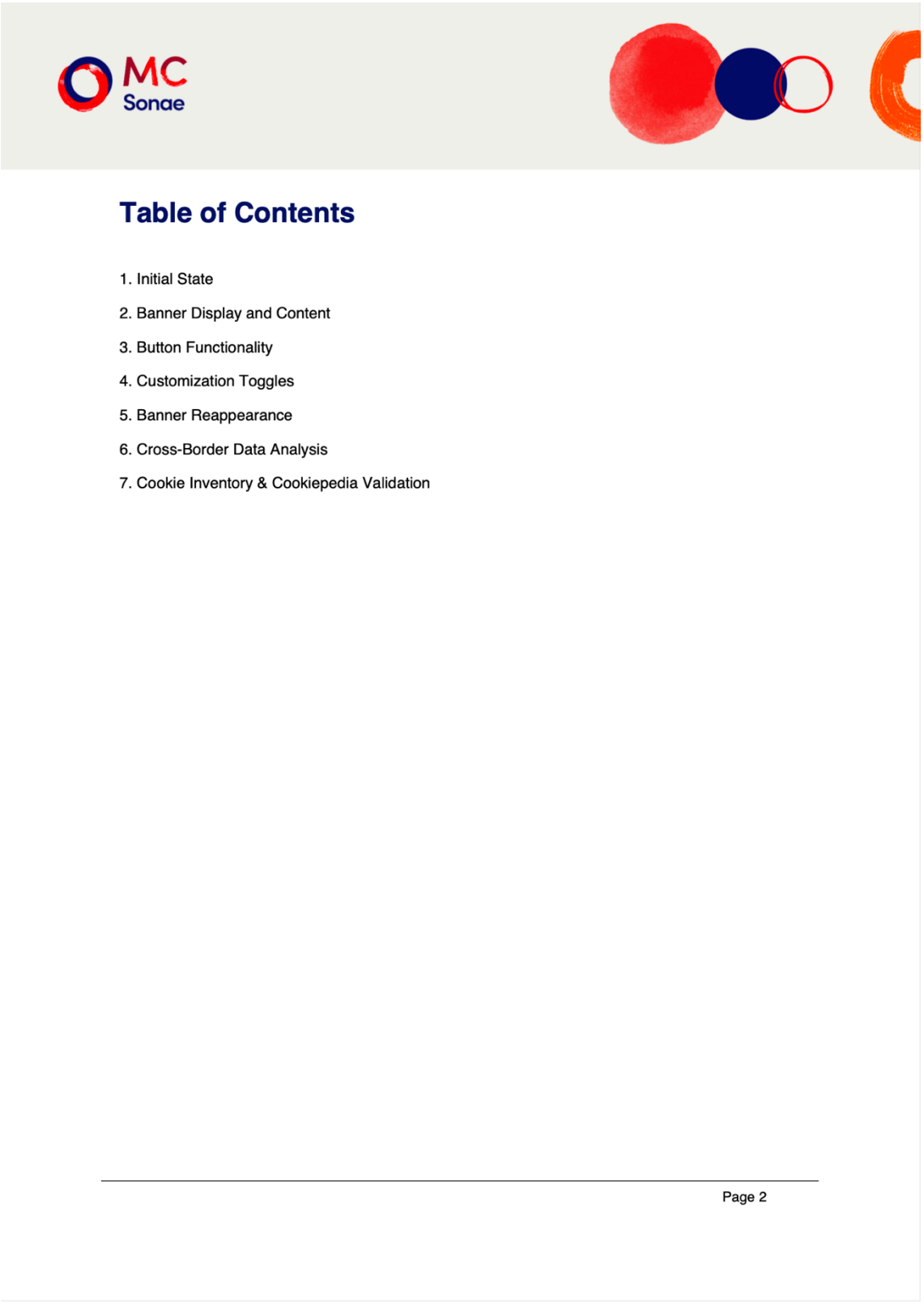
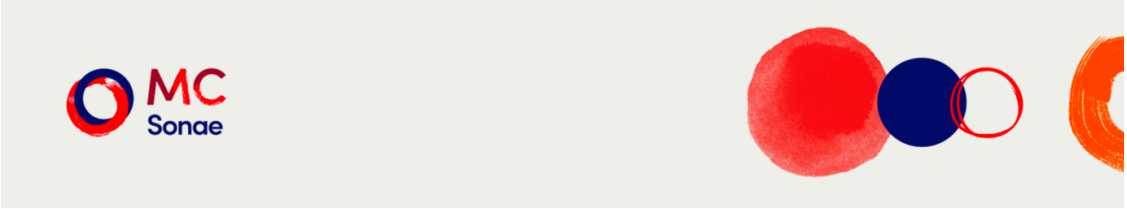


Figure 12 - Illustrative table of contents of the report



1. Initial State

Initial State: OK - Banner is visible and no consent cookie is set.

