A Privacy Tool for Detecting and Preventing Youtube Data Leaks

Jiaxi Chen

Master of Science
Cyber Security, Privacy and Trust
School of Informatics
University of Edinburgh
2024

Abstract

This work presents the design and evaluation of a Chrome browser-based plugin aimed at detecting privacy leaks in YouTube videos, specifically tailored for content creators on video-sharing platforms. The plugin addresses critical privacy issues by identifying unintended facial appearances, sensitive text, audio disclosures, and explicit content within video content. The evaluation of this work was conducted through two user studies, where participants were engaged in hands-on testing of the plugin's functionalities. These studies employed questionnaires to gather user feedback, focusing on both the effectiveness of the privacy detection features and the overall user experience. Based on the responses, we made several targeted improvements to enhance the plugin's performance and usability. The final version is widely regarded as effectively addressing the current privacy leak issues.

Research Ethics Approval

This project obtained approval from the Informatics Research Ethics committee.
Ethics application number: 963374
Date when approval was obtained: 2024-07-01
The participants' information sheet and a consent form are included in the appendix.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*Jiaxi Chen*)

Acknowledgements

I would like to express my heartfelt gratitude to my supervisor, Nadin Kökciyan, for her unwavering support throughout the entire project, from development to completion. Her invaluable guidance, technical expertise, and constant encouragement were instrumental in keeping me motivated and focused. Whether through our weekly meetings or by answering my questions whenever I was uncertain, her dedication played a crucial role in refining my project.

I also want to extend my deepest thanks to my parents, whose unconditional support has been the foundation of my journey. Without them, I would not have come this far. Lastly, I am profoundly grateful to my friends for their encouragement, which helped me persevere through challenges, and to someone special who has supported me from afar. Your belief and support in me made all the difference.

Table of Contents

Chapter 1

Introduction

1.1 Motivation

With the rise of social media and video-sharing platforms, more and more individuals and organizations are using these platforms to share video content [49]. However, these videos may contain sensitive information, especially if they haven't been properly screened [7]. Privacy breaches can not only affect an individual's safety and privacy but can also lead to legal and ethical issues [24]. Thus, finding effective ways to detect and prevent privacy leaks in videos has become an urgent and important topic. Current mainstream privacy protection solutions mostly focus on text and images while there's a gap in handling video content [5]. Existing video detection tools tend to concentrate on specific issues, like explicit content detection, and lack comprehensive solutions, making it hard to effectively identify and address various types of privacy violations. In addition, users must rely on multiple tools to meet complex needs because there is currently a lack of tools that integrate multiple detection functions, which not only complicates the process but also may ignore some potential privacy risks. Therefore, video creators need a one-stop tool that can provide video privacy reminders and provide users with a comprehensive and interface-optimized solution.

To solve the concerns listed above, we developed a Chrome extension tool that works to provide a comprehensive solution. The tool supports users in identifying and resolving potential privacy problems in videos by integrating several features like face detection, text detection, audio analysis, and explicit content detection. It contributes to the enhancement of video content compliance and security. By creating this tool, we seek to increase public awareness of the importance of protecting private video content.

1.2 Problem Statement

This plugin focuses on addressing the key challenges video creators face in privacy and content management, improving solutions for the following four issues:

- **Unintended Facial Appearance:** Videos could accidentally reveal faces without authorisation, violating rules about privacy and perhaps causing security risks.
- **Text Revealing Private Information:** The video could reveal private information that is too sensitive, including location or identity.
- **Unintended Privacy Disclosure in Audio:** Privacy violations could result from speech or background noise unintentionally disclosing sensitive information or private talks.
- **Explicit Content Exposure:** Explicit content, once published, will be disseminated quickly and widely, posing a potential threat to and infringement of personal privacy.

1.3 Research Questions

This project aims to design a tool that actively assists video creators in detecting and preventing privacy leaks through a simple and user-friendly interface with powerful features. The research questions can be broken down into the following research questions:

- **RQ1:** Could we design a tool to detect multiple issues at a time?
- **RQ2:** How a tool can be implemented by integrating existing technologies?
- **RQ3:** Could such a tool would be usable by users?
- **RQ4:** What criteria should be used to evaluate plug-in designs, including user experience, functional effectiveness, and detection accuracy, and how can these criteria and methods be determined?

1.4 Project Overview

This plugin uses a front-end and back-end separation architecture. The frontend is primarily responsible for displaying the user interface and handling user interactions.

When users input url content in the text area, the frontend captures the input and sends it to the backend through an interface. The frontend then dynamically updates the interface based on the suggestions and filtering results returned by the backend, displaying relevant prompts and content. The backend is mainly responsible for parsing specific patterns in the user input data, such as video URLs, invoking APIs, and interacting with the database. The database is used for encryption, decryption, and storage of data. For user-centered tools, evaluation is also crucial. We designed two user studies to gather feedback from our target users. The first study focused on the prototype of the plugin to establish a feedback loop, thereby improving the overall design. The second study involved a similar survey on the final version of the plugin to assess the results of the improvements. Figure 1.1 outlines the project.

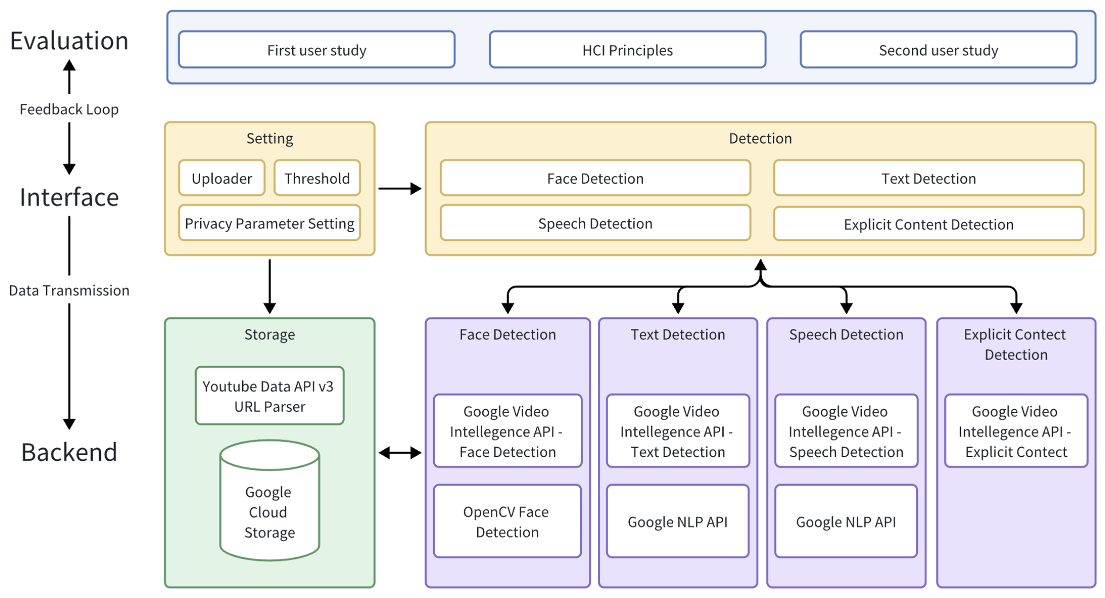


Figure 1.1: Project Overview

1.5 Dissertation Structure

Chapter 2 discusses the existing situation, what comparable software is available already, and a critique of its drawbacks. Chapter 3 explains the full design process, including user interface, software selection, design concepts, and feedback loop-based improvement. The functional logic and architecture of the back-end system are described in Chapter 4. The assessment procedure is described in Chapter 5, along with a comparison and analysis of the findings from two user studies. Chapter 6 provides an overview of our existing research and recommendations for further development.

Chapter 2

Background

2.1 Privacy in Social Media

Privacy means the right not to have personal information, behaviour or statements accessed, used or disclosed by others without authorisation [3]. Such information typically includes, but is not limited to, personally identifiable information, financial data, health records, geographic location, social relationships, online behaviour, and so on. With the popularity of social media, the importance of privacy is ever more pronounced, as personal information may be collected, stored and used for a wide range of purposes, including both legitimate uses and activities that may infringe on an individual's rights [57].

Social media platforms (e.g., Facebook, Instagram, YouTube, TikTok, etc.) provide users with the opportunity to share their lives, express their opinions, and build social networks. However, these platforms have also become high-risk areas for privacy breaches. What users share on social media may unintentionally expose personal information, such as geographic location, family members, and daily habits [50].

2.2 Privacy Leakage Risks

Facial information in images and videos: When users post content that includes their own or others' facial information without proper protection, it may be used for facial recognition, identity theft, or other illegal purposes. For example, TikTok faced a class-action lawsuit for collecting facial recognition data without users' consent and applying it to its algorithms, thereby violating Illinois' strict Biometric Information Privacy Act (BIPA) [48].

Text information, such as backgrounds and landmarks in videos or images, can be used by malicious actors to determine a user's geographic location, potentially leading to stalking or targeted crimes. Personal data like phone numbers, email addresses, or other identifying details shared in content can easily be exploited for online fraud, phishing, or identity theft [13]. Additionally, unintentionally recorded conversations or background audio in videos may expose sensitive personal information. A study by Friedland and Choi [21] demonstrates how semantic computing can be used to infer geographic locations from data sources like YouTube and Flickr videos using tag descriptions, highlighting the significant privacy risks and potential for "cybercasing" attacks [22].

2.3 Proposed Solutions In The Past

In response to growing concerns over privacy on social media, several models and frameworks have been proposed to understand and mitigate privacy risks. Early on, Westin introduced the concept of "privacy as control" in 1967 where privacy is seen as the individual's right to control information about themselves [56]. This notion laid the groundwork for many privacy protection tools that emphasize user control over their data. Palen and Dourish (2003) expanded on this by introducing the "control paradox" highlighting the conflict between users' desire to share information and their need for privacy [43].