2. Banner Display and Content

Banner Dimensions: OK - Banner is visible with dimensions (536x283).

Banner Descriptive Text: OK - Banner contains visible descriptive 'cookie' text.

Policy Link: OK - Banner contains a visible link to the cookie/privacy policy.

Allow selection/Personalizar Button: OK - Button for details is visible.

3. Button Functionality

Scenario: Accept All

OK - Banner hidden and all categories consented.

Scenario: Refuse All

FAIL - The 'Refuse All' button was not found or not immediately visible on the first layer of the banner. This may violate GDPR's 'equal ease' principle.

4. Customization Toggles

Scenario: Custom Statistics

OK - Consent set correctly for custom_statistics.

Scenario: Custom Preferences

OK - Consent set correctly for custom_preferences.

Scenario: Custom Marketing

OK - Consent set correctly for custom_marketing.

Scenario: Custom Minimal

OK - Consent set correctly for custom_minimal.

Figure 13 - Illustrative output for initial state, banner content, button, and customization tests

5. Banner Reappearance

Banner Reappearance: OK - Banner correctly reappeared after clearing cookies.

6. Cross-Border Data Analysis

FAIL - Confirmed non-EU data transfers identified

- Domain: c.bing.com, IP: 150.171.27.10, Location: Redmond, Washington, US
- Domain: c.clarity.ms, IP: 13.107.246.76, Location: Redmond, Washington, US
- Domain: cdnjs.cloudflare.com, IP: 104.17.24.14, Location: San Francisco, California, US
- Domain: dev.visualwebsiteoptimizer.com, IP: 34.107.218.251, Location: Kansas City, Missouri, US
- Domain: i.clarity.ms, IP: 4.153.72.49, Location: Boydton, Virginia, US
- Domain: js-agent.newrelic.com, IP: 162.247.243.39, Location: San Francisco, California, US
- Domain: region1.analytics.google.com, IP: 216.239.34.36, Location: Mountain View, California, US
- Domain: region1.google-analytics.com, IP: 216.239.32.36, Location: Mountain View, California, US
- Domain: scripts.clarity.ms, IP: 13.107.246.76, Location: Redmond, Washington, US
- Domain: unpkg.com, IP: 104.18.0.22, Location: San Francisco, California, US
- Domain: www.clarity.ms, IP: 20.250.198.32, Location: Zürich, Zurich, CH

INFO - Confirmed domains were found to be within the EU

- Domain: ams.creativecdn.com, IP: 185.184.8.90, Location: NL
- Domain: analytics-ipv6.tiktokw.us, IP: 96.16.84.14, Location: ES
- Domain: analytics.tiktok.com, IP: 95.100.100.120, Location: PT
- Domain: automaise.com, IP: 20.224.110.156, Location: NL
- Domain: cdn.cquotient.com, IP: 3.160.139.127, Location: PT
- Domain: coltaticstorage.blob.core.windows.net, IP: 20.209.105.139, Location: IE
- Domain: connect.facebook.net, IP: 157.240.212.14, Location: PT
- Domain: consent.cookiebot.com, IP: 2.19.54.11, Location: PT
- Domain: consentcdn.cookiebot.com, IP: 184.24.0.170, Location: ES

Figure 14 - Illustrative output for banner reappearance and cross-border data analysis

- Domain: fonts.googleapis.com, IP: 216.58.209.74, Location: ES
- Domain: fonts.gstatic.com, IP: 142.250.184.163, Location: ES
- Domain: google.com, IP: 142.250.184.14, Location: ES
- Domain: googleads.g.doubleclick.net, IP: 142.250.184.162, Location: ES
- Domain: ib.adnxs.com, IP: 37.252.171.21, Location: DE
- Domain: imgsct.cookiebot.com, IP: 184.24.0.170, Location: ES
- Domain: p.cquotient.com, IP: 54.229.247.198, Location: IE
- Domain: pagead2.googlesyndication.com, IP: 142.250.200.130, Location: ES
- Domain: securepubads.g.doubleclick.net, IP: 142.250.200.98, Location: ES
- Domain: stats.g.doubleclick.net, IP: 66.102.1.157, Location: BE
- Domain: tags.creativecdn.com, IP: 185.76.11.66, Location: ES
- Domain: www.automaise.com, IP: 20.224.110.156, Location: NL
- Domain: www.facebook.com, IP: 157.240.212.35, Location: PT
- Domain: www.google.com, IP: 142.250.200.132, Location: ES
- Domain: www.googleadservices.com, IP: 216.58.215.162, Location: ES
- Domain: www.googletagmanager.com, IP: 142.250.184.168, Location: ES
- Domain: www.googletagservices.com, IP: 142.250.185.2, Location: ES

7. Cookie Inventory & Cookiepedia Validation

Found a total of 183 cookie declarations.