Building on these theories, Marwick and Boyd introduced the "networked privacy" model [37, 36, 38], which shifts the focus from individual control to the networked nature of privacy. They argue that privacy is managed not only by individuals but also through navigating the complex social contexts and relationships on platforms like Facebook and Twitter. The model emphasises that "context collapse" or the unintentional exposure of content intended for one audience to another [37], is a common cause of privacy violations on social media.

Many privacy protection tools have been developed with guidance from these theoretical frameworks. Social media platforms such as Facebook and Instagram allow users to manage who can see their post through privacy settings. These settings, which are predicated on the idea of "privacy as control" face criticism for being unduly intricate and challenging to use [51]. In addition to measures focused on the user, automatic techniques for detecting and protecting privacy have also been developed [55]. To identify and prevent the sharing of sensitive information in text and photographs, Facebook, for example, uses machine learning algorithms [30]. Similar to this, Instagram automatically flags offensive or delicate content before it is shared publicly by using content review tools. Google Photos uses face recognition technology to blur or mask faces in photographs, preserving users' identities in shared photos [12]. These solutions, however, have been built to handle static content, like text and images, and are ineffective for dynamic content, like videos, where context collapse and the fluidity of video sharing pose more privacy problems [37].

YouTube has implemented a variety of efforts to support creators in safeguarding their privacy in recent years. The ability to mark videos as "Private" or "Unlisted" which limits who can watch the content, is crucial among these [4]. In addition, YouTube has options to block particular people, hide subscriber figures, and disable comments. The site offers tools for creators who are worried about unauthorised content, such reuploads of their videos, to report and remove it [52]. The goal of these technologies is to assist creators in more securely managing their internet presence.

Additionally, a number of technologies and tools for video privacy detection have been developed, including Microsoft Azure Video Indexer [39] and Google Cloud Video Intelligence [45]. These tools analyse video content and discover critical information using cutting-edge AI and machine learning algorithms. However, these tools often require programming knowledge due to their dependency on API interfaces, which limits their widespread availability [47]. Additionally, their lack of user-friendly interfaces hinders acceptance among video creators, making them less suitable for general use [33].

2.4 Existing Problems

Videos have a larger potential for violating privacy than text or photos. Unlike static content, videos frequently include more sensitive information, such as speech, surroundings, facial features, and unintentional movements [53]. As a result, there is a higher risk of privacy violations since malicious actors can more easily extract private information from videos. For instance, a 2010 study by Friedland and Choi highlighted the serious privacy threats posed by video sharing by demonstrating how semantic computing could be used to infer geolocation from video content on platforms like YouTube and Flickr [22].

Furthermore, a study by Acquisti, Gross, and Stutzman investigated that in comparison with text and images, visual and audio information in videos can unintentionally reveal more personal information than intended, raising the risk of privacy breaches [51, 1]. According to study conducted by De Hart and colleagues, recordings may reveal a variety of personal details, including as a person's routines, everyday activities, and even the inner workings of their family, all of which are frequently filmed unintentionally [18].

Nevertheless, the utility of current video privacy protection tools, such as CapCut and Adobe Photoshop, remains limited. While these programs offer powerful editing and privacy protection features, they often require a substantial time investment to learn and master. The complexity of these tools can be a significant barrier, especially for average users who may lack the technical expertise or time to use them effectively. This steep learning curve diminishes the practical usefulness of these tools for many potential users.

Beyond this, there are still gaps in platforms like YouTube's privacy policies for content creators, particularly the lack of pre-upload and post-upload content recognition tools. This gap allows anyone to publish videos that can quickly go viral and reach large audiences without properly analyzing potentially sensitive data such as faces, text, and audio [9]. Due to the absence of pre-upload screening, creators risk inadvertently sharing explicit or personal content, which could violate community guidelines or infringe on privacy rights [34]. Moreover, YouTube's post-upload analysis is often limited, passive, and reliant on user reports rather than proactive detection, further increasing the likelihood that private information will remain publicly accessible [4]. This approach not only adds to content creators' workload for content monitoring but also increases the risk of ongoing privacy issues. Therefore, more comprehensive and proactive privacy solutions are essential to protect YouTube creators and their audiences.

Advanced AI tools like Google Cloud Video Intelligence and Microsoft Azure Video Indexer are designed to detect privacy risks [39, 45]. However, they are often inaccessible to the general public because they rely on APIs and require programming knowledge. The lack of intuitive, user-friendly interfaces further complicates their use, making it difficult for non-technical users to protect their privacy effectively.

Chapter 3

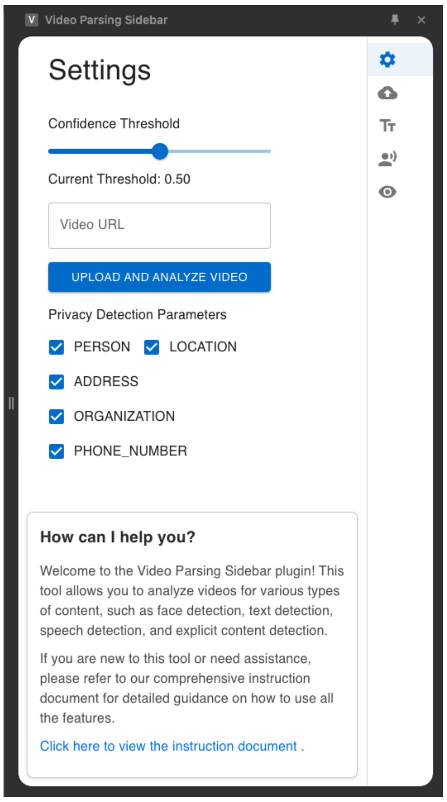
User Interface

3.1 Software Selection

The software is a browser-based plug-in tool that was developed specifically as a Chrome extension. This is because it allows for seamless integration with the browser, providing an intuitive and easily accessible user experience. The Chrome extension runs directly within the browser environment, allowing users to interact with the tool without leaving their current web session. This integration is critical to ensuring that the software remains lightweight and highly accessible [11]. Additionally, it leverages the browser's own resources to handle concurrent tasks without relying on additional server resources. This model embeds the tool directly into the browsing experience, facilitating a smoother and more efficient workflow [26]. Figure 3.1 shows the final interface of setting page (main page).

In terms of compatibility, the Chrome extension offers significant advantages. As one of the most widely used browsers globally, Google Chrome ensures broad user coverage. Additionally, Chrome's cross-platform nature allows our extension to perform consistently across operating systems like Windows, macOS, and Linux. This consistency is crucial for providing a uniform user experience [35], especially when the extension is used across various devices and system environments.

The extension's interface and interactions are built with modern web technologies, including Manifest v3 and React. Manifest v3, the latest standard for Chrome extensions [26], was chosen for its enhanced security and performance. It introduces a Service Worker-based architecture, reducing memory usage and improving the efficiency of background tasks. This architecture efficiently handles concurrent requests without blocking the user interface, offering significant performance advantages for managing


Video Parsing Sidebar

Settings

Confidence Threshold

Current Threshold: 0.50

Video URL

UPLOAD AND ANALYZE VIDEO

Privacy Detection Parameters

✓ PERSON ✓ LOCATION

✓ ADDRESS

✓ ORGANIZATION

✓ PHONE_NUMBER

How can I help you?

Welcome to the Video Parsing Sidebar plugin! This tool allows you to analyze videos for various types of content, such as face detection, text detection, speech detection, and explicit content detection.

If you are new to this tool or need assistance, please refer to our comprehensive instruction document for detailed guidance on how to use all the features.

Click here to view the instruction document .

Figure 3.1: Setting Page

numerous asynchronous tasks [44]. Additionally, the strict control over permissions and declarative management of resource requests in Manifest v3 enhance the extension's security and stability in complex application scenarios.

React, a JavaScript library developed by Facebook, is the core technology for frontend development [23]. It was chosen for its component-based architecture and virtual DOM, which not only simplify development but also deliver exceptional performance in handling high-concurrency user interactions. The virtual DOM minimizes direct DOM operations, significantly reducing the browser's rendering load and ensuring a smooth user interface [2]. React's component-based design allows us to decompose complex user interfaces into reusable modules, enhancing code maintainability and extensibility. Additionally, React's robust ecosystem, including Hooks and the Context API, simplifies state management and side-effect handling, making development in complex application scenarios more efficient and intuitive.

For frontend UI framework selection, we opted for Material-UI (MUI) due to its ability to rapidly build user interfaces that adhere to modern design standards while maintaining a high degree of customization flexibility. MUI, built on React, supports responsive design, ensuring consistent performance of the extension across various devices and screen sizes. Additionally, MUI's pre-built components are optimized for high-concurrency user interactions, enhancing the stability and efficiency of the user experience [6].

3.2 Design Principles

In developing the UI for this software, the primary focus was on creating a functional, user-friendly tool that even novice users could easily navigate. Established design principles guided the creation of an interface that is both effective and intuitive. This section outlines the key design guidelines that shaped the development of the UI, ensuring it meets the needs of all users.

The design of this browser-based tool follows several fundamental principles derived from modern UX and UI design concepts. A key approach was integrating active user involvement in decision-making processes to enhance interaction and control. This approach aligns with the Human In the Loop (HITL) framework [17], as well as established guidelines like Jakob Nielsen's "10 Heuristics for User Interface Design" and Yablonski's "Laws of UX" [41, 58]. These principles ensure the tool is accessible to both novice and experienced users.

3.2.1 Active User Involvement and Control

Emphasizing active user involvement ensures the system responds to input and provides real-time feedback [17]. For instance, before executing irreversible actions like deleting data, the system prompts users with a clear warning, allowing them to confirm or cancel the operation. Interactive tutorials and contextual help guide new users through complex tasks, making the tool accessible to all experience levels.

3.2.2 Use the User's Language

The interface is designed to be intuitive by using familiar terms and concepts. We avoided technical jargon and opted for language that resonates with the user's everyday experiences. For instance, common actions like "Parse Video" and "Upload and Analyse Video" are labeled in ways that align with user expectations, reducing the learning curve and enabling people up to concentrate on their work instead of figuring out how to use the interface.

3.2.3 Ensure User Flexibility and Control

Providing users with the ability to easily reverse actions is crucial, and this principle guides the global video presentation and saving of results, allowing users to jump around within the tool without losing progress, thus preventing potential errors when using different features.

3.2.4 Make design minimalist and elegant

Given that this tool functions as a browser plugin rather than a standalone application, we've designed the interface to be clean and focused. We prioritize presenting only essential information and controls to the user, minimizing clutter that could distract or overwhelm. This minimalist design approach not only improves usability but also enhances performance by eliminating unnecessary visual elements that could slow down the interface.

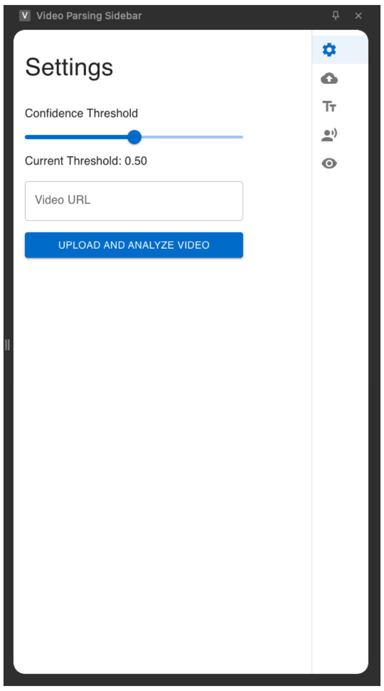
3.3 Design Process and Improvements

We structured the functional design process into five distinct parts: *SettingsPage*, *FaceDetection*, *TextDetection*, *SpeechDetectionPage*, and *ExplicitDetectionPage*. Each function was intentionally separated to minimize coupling between functional pages, making the tool easier for users to learn. To refine the design, we conducted user research with YouTube creators, gathering feedback to guide further improvements. This research was essential for establishing a feedback loop, ensuring that the plugin meets the needs of its target users. In this section, we outline the design process and the improvements implemented based on user feedback.

3.3.1 SettingsPage

SettingsPage is a core component of the plugin, allowing users to configure its behavior, such as setting confidence thresholds, uploading video URLs for analysis, and selecting privacy detection parameters. During the design process, we prioritized simplicity and intuitive user experience, continuously refining the interface based on user feedback. The following section provides a detailed analysis of the *SettingsPage* functionality and the design improvements made.

1. **Confidence Threshold Setting**



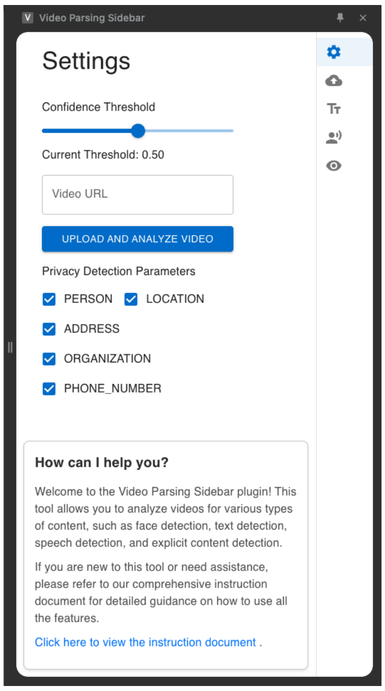
Settings

Confidence Threshold

Current Threshold: 0.50

Video URL

(a) Initial design of Setting Page



Settings

Confidence Threshold

Current Threshold: 0.50

Video URL

Privacy Detection Parameters

- [x] PERSON
- [x] LOCATION
- [x] ADDRESS
- [x] ORGANIZATION
- [x] PHONE_NUMBER

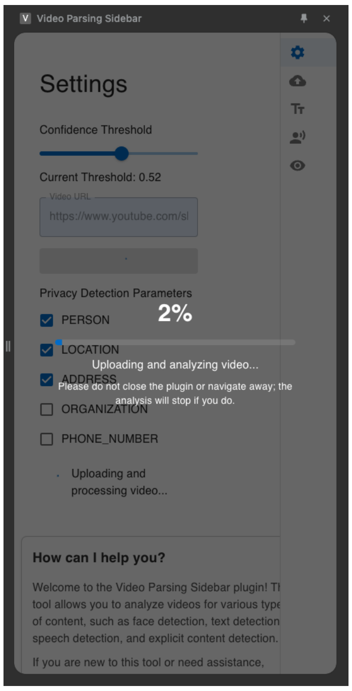
How can I help you?

Welcome to the Video Parsing Sidebar plugin! This tool allows you to analyze videos for various types of content, such as face detection, text detection, speech detection, and explicit content detection.

If you are new to this tool or need assistance, please refer to our comprehensive instruction document for detailed guidance on how to use all the features.

Click here to view the instruction document .

(b) Final design of Setting Page



Settings

Confidence Threshold

Current Threshold: 0.52

Video URL https://www.youtube.com/s

Privacy Detection Parameters

- [x] PERSON
- [x] LOCATION
- [x] ADDRESS
- [ ] ORGANIZATION
- [ ] PHONE_NUMBER

Uploading and analyzing video...

Please do not close the plugin or navigate away; the analysis will stop if you do.

Uploading and processing video...

How can I help you?

Welcome to the Video Parsing Sidebar plugin! Tl tool allows you to analyze videos for various type of content, such as face detection, text detection speech detection, and explicit content detection.

If you are new to this tool or need assistance,

(c) Final design of loading component

Figure 3.2: Comparison between the initial and final designs of the Setting Page

In Figure 3.2, we used a `slider` component to allow users to adjust the confidence threshold in the range of 0 to 1. This threshold directly affects the sensitivity of the plugin in identifying suspicious content during video analysis.

2. **Video URL Upload and Analysis**

   Users can upload videos to the server for analysis by entering the video URL. We used a `TextField` component to accept user input and trigger the upload and analysis process when the "Upload and Analyse Video" button is clicked. During the upload process, we introduced a *CircularProgress* component and a *GlobalLoadingOverlay* to visually display the loading status, preventing user confusion during the wait. After the backend successfully receives and analyses the video, the analysis results are returned in JSON format and stored for use by other functional pages (such as *FaceDetection*, *TextDetection*, etc.).

3. **Privacy Detection Parameter Selection**

   In Figure 3.2b, the plugin provides users with multiple privacy detection parameters, such as detecting "PERSON", "LOCATION" and "ADDRESS" etc. Users can customize the types of privacy detection during the analysis process by checking the checkboxes. This design provides flexibility, allowing users to customize the depth and scope of analysis according to their needs.

During user research, we gathered feedback from several YouTube creators and made corresponding improvements to the *SettingsPage*. These improvements focused on enhancing the fluidity of the user experience and the clarity of the functionality.

3.3.2 Improvements Based on User Feedback

1. **Loading State and Feedback Improvements**

   In the initial design, users lacked clear progress feedback during video upload and analysis, which could lead to doubts about the operation. To address this, we added the *GlobalLoadingOverlay* component in Figure 3.2c so that users can clearly see that the system is processing when background tasks are running. Through more intuitive loading feedback and error message prompts, users can better understand the current status of the operation, reducing operational anxiety.

2. **Privacy Detection Parameter Improvements**

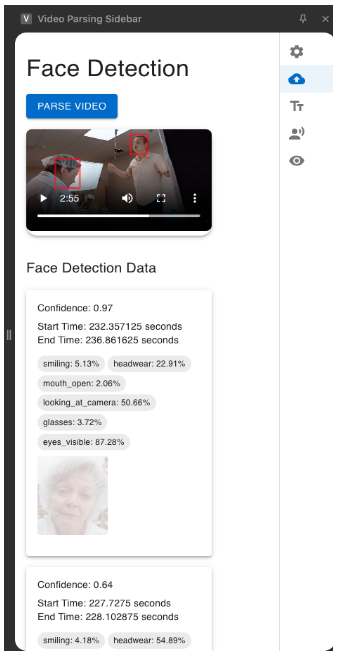
   Users prefer to choose what they want to detect, rather than checking all of them leading to too much clutter and difficulty in identifying the problem, so we set up a parameter selection section where all privacy detection parameters are presented in the form of checkboxes, making the detection results more accurate.

3. **Video URL Upload Experience Optimization**

   Based on user feedback, we optimized the user experience of the video URL input box. For example, when users enter an invalid URL, the system will promptly provide feedback, prompting them to check the correctness of the input content. This improvement increases the success rate of video uploads and reduces operational failures due to incorrect input.

3.3.3 Face Detection

The initial design of *FaceDetection* revolved around the following core tasks: capturing faces in the video, displaying bounding boxes around the faces, and displaying the appearance time of the faces on a timeline. These tasks ensure that users can easily identify and analyse the facial content in the video.