Found 124 unique cookies for validation.

Cookie Name	Declared Category	DB/Cookiepedia Category	Status
AWSALB	Necessary	Necessary	OK
AWSALBCORS	Necessary	Necessary	OK
AWSALBTG	Necessary	Necessary	OK
AWSALBTGCORS	Necessary	Necessary	OK
test_cookie	Necessary	Marketing	MISMATCH
rc::a	Necessary	Marketing	MISMATCH
rc::c	Necessary	Marketing	MISMATCH
JSESSIONID	Necessary	Necessary	OK
c	Necessary	Necessary	OK

Page 5

Figure 15 - Illustrative output for cross-border data analysis, and the initial cookie categorization table

ts	Necessary	Functional	MISMATCH
weird_get_top_level_domain	Necessary	No Info	MISMATCH
byside_cookie_service_level	Necessary	No Info	MISMATCH
rbzid	Necessary	Unknown	MISMATCH
realUserVerifier	Necessary	No Info	MISMATCH
ar_debug	Necessary	Marketing	MISMATCH
CookieConsent	Necessary	Necessary	OK
ARRAffinity	Necessary	Necessary	OK
ARRAffinitySameSite	Necessary	Necessary	OK
ASLBSA	Necessary	Necessary	OK
ASLBSACORS	Necessary	Necessary	OK
BrowserId	Necessary	Functional	MISMATCH
CookieConsentPolicy	Necessary	Necessary	OK
LSKey-c$CookieConsentPolicy	Necessary	Necessary	OK
__anact	Necessary	Functional	MISMATCH
__cq_dnt	Necessary	Functional	MISMATCH
__cqact	Necessary	Functional	MISMATCH
__dc_env	Necessary	No Info	MISMATCH
__dcact	Necessary	No Info	MISMATCH
cqcid	Necessary	Unknown	MISMATCH
cquid	Necessary	Unknown	MISMATCH
dw_dnt	Necessary	Necessary	OK
dwac_#	Necessary	Marketing	MISMATCH
dwanonymous_UID{32}	Necessary	Marketing	MISMATCH
dwsid	Necessary	Necessary	OK
MailingAnonID	Necessary	No Info	MISMATCH
sid	Necessary	Necessary	OK
ssoSession	Necessary	Unknown	MISMATCH
tabHistory_#	Necessary	No Info	MISMATCH
g	Functional	Marketing	MISMATCH
_ga	Statistics	Statistics	OK
_ga_#	Statistics	Unknown	MISMATCH
_gcl_au	Statistics	Marketing	MISMATCH

Page 6

Figure 16 - Illustrative table for the cookie categorization

CLID	Statistics	Functional	MISMATCH
NRBA_SESSION	Statistics	No Info	MISMATCH
cq.recoUUID	Statistics	No Info	MISMATCH
cq.viewCategory	Statistics	No Info	MISMATCH
cq.viewReco	Statistics	No Info	MISMATCH
__cq_uuid	Statistics	Unknown	MISMATCH
_tt_enable_cookie	Statistics	Marketing	MISMATCH
eu01/v.gif	Statistics	No Info	MISMATCH
vwo_apm_sent	Statistics	No Info	MISMATCH
_vwo_uuid_v2	Statistics	No Info	MISMATCH
undefined	Statistics	Marketing	MISMATCH
log/error	Marketing	Unknown	MISMATCH
_fbp	Marketing	Marketing	OK
C	Marketing	Necessary	MISMATCH
uid	Marketing	Unknown	MISMATCH
uuid2	Marketing	Marketing	OK
XANDR_PANID	Marketing	No Info	MISMATCH
_gat	Marketing	Statistics	MISMATCH
_gid	Marketing	Statistics	MISMATCH
IDE	Marketing	Marketing	OK
NID	Marketing	Marketing	OK
pagead/1p-conversion/#/	Marketing	No Info	MISMATCH
pagead/1p-user-list/#	Marketing	No Info	MISMATCH
pagead/gen_204	Marketing	No Info	MISMATCH
pagead/gen_204/	Marketing	No Info	MISMATCH
csi	Marketing	Unknown	MISMATCH
__rtbh.lid	Marketing	Marketing	OK
__rtbh.uid	Marketing	Marketing	OK
cq.anchor	Marketing	No Info	MISMATCH
__cq_bc	Marketing	Unknown	MISMATCH
__cq_seg	Marketing	Functional	MISMATCH
uuid	Marketing	Marketing	OK
tt_appInfo	Marketing	No Info	MISMATCH
tt_pixel_session_index	Marketing	No Info	MISMATCH
tt_sessionId	Marketing	No Info	MISMATCH