Video Parsing Sidebar

Face Detection

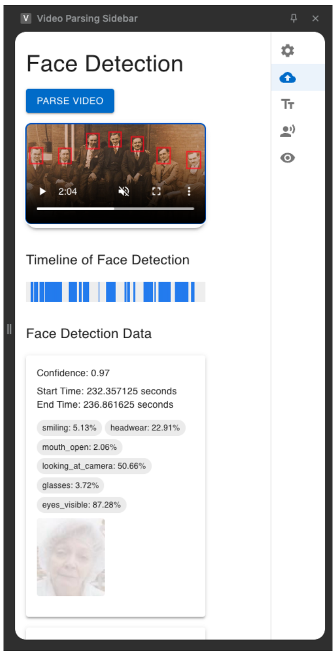
PARSE VIDEO

Face Detection Data

Confidence: 0.97
Start Time: 232.357125 seconds
End Time: 236.861625 seconds
smiling: 5.13% headwear: 22.91%
mouth_open: 2.06%
looking_at_camera: 50.66%
glasses: 3.72%
eyes_visible: 87.28%

Confidence: 0.64
Start Time: 227.7275 seconds
End Time: 228.102875 seconds
smiling: 4.18% headwear: 54.89%

(a) Initial design of Face Detection Page


Video Parsing Sidebar

Face Detection

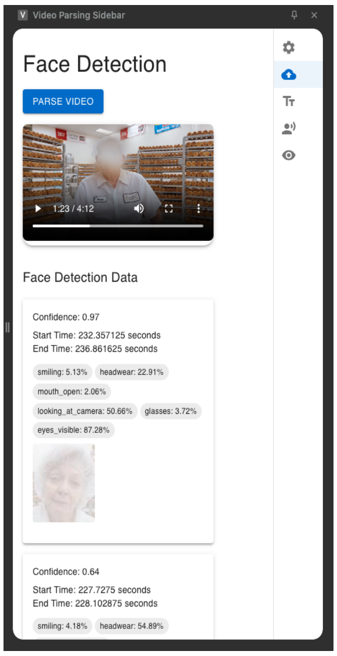
PARSE VIDEO

Timeline of Face Detection

Face Detection Data

Confidence: 0.97
Start Time: 232.357125 seconds
End Time: 236.861625 seconds
smiling: 5.13% headwear: 22.91%
mouth_open: 2.06%
looking_at_camera: 50.66%
glasses: 3.72%
eyes_visible: 87.28%

(b) Face Detection with Timeline


Video Parsing Sidebar

Face Detection

PARSE VIDEO

Face Detection Data

Confidence: 0.97
Start Time: 232.357125 seconds
End Time: 236.861625 seconds
smiling: 5.13% headwear: 22.91%
mouth_open: 2.06%
looking_at_camera: 50.66% glasses: 3.72%
eyes_visible: 87.28%

Confidence: 0.64
Start Time: 227.7275 seconds
End Time: 228.102875 seconds
smiling: 4.18% headwear: 54.89%

(c) Final design of Face Detection Page

Figure 3.3: Comparison between the initial, final, and mosaic designs of Face Detection Page

1. **Face Detection and Capture**

   The front end receives the JSON content returned by the Google Cloud Vision API and extracts the `Face_Frame` class to handle the facial information detected in each frame of the video. It calculates the actual coordinates of the face bounding boxes based on the video resolution (`video_height` and `video_width`). `Face_Track` is used to represent a complete facial trajectory detected in the video. A trajectory consists of multiple `Face_Frame` objects.

2. **Face Bounding Box Display**

   While the video is playing, the plugin draws the bounding boxes returned by the API in each frame. These bounding boxes are overlaid on the video as red rectangles, using the canvas's `CanvasRenderingContext2D`. We use the methods provided by the canvas to directly draw the boxes, visually representing the face positions in the video.

   To ensure the bounding boxes accurately cover the faces, we dynamically adjust the size and position of the bounding boxes based on the video's resolution. This is achieved by converting the standardized coordinates returned by the API to the actual video dimensions.

3. **Face Timeline Visualization**

   Figure 3.3b shows the function to help users better understand the appearance and disappearance of faces in the video, we implemented a face display function on the timeline. Users can quickly check whether a face appears at a particular moment through the timeline and click on the mark on the timeline to jump directly to the corresponding time point. This feature marks the time periods when faces appear on the timeline by traversing the face data in each frame. The color and length of the marks correlate with the frequency and duration of face appearances, helping users quickly identify dense areas of face appearances in the video.

After initial user testing, we collected user feedback on the *Face Detection* feature and made several design improvements, the details will be listed in Chapter 5. These improvements focused on enhancing the accuracy of face detection and the display effect of the bounding boxes, improving users' experience of identifying faces in the video.

1. **Optimization of Bounding Box Display**

   In the initial design, users reported that the bounding box sometimes failed to accurately cover the face, especially in fast-moving scenes. To address this, we optimized the bounding box drawing logic. We used the `useEffect` hook to listen to the video's `play` event and used `requestAnimationFrame` to process the video frame by frame. The `getFacesAtTime` function retrieves the face position corresponding to the current frame time from the detection data, ensuring that the bounding box can keep up with face movement under high-frequency processing. In this way, we can process up to 30 frames of video per second, significantly improving the accuracy of the bounding boxes.

   Additionally, we introduced a gradual bounding box transition effect, allowing the bounding box to follow the face more smoothly during face movement in the video, reducing the visual jumping effect. We used `previousFacePositions` object to store each face's position in the previous frame. The `drawFaceBoundingBox` function uses interpolation to smoothly transition the position, width, and height of the bounding box. This effectively reduces the visual jumping effect, allowing the bounding box to follow the face more smoothly.

Based on users' privacy protection needs, we added the functionality of using OpenCV API to mosaic the faces in the video instead of displaying the bounding boxes. Mosaic design style as shown in Figure 3.3c.

2. **Timeline Display Improvements**

   In the initial design, users found it challenging to accurately locate the time point when faces appeared on the timeline. We improved this by changing the face markers on the timeline to clickable thumbnails. After users click on the thumbnail, the video will jump directly to that time point and pause, allowing users to review the content carefully.

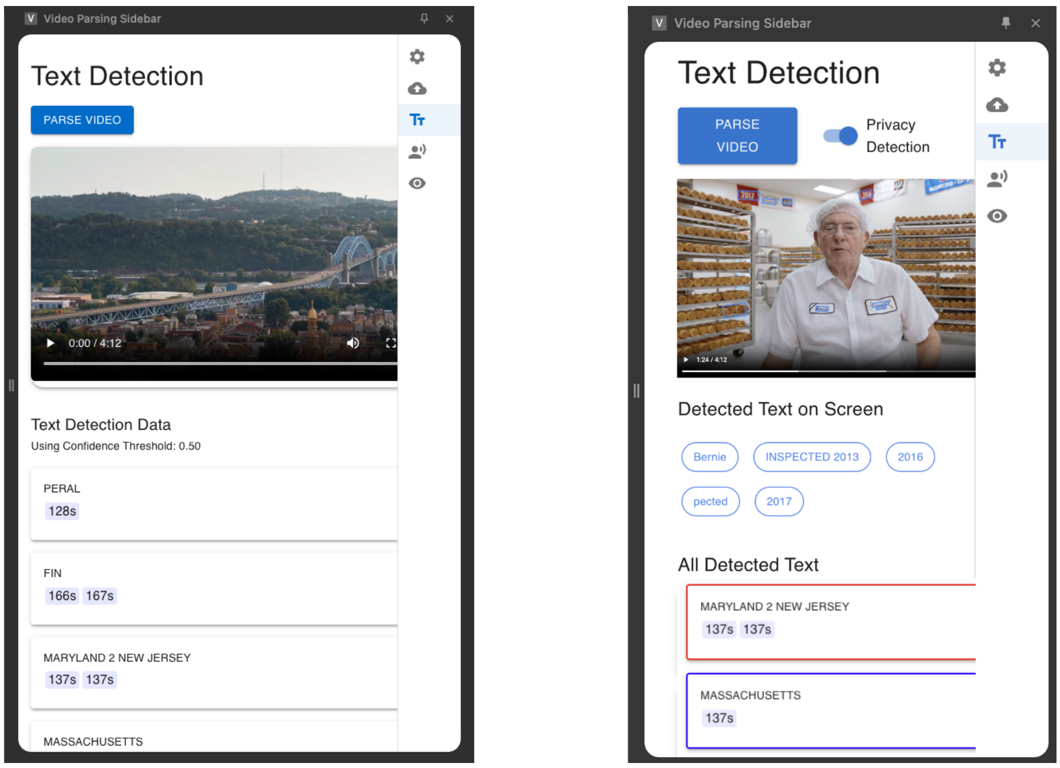
   This improvement not only increased users' efficiency in locating faces but also enhanced the interactive experience, making the face detection feature more intuitive and easy to use.

3.3.4 Text Detection

The initial design of *Text Detection* revolved around the following core tasks: capturing text in the video, displaying bounding boxes around the text, and displaying the appearance time of the text on a timeline. These tasks ensure that users can easily identify and analyse the textual content in the video, see Figure 3.4.

1. **Text Detection and Capture**

   The plugin leverages the Google Cloud Vision API to provide highly reliable and accurate text recognition capabilities, enabling it to identify text in videos and return precise bounding box information. The plugin primarily utilizes the `Text_Segment` class to retrieve all text segments in the video, including the start and end times of each segment, as well as the bounding box information for each frame within the segment. The `Text_Frame` class is then used to access each frame in the video and record the bounding boxes of the detected text in that frame. Additionally, we have introduced a *Privacy Detection* feature based on the Google Cloud Natural Language API, designed to analyse the detected text content and identify and flag potentially sensitive information that may involve privacy concerns.


Video Parsing Sidebar

Text Detection

PARSE VIDEO

0:00 / 4:12

Text Detection Data

Using Confidence Threshold: 0.50

PERAL
128s

FIN
166s 167s

MARYLAND 2 NEW JERSEY
137s 137s

MASSACHUSETTS

Video Parsing Sidebar

Text Detection

PARSE VIDEO

Privacy Detection

1:24 / 4:12

Detected Text on Screen

Bernie INSPECTED 2013 2016 pected 2017

All Detected Text

MARYLAND 2 NEW JERSEY
137s 137s

MASSACHUSETTS
137s

(a) Initial design of Text Detection (b) Final design of Text Detection

Figure 3.4: Comparison between the initial and final designs of Text Detection Page

2. **Text Bounding Box Display**

   While the video is playing, the plugin draws the text bounding boxes returned by the API in each frame. These bounding boxes are overlaid on the video as blue rectangles, visually representing the positions of the text in the video. This is achieved by referencing the video element through `videoRef` and the canvas element through `canvasRef`. The canvas is used to draw the text bounding boxes on the video, for text bounding box design styles, see Figure 3.4b.

   To ensure that the bounding boxes accurately cover the text, we dynamically adjust the size and position of the bounding boxes based on the video's resolution. This is done by converting the standardized coordinates returned by the API to the actual video dimensions. The `drawTextBoundingBoxes` function listens to the video's `play` event and draws the text bounding boxes in real-time on each frame. By calculating and drawing the bounding boxes based on the text position and size information returned by the API, users can intuitively view the text content in the video. The `drawBoundingBoxes` function is called continuously during video playback, obtaining the text bounding boxes at the current time point and drawing them on the canvas. The canvas is accessed through the `CanvasRenderingContext2D` object, which provides various methods for drawing on the canvas, including those needed to draw polygons for text detection.

After initial user testing, we collected user feedback on the *Text Detection* feature and made several design improvements. These improvements focused on enhancing the accuracy of text detection and the display effect of the bounding boxes, improving users' experience of identifying text in the video.

1. **Design Improvements Based on User Feedback**

   Privacy Detection Feature: *Privacy Detection* feature to meet users' requirements for detecting text with potential privacy leaks, see Figure 3.4b.

   Optimization of Bounding Box Display: In the initial design, users reported that the bounding box sometimes failed to accurately cover rapidly changing text. To address this, we optimized the bounding box drawing logic. The `previousTextPositions` object stores each text's position in the previous frame and smoothly transitions the position, width, and height through interpolation, allowing the bounding box to follow the text more smoothly during movement.

   The text in the current screen

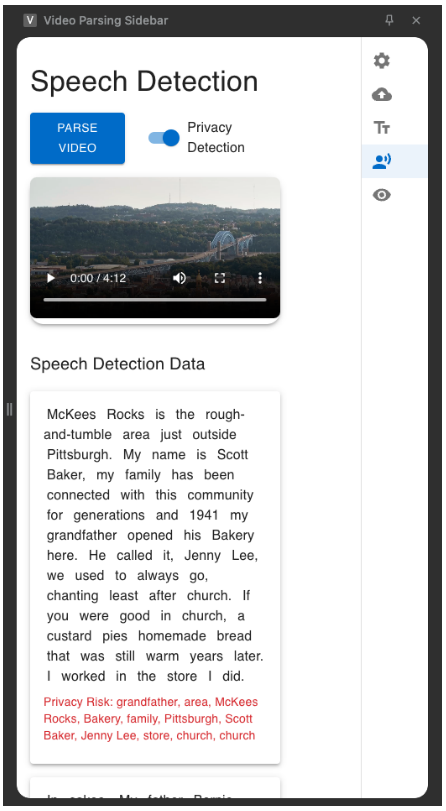
   To help users clearly see which words may be leaked, we display the text on the current screen in real time to improve the user's experience, see Figure 3.4b.

3.3.5 Speech Detection

The initial design of *Speech Detection* revolved around the following core tasks: capturing speech in the video, displaying the converted text, and displaying the appearance time of the speech on a timeline. These tasks ensure that users can easily identify and analyse the speech content in the video.

1. **Initial Design and Functional Implementation**

   We chose the Google Cloud Speech-to-Text API for speech detection and conversion. The Google Cloud Speech-to-Text API provides efficient speech recognition capabilities, converting speech in videos to text and providing timestamp information. This allowed us to quickly implement the speech detection feature and ensure a high level of recognition accuracy.



McKees Rocks is the rough-and-tumble area just outside Pittsburgh. My name is Scott Baker, my family has been connected with this community for generations and 1941 my grandfather opened his Bakery here. He called it, Jenny Lee, we used to always go, chanting least after church. If you were good in church, a custard pies homemade bread that was still warm years later. I worked in the store I did.

Privacy Risk: grandfather, area, McKees Rocks, Bakery, family, Pittsburgh, Scott Baker, Jenny Lee, store, church, church

Figure 3.5: Speech Detection Page

The *Privacy Detection* feature, based on the Google Cloud Natural Language API, can identify sensitive information within the detected text and return its categories (such as names, locations, phone numbers, etc.). When sensitive information is detected in the text, the plugin highlights this information on the interface and alerts the user to potential privacy risks.

2. **Design Improvements and User Feedback**
   Users found it difficult to locate the exact time when problematic sentences appeared. To address this, we replaced the timeline with clickable cards. When a card is clicked, the video jumps directly to the time point where the sentence appears, making it easier for users to quickly locate and review the problematic sentence.

3.3.6 Explicit Content Detection

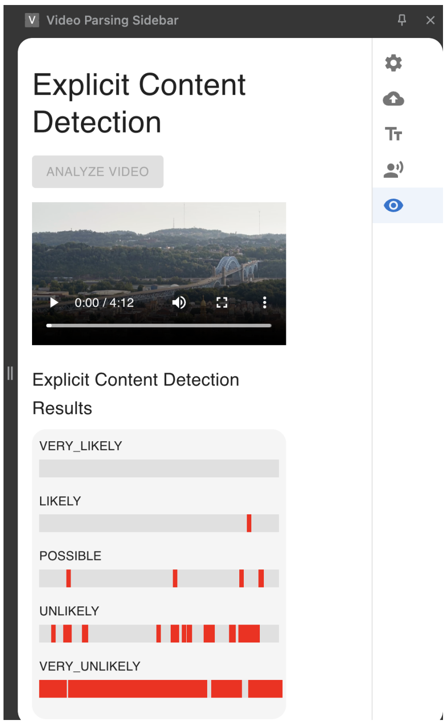


Figure 3.6: Explicit Content Detection Page

The initial design of *Explicit Content Detection* revolved around the following core tasks: detecting explicit content in the video, displaying bounding boxes and labels of detection results, and displaying the time of explicit content appearance on a timeline. These tasks ensure that users can easily identify and handle sensitive content in the video.

1. **Initial Design and Functional Implementation**
   Based on the Google Cloud Video Intelligence API, explicit content detection can identify explicit or inappropriate content in videos and return relevant metadata, including timestamps and content types. Due to its powerful analysis capabilities and seamless integration with other Google services, the Google Cloud Video Intelligence API was our first choice for initial development.

2. **Design Improvements and User Feedback**
   In the initial design, users found it challenging to accurately locate the time point of inappropriate content on the timeline. We improved this by changing the content markers on the timeline to clickable thumbnails. After users click on the thumbnail, the video will jump directly to that time point and play the

corresponding content, allowing users to quickly review it. When the Google Cloud Video Intelligence API returns detection results, it provides a set of scores (usually expressed numerically) to describe the likelihood of explicit content. To make this data easier to understand, we converted these scores into more descriptive labels such as “VERY UNLIKELY”, “UNLIKELY”, “POSSIBLE”, “LIKELY” and “VERY LIKELY” and used these labels to display the detection results. This conversion allows users to quickly understand the sensitivity of the content without interpreting complex numerical values.

With the above improvements, the final design of the plugin is clearer and more intuitive, allowing users to find the necessary functions more quickly and complete configuration operations. By responding promptly to user feedback, we ensured that the page’s features truly meet users’ needs and provide a consistent and pleasant experience during use. At the same time, each feature is reasonably separated from other plugin pages, such as *FaceDetection* and *TextDetection*, avoiding overly coupled functionalities, thus reducing the user’s learning cost. Through this modular design, users can gradually become familiar with the various functions of the plugin and flexibly adjust their usage strategy according to their needs.

Chapter 4

Back-end Design

In the backend design of our plugin, we adopted the Flask framework as the core, combined with Google Cloud Storage for data storage, and utilized technologies such as Google Cloud Video Intelligence API, OpenCV, Google Natural Language Processing API, and YouTube Data API v3. The integration of these APIs enables our system to handle various video content needs, including intelligent analysis, face detection, text detection, and privacy protection. The entire system is designed with a modular approach using a microservices architecture. The combination of these technologies and functionalities allows us to build an efficient, reliable, and easily scalable backend system. The following sections will detail the implementation of each technology and its corresponding functional module.

To understand how we designed the system, it is important to first introduce the working principle of the plugin. The user starts by navigating to the `Settings` page, where they can input the URL to be analysed, the parameters for detection, and the desired confidence threshold. After entering this information, the user uploads the video by clicking the `Upload` button, which then uploads the video to Google Cloud Storage. Based on their specific needs, the user can choose from four detection functionalities: `Face Detection`, `Text Detection`, `Speech Detection`, and `Explicit Detection`. From a backend design perspective, the system adopts a microservices architecture, separating the four functionalities—`Face Detection`, `Text Detection`, `Speech Detection`, and `Explicit Detection`—into distinct services, each dedicated to a specific functionality. The reasons for choosing this architecture will be further elaborated in the following sections. The specific details of the design and implementation principles will be explained in the subsequent sections, Back-end system design architecture, refer to Figure 4.1.