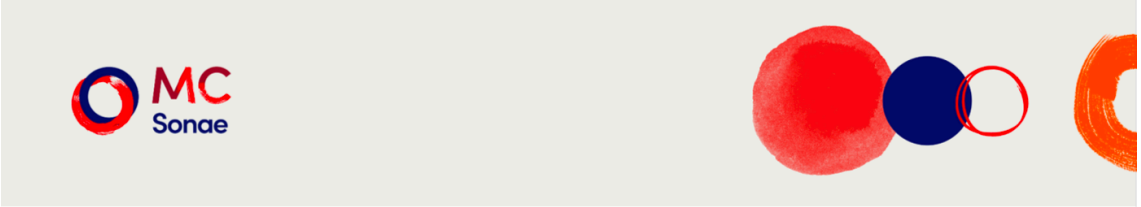
Page 7

Figure 17 - Illustrative table for the cookie categorization

_ttp	Marketing	Marketing	OK
#-#	Marketing	No Info	MISMATCH
iU5q-!O9@[#COOKIETABLE_ADVERTISING#]nbsp;	Marketing	No Info	MISMATCH
LAST_RESULT_ENTRY_KEY	Marketing	Unknown	MISMATCH
nextId	Marketing	Unknown	MISMATCH
requests	Marketing	Unknown	MISMATCH
yt.innertube::nextId	Marketing	Unknown	MISMATCH
yt.innertube::requests	Marketing	Unknown	MISMATCH
ytidb::LAST_RESULT_ENTRY_KEY	Marketing	No Info	MISMATCH
YtIdbMeta#databases	Marketing	No Info	MISMATCH
yt-remote-cast-available	Marketing	Unknown	MISMATCH
yt-remote-cast-installed	Marketing	Unknown	MISMATCH
yt-remote-connected-devices	Marketing	Unknown	MISMATCH
yt-remote-device-id	Marketing	Unknown	MISMATCH
yt-remote-fast-check-period	Marketing	Unknown	MISMATCH
yt-remote-session-app	Marketing	Unknown	MISMATCH
yt-remote-session-name	Marketing	Unknown	MISMATCH
__Secure-ROLLOUT_TOKEN	Marketing	No Info	MISMATCH
__Secure-YEC	Marketing	Unknown	MISMATCH
LogsDatabaseV2:V#IILogsRequestsStore	Marketing	No Info	MISMATCH
remote_sid	Marketing	Unknown	MISMATCH
ServiceWorkerLogsDatabase#SWHealthLog	Marketing	No Info	MISMATCH
TESTCOOKIESENABLED	Marketing	Necessary	MISMATCH
VISITOR_INFO1_LIVE	Marketing	Marketing	OK
YSC	Marketing	Unknown	MISMATCH
byside_webcare_session_tid	Marketing	Functional	MISMATCH
byside_webcare_tuid	Marketing	Marketing	OK
lastExternalReferrer	Marketing	Unknown	MISMATCH

Page 8

Figure 18 - Illustrative table for the cookie categorization



lastExternalReferrerTime	Marketing	No Info	MISMATCH
topicsLastReferenceTime	Marketing	No Info	MISMATCH
sp_ssid	Marketing	Marketing	OK
spid	Marketing	Marketing	OK
syndi_pageid_timestamps	Marketing	No Info	MISMATCH
event/a.gif	Marketing	Unknown	MISMATCH
event/p.gif	Marketing	Unknown	MISMATCH
_gcl_ls	Marketing	No Info	MISMATCH
v1/tiles	Unclassified	Unknown	OK
ttcsid	Unclassified	No Info	OK
ttcsid_CUVG0JJC77U9HMCID3V0	Unclassified	No Info	OK
automaise-botchat-es5	Unclassified	No Info	OK
automaise-botchat-es5-version	Unclassified	No Info	OK
_taggstar_cfg_continentept	Unclassified	No Info	OK
_taggstar_exps	Unclassified	No Info	OK
_taggstar_ses	Unclassified	Marketing	MISMATCH
_taggstar_vid	Unclassified	Necessary	MISMATCH
waap_id	Unclassified	No Info	OK
event/h.gif	Unclassified	Unknown	OK

Page 9

Figure 19 - Illustrative table for the cookie categorization