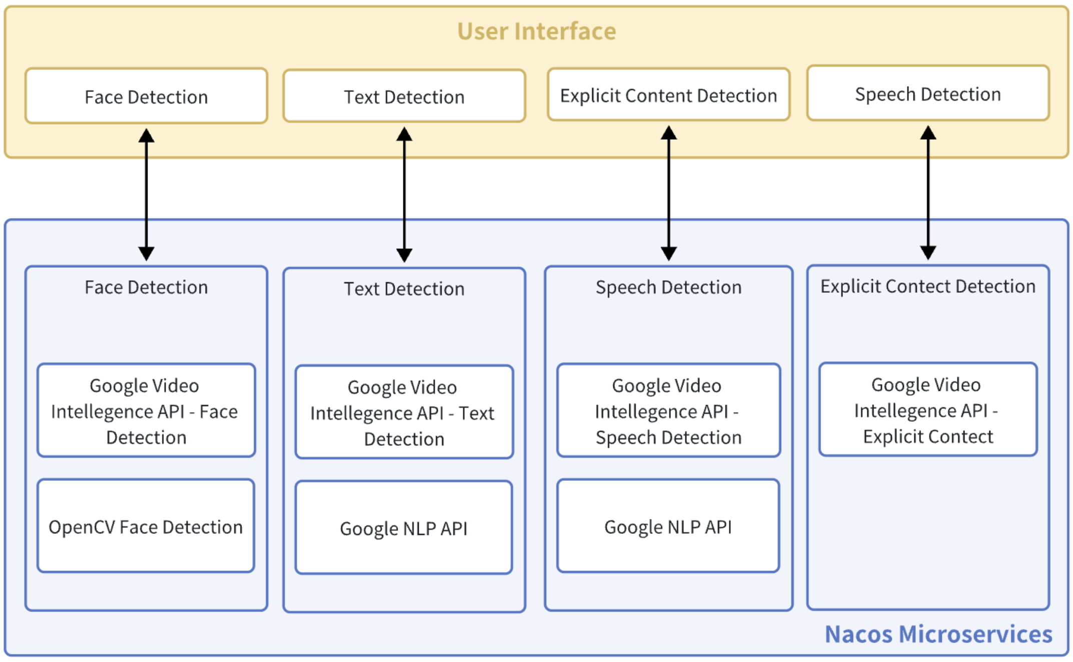

User Interface
Face Detection
Text Detection
Explicit Content Detection
Speech Detection
Face Detection
Google Video Intelligence API - Face Detection
OpenCV Face Detection
Text Detection
Google Video Intellegence API - Text Detection
Google NLP API
Speech Detection
Google Video Intellegence API - Speech Detection
Google NLP API
Explicit Contect Detection
Google Video Intellegence API - Explicit Contect
Nacos Microservices

Figure 4.1: Backend Architecture

4.1 Flask Framework

Our backend service is built on the Flask framework. Flask is known for its lightweight nature and flexibility, without enforcing the use of specific libraries or tools. Developers can freely choose the database, template engine, and other third-party libraries that suit the project's needs. This flexibility allows us to customize based on specific project requirements without being constrained by the framework [28]. Flask also provides simple deployment methods, making it easier to run applications in various environments compared with SpringBoot and Django.

SpringBoot, a powerful framework for enterprise-level Java applications, tends to be more heavyweight and opinionated, requiring extensive configurations that can slow down development for small projects or rapid prototyping [54]. Django, a high-level Python framework with a ”batteries-included” philosophy, offers many built-in features but can be restrictive and introduce unnecessary overhead for smaller projects [29]. In contrast, Flask offers a minimalist approach, making it easier to configure and deploy applications. Whether in a local development environment or on cloud servers in a production environment, Flask can be easily configured and deployed. Therefore, it is particularly suitable for small projects or scenarios requiring rapid prototyping [28]. In this project, we used Flask to implement the video processing API, including video upload, video analysis, and result return functionalities.

- **API Design:** Flask offers simple yet powerful routing capabilities, enabling us to quickly create RESTful APIs. Each functional module (e.g., video upload, video analysis) is managed through independent routes, ensuring clear and maintainable code.

- **CORS Support:** Since the frontend and backend of the plugin operate in different domains, we supported Cross-Origin Resource Sharing (CORS) through Flask's `flask_cors` extension, ensuring that the frontend can safely call the backend API [19]. This feature is especially important for modern web applications as it involves data exchange between different origins.

4.2 API and Technology Integration

In the backend system of our plugin, several key APIs have been integrated to handle complex tasks related to video processing, analysis, and data management. We primarily use Google Cloud Video Intelligence API, OpenCV, Google Natural Processing API, and YouTube Data API v3. The combination of these technologies enables our system to meet various video content needs, including intelligent analysis, face detection, text detection, and privacy protection. Below is a summary of these APIs and their applications in the project.

4.2.1 Google Cloud Video Intelligence API

This API Supports various analyses, such as Object Tracking, Label Detection, Explicit Content Detection, Text Detection, and Face Detection [16]. Users can extract rich information from videos through these functions. In our plugin, this API analyses videos uploaded by users and returns structured data for frontend display. For example, when a user uploads a video, the API is called to analyse it, obtaining detailed information on explicit content, text, and faces present in the video. These analysis results are structured in JSON format, stored in Google Cloud Storage, and used by the frontend system. Compare with other APIs, such as Amazon Rekognition and Microsoft Azure Video Indexer, Google' API pricing model and support for a wide range of video formats made it more suitable for our project requirements [20].

4.2.2 OpenCV

OpenCV, the Open Source Computer Vision Library, is primarily used for local video processing tasks in our project, such as real-time face detection and adding mosaics to faces for privacy protection. OpenCV offers a range of image processing functions, including face detection, object recognition, and image transformation [42]. Its efficiency and flexibility make it a suitable replacement for some Google Cloud Vision API functionalities, especially in scenarios requiring higher real-time performance and data privacy. For instance, OpenCV can be used to add a mosaic to face areas in the video, reducing dependency on external APIs and enhancing data privacy.

4.2.3 Google Natural Language Processing API

Google Natural Language Processing API analyses text content, particularly for privacy detection in our project. It identifies sensitive information, such as personal names, addresses, and phone numbers, helping users protect privacy in video content [14]. The API extracts various information, including entities, sentiments, and categories, which are crucial for identifying potential privacy-invading entities, such as locations or addresses.

4.2.4 YouTube Data API v3

YouTube Data API v3 is crucial for handling tasks related to YouTube videos, enabling developers to access YouTube platform data, including videos, playlists, and channels [27].

4.3 Data Storage: Google Cloud Storage

In our plugin, we chose Google Cloud Storage as the primary storage solution. Google Cloud Storage provides highly reliable object storage services suitable for storing and retrieving large-scale video data [15]. The videos uploaded by users are first stored in Google Cloud Storage. Through the Google Cloud Storage API, we can easily manage these files, including uploading, downloading, and public access.

4.4 Microservice Architecture

In our backend design, the Google Cloud Video Intelligence API is a core component used for analyzing user-uploaded videos. However, the default behavior of the Google Video API is to invoke all functional modules simultaneously, such as object tracking, explicit content detection, text detection, and face detection, generating a JSON file containing all analysis results. While this one-time invocation is simple and easy to use, it has certain drawbacks, especially in agile development environments that require quick responses.

- **Performance and Efficiency Issues:** When we invoke the Google Cloud Video Intelligence API, all detection functions are executed simultaneously, even if the user only needs a subset of these functions. This approach not only wastes computational resources but also causes unnecessary delays, as some unneeded detection functions consume system resources and slow down the overall processing speed. For example, if the user only needs text detection, the system will still perform explicit content detection and face detection, which inevitably increases the video parsing time.
- **Issues in Agile Development:** In agile development [10], requirements often change dynamically. The design of invoking all functions at once is inflexible when adapting to changing requirements. For instance, when we need to add or remove certain detection functions during iterations, this design can lead to significant system refactoring and testing, affecting development efficiency.

Based on the above problems, our solution is:

- **Functional decoupling:** In our project, we separated the different functional modules of video processing (such as video upload, video analysis, result storage, user management, etc.) into independent services. These services communicate through REST APIs or message queues, forming a loosely coupled system architecture. This architecture allows us to independently scale each service's processing capabilities and perform maintenance or upgrades on specific services without affecting the entire system's operation. More importantly, we decided to separate the detection function modules of the Google Cloud Video Intelligence API. This means that we can selectively invoke different detection functions according to user needs, improving the system's response speed and resource utilization. For example, if the user only needs explicit content detection, the system will only invoke the corresponding API module, avoiding unnecessary computations and enhancing system flexibility and efficiency.

- **Microservices Architecture:** Based on this solution, we proposed a microservices architecture design. Microservices architecture is an architectural pattern that decomposes an application into multiple small, independently deployable services [25]. Each service is typically built around a specific business capability and can be independently developed, tested, deployed, and scaled. Compared to traditional monolithic architectures, microservices architecture offers greater flexibility and scalability, making it particularly suitable for complex systems that require frequent updates and expansions. Each detection function module is designed as an independent microservice, allowing for independent scaling or reduction as needed. This design not only enhances system scalability but also reduces errors caused by dynamically adjusting service instances.

- **Nacos Integration:** Nacos is an open-source service discovery and configuration management platform that provides service discovery, configuration management, dynamic DNS services, and health monitoring [40]. Nacos allows services in a distributed system to automatically register and discover each other, enabling load balancing and failover. Using a microservices architecture based on Nacos allows us to dynamically manage service instances when handling high-concurrency requests. For example, when the demand for detection increases, we can automatically scale the instances of a specific detection module to ensure system stability and efficiency. Conversely, when demand decreases, the number of instances can be dynamically reduced to save resources. With Nacos' service discovery and configuration management, we can easily scale and manage microservice instances, ensuring that the system continues to operate efficiently even under varying loads.

4.4.1 Optimization Evaluation

We conducted a comprehensive evaluation of the system architecture optimization through experiments, using 200 processes to simultaneously utilize the plugin and test the four functions separately. The evaluation dimensions included the average detection time before and after microservices architecture optimization (average time for the detection process of the four functions), the success rate in stress testing, and user satisfaction ratings (including the evaluation of detection smoothness and the experience rating of whether there were bugs based on Chapter 5). The detection results are shown in the table below. It can be observed that the microservices architecture significantly improves the detection efficiency of the system.

Detection Method	**Detection Time (seconds)**	**Stress Test Success Rate (%)**	**User NPS**	**Bug NPS**
Pre-optimization Detection	120	85	-20	-60.0
Optimized Detection (Average)	30	95	33.3	10

Table 4.1: Comparison of system efficiency and user experience before and after optimization

4.5 Project Implementation

This section introduces the implementation of the project, focusing on various modules designed to enhance video analysis capabilities. Each module leverages powerful APIs and libraries to provide robust solutions for different video processing needs.

4.5.1 Face Detection Module

The Face Detection module is based on the Google Video Intelligence API, which offers highly reliable and accurate image analysis capabilities for identifying faces in videos and returning detailed information such as bounding boxes and timestamps for each detected face. We use the `videointelligence.FaceDetectionConfig` to configure the face detection functionality, ensuring that bounding boxes and other related attributes are returned for each face. The API results include the position information (bounding boxes) of each detected face, the timestamps when they appear, and the detection confidence. Additionally, this plugin integrates the OpenCV API, an open-source library, to customize and optimize the face recognition algorithm according to specific needs. We have implemented facial blurring features and video rendering to ensure that face-protective mosaics can be embedded into the video and exported, thus overcoming the limitation of the Google Video API, which only provides detection capabilities.

4.5.2 Text Detection Module

The Text Detection module is designed to identify text information appearing in the video. This functionality is particularly useful for recognizing subtitles, identifiers, billboard text, and other content within the video and returning this information with timestamps and locations. The text detection function uses the `Feature.TEXT_DETECTION` API to return all detected text information, including the content of each text segment, the time of appearance, and the specific location in the video (represented as bounding boxes). Additionally, to meet users' needs for detecting potentially privacy-exposing text, this plugin incorporates the Google Natural Language Processing API (Google NLP API) to analyse the text content. When users select the *Privacy Detection* option, it will analyse the detected text content for potential privacy risks, based on the results returned by the Google Video Intelligence API.

4.5.3 Speech Detection Module

The Speech Detection module is used to extract speech content from videos and transcribe the speech into text. This functionality can identify dialogue, narration, or other audio content in the video and return transcriptions with timestamps. We use the `videointelligence.SpeechTranscriptionConfig` to configure speech transcription, including setting the language code (e.g., `en-US`) and enabling automatic punctuation. The returned results include the transcribed text along with the corresponding timestamps, allowing users to view specific dialogue content in the video. Additionally, the plugin integrates the Google Natural Language Processing API (Google NLP API) to analyse the text content, alerting users to any content that may pose a privacy risk.

4.5.4 Explicit Content Detection Module

The Explicit Content Detection module also utilizes the Google Video Intelligence API to detect potentially explicit content in videos, such as adult content or violent scenes. This feature assigns confidence scores based on the detected content and marks segments that may contain explicit material. The explicit content detection function returns the timestamps and confidence scores (e.g., `VERY_UNLIKELY`, `UNLIKELY`, `POSSIBLE`, `LIKELY`, `VERY_LIKELY`) for each marked segment.

4.6 Backend Limitations Analysis

Although our backend system design leverages the powerful capabilities of the Google Cloud Video Intelligence API and optimizes processing workflows through microservices architecture and message queue systems such as Kafka, there are still some limitations in practical applications. These limitations primarily manifest in the following areas:

- **Lack of Real-Time Monitoring:** Due to the design of the Google Cloud Video Intelligence API, video analysis is performed as an asynchronous task, and developers cannot monitor its progress in real-time during the task execution. This means that after a user submits a video analysis request, the system must wait for the entire analysis process to complete before returning results. This approach can lead to long wait times, particularly when processing longer videos.

- **Extended Unresponsiveness:** When a user inputs a video URL and initiates an analysis request, the uncertain duration of the analysis process, especially for longer or more complex videos, may result in a prolonged period of unresponsiveness. This delay not only impacts user experience but may also lead users to mistakenly believe that the system is unresponsive or malfunctioning.

- **Limited API Support from Platforms:** In practical applications, only YouTube provides publicly accessible video data APIs (YouTube Data API v3), allowing us to directly access and analyse public videos. However, many mainstream short video platforms, such as TikTok and Instagram, do not offer similar APIs to access their video content. This significantly limits the applicability of our system on these platforms, as it cannot directly analyse videos from them.

- **API Access Restrictions:** Even on the YouTube platform, API usage is subject to several restrictions, such as request frequency limits and privacy constraints. These restrictions further limit the system's scope of use and flexibility.

Chapter 5

Evaluation

5.1 User Study

User study is a key way to understand how users interact with software. Developers often assume that users will use tools as intended, but this assumption might be influenced by usability biases, so collecting direct feedback through user research is essential.

We conducted two user studies, both involving YouTube content creators. The first study (see Appendix B) took place during the early stages of plugin development, aiming to gather qualitative feedback from users. Participants completed tasks like uploading videos and applying various checks, and their comments helped us identify usability issues and improve the plugin. The second study (combine Appendix A and Appendix B) was conducted after the first round of improvements, adding a section aimed at building user privacy preferences to evaluate whether the plugin meets user needs. We asked testers to select three videos from eight provided on our YouTube account that were most similar to the kinds of videos they usually upload on social media, assessing whether the plugin can detect privacy issues and evaluating the impact of privacy breaches on their upload behavior. The details of the videos are shown in Table 5.2.

These video choices cover a wide range of video types typically uploaded by YouTube creators. This strategic choice ensures that the privacy tool's evaluation is comprehensive and reflects the diverse video production habits of individual creators. Each video type was specifically selected not only to represent different content types, but also because they inherently contain common privacy issues that creators often overlook. Our aim is to validate its effectiveness in identifying and alerting creators to potential privacy breaches that may inadvertently occur due to creator oversight during the typical video production process. This approach facilitates the validation of the tool's usefulness in real-world.

The remaining questions mirrored those in the first study, quantitatively evaluating the plugin's effectiveness and usability, as participants performed simulated tasks in a structured setting, such as detecting video content or identifying sensitive faces in a video.

Table 5.1: YouTube Video Types and Privacy Concerns for Individual Creators

Video Type	**Content Label**	**Privacy Concerns**
Daily Vlog	Personal Life Sharing	Home layout, family members, daily routines
Tutorial	Skill Teaching	Personal workspace details, screen risks
Product Review	Tech and Gadgets	Identifiable information on products, invoices
Travel Vlog	Adventure and Trips	Location details, real-time activity disclosure
Fitness Guide	Workout Routines	Home gym setup, identifiable location features
Cooking Demo	Recipe Walkthrough	Kitchen layout, family-related items
DIY Project	Arts and Crafts	Home details, personal designs in background
Street Interview	Public Interaction	Face of Passerby, sensitive conversations

We collected a total of 20 responses, with 10 participants for each study. The average time spent on the experiment was around 20-30 minutes. This indicates that the length was acceptable and allowed users to focus on completing the experiment. All participants were students from the University of Edinburgh and were either YouTube creators or regular users of YouTube.

The first part of the user study involved having participants fill out a survey discussing the importance of certain privacy preferences and how much they value maintaining their privacy. After completing the survey, they received a link to access the main application and the second part of the usability study.

To complete the final survey, users needed to use a computer with the plugin we provided. Then, they were taken to a page where they could choose their privacy preferences. Once on the settings page, participants selected three videos that closely resembled the ones they uploaded on social media and uploaded them to the plugin. We asked users if there were any privacy breaches in the videos. Additionally, we inquired whether, upon discovering a privacy breach, they would avoid posting the video or continue to publish it. Finally, before answering questions related to usability, users could explore the application and check out its features.

The results of the user study indicate that significant improvements were achieved after implementing modifications based on initial feedback. The marked increase in SUS scores across various areas demonstrates that participants completed tasks more efficiently and accurately. This not only reflects that the updates effectively addressed users' initial concerns but also significantly enhanced the plugin's effectiveness and user-friendliness, proving its usability and popularity. Furthermore, the updates to the plugin not only improved usability but also positively influenced user behavior. In the second study, the process showed that users were more willing to utilize the plugin's features and exhibited higher levels of engagement with it.

5.2 Questionnaire design

5.2.1 Pre-survey

The purpose of the pre-survey was to understand users' privacy preferences. When asked to rate the importance of five privacy aspects on a scale of 1 to 10 (with 10 being extremely private and important), facial and location privacy scored 9.0 and 8.5, respectively, while pets scored the lowest at 5.2. The plugin's features largely align with users' needs based on the privacy model from the pre-survey, but location detection remains a shortcoming. In the second part of the survey, we found the plugin covers most issues users face in actual videos, though items like passports and ID cards are not detected, which will be a focus for future optimization, results refer Figure 5.1.

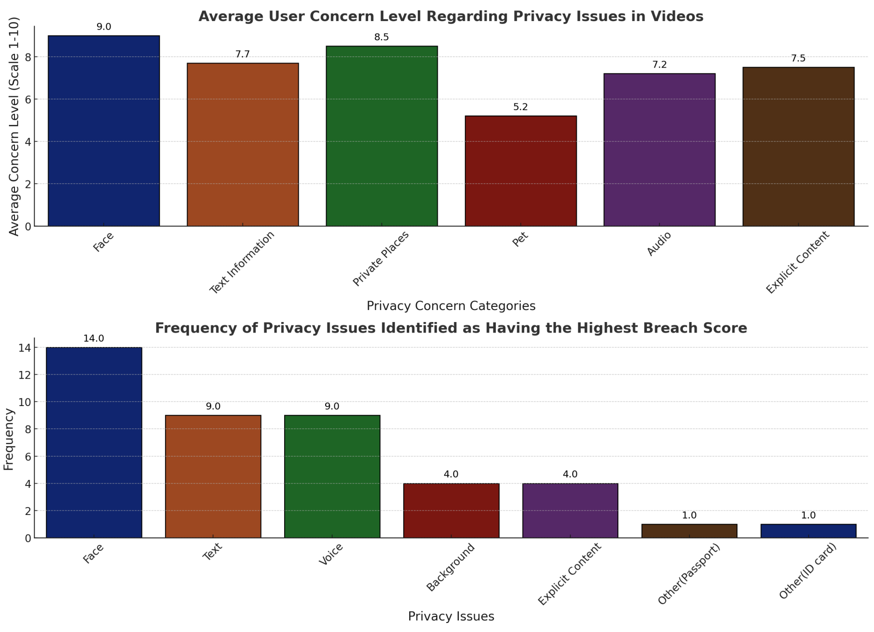


Figure 5.1: Pre-survey Result

In our analysis, we utilized logistic regression to determine the effect of privacy concerns on the decision to upload videos. To simplify the model, responses indicating "Maybe" were recoded as "No" making it a binary classification issue. The logistic regression model is defined as follows:

$$p = \frac{1}{1 + e^{-(\beta_0 + \beta_1 \times \text{Face} + \beta_2 \times \text{Text} + \beta_3 \times \text{Voice})}} \tag{5.1}$$

We analyzed which privacy breach factors might discourage users from uploading videos based on a logistic regression model by Equation 5.1, which helps us in constructing a user privacy model. Figure 5.2 illustrates which features influence user uploading behavior. facial information (β_1) tends to increase the likelihood of uploading videos (It may be due to the personal video blog nature of these eight videos, which leads to some differences with the user privacy model), whereas text (β_2) and voice (β_3) information can discourage uploads. This model helps illustrate how various privacy issues differently impact user decisions to upload content.

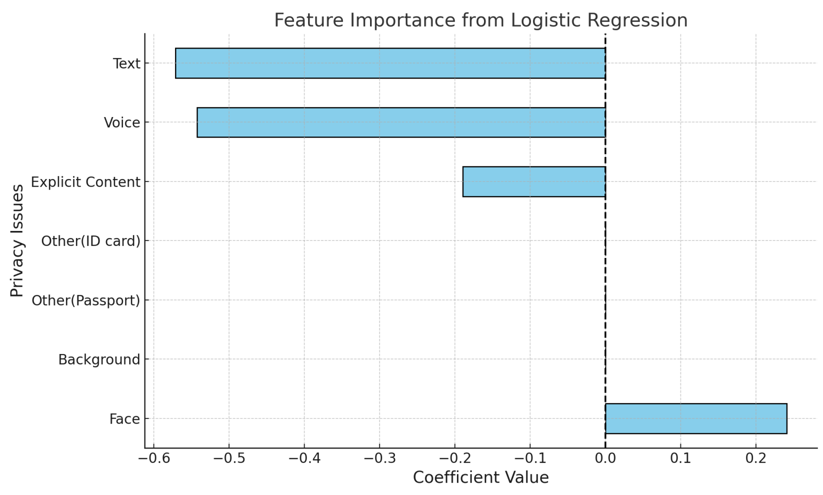


Figure 5.2: Features influencing user uploading behaviour

5.2.2 SUS evaluation

The System Usability Scale (SUS) is an effective product evaluation tool. SUS is a “quick and easy” usability scale created by Brooke and others in 1996 [8], consisting of 10 questions, each with five options ranging from “strongly disagree” to “strongly agree” These questions primarily focus on the usability of the product, and the participants’ responses are converted into a score, which then generates a percentile ranking. Since the plugin needs to be assessed from multiple angles, we only use SUS as a reference when designing the questionnaire.

We created a survey consisting of eleven items. In order to better classify users, Q1 asks users how frequently they utilise the service. Q2–Q4 concentrate on users’ prior worries regarding breaches of video privacy and their encounters with detection technologies. Q5 through Q8, which focus on interface, functionality, and user experience, respectively, are on a Likert scale. A closed-ended scale with forced options from one end to the other is the Likert scale [31]. Each question in Q5–Q7 has six possible answers: strongly disagree, disagree, agree, strongly agree, and prefer not to tell. Q9 calculates the Net Promoter Score (NPS), which ranges from 0 to 10 and indicates the possibility that users will use the plugin in the future [32]. Open-ended questions Q10–Q11 invite users to offer helpful criticism on the program and make recommendations in the following areas: The user demographics, Likert scale, NPS, and open-ended questions sections make up the four sections of the questionnaire.

Below is the analysis of the results of the first four questions, we found that almost everyone is concerned about video privacy leakage, but none of them have tried the related detection tools, indicating that the plugin will have potential in the market, Figure 5.3 is the user data from second study.

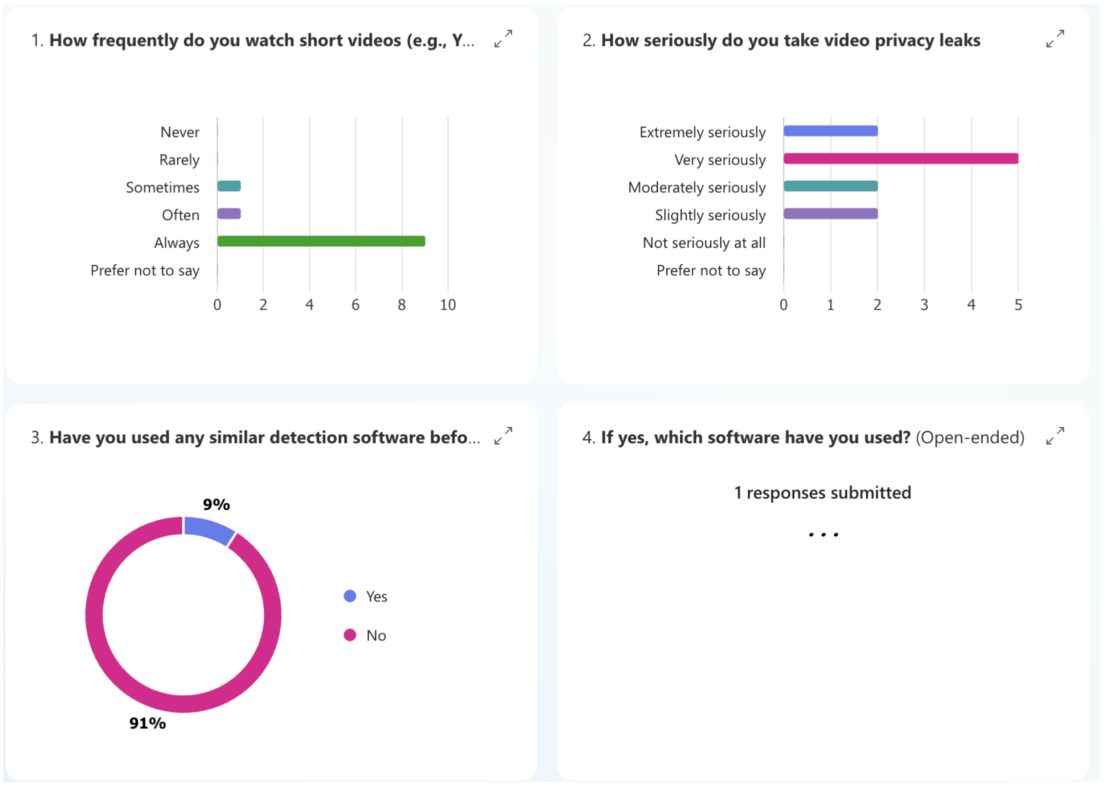


Figure 5.3: User attribute statistics results from second study

The remaining seven questions focused on evaluating the software from a user experience perspective. In the first experiment, the main problems centred on the presentation of the results not being clear and the parsing of the video being too long or even incorrect, we made detailed modifications to address the two problems, the modifications are discussed in detail in Chapter 3, and ultimately, the improved version received satisfactory feedback, and the overall level of satisfaction was improved. Below is the distribution of the results of the responses, we compared the results of the study by converting the options into scores, Strongly Disagree - 1 score, Disagree - 2 scores, Neutral - 3 scores, Agree - 4 scores, and Strongly Agree - 5 scores. As no one responded with the option don't know/don't want to say, we will ignore this option in our analyses. The results show in Table 5.2 and Table 5.3.

First User Study					Second User Study				
Category	Min	Max	Avg	Med	Category	Min	Max	Avg	Med
Interface	56.0	92.0	77.2	78.0	Interface	60.0	100.0	87.2	92.0
Feature	33.3	100.0	64.0	63.3	Feature	60.0	100.0	86.0	86.7
UX	46.7	86.7	68.0	73.3	UX	60.0	100.0	87.3	90.0
Overall	**32.0**	**88.0**	**66.8**	**68.0**	**Overall**	**56.0**	**100.0**	**86.0**	**90.0**

Table 5.2: Comparison of User Ratings between First and Second User Studies

Category	μ_1	s_1	μ_2	s_2	n	**t-score**	**p-value**
Interface	77.2	11.32	87.2	14.70	10	-2.821	0.0115
Feature	64.0	26.70	86.0	13.13	10	-1.704	0.1067
UX	68.0	15.01	87.3	13.13	10	-2.338	0.0359
Overall	**66.8**	**16.23**	**86.0**	**14.14**	**10**	**-3.066**	**0.0068**

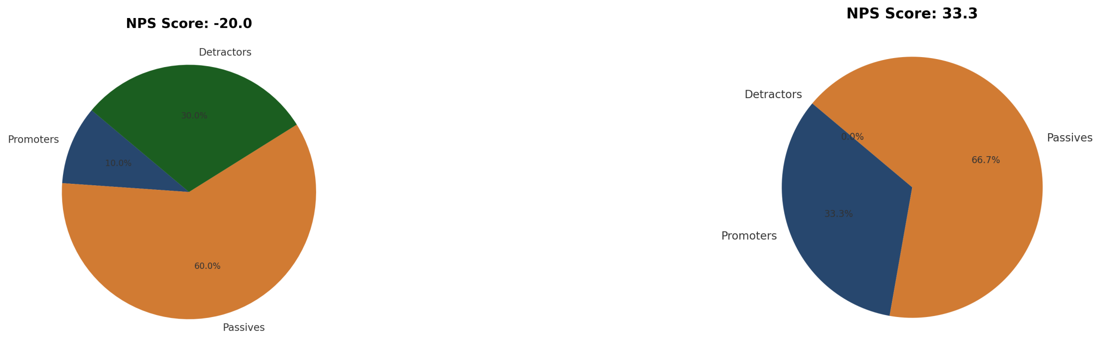
Table 5.3: Welch's test result comparing two user studies

We present the statistics of two user studies, focusing on four key areas: Interface, Feature, UX, and Overall scores. For each area, including the overall score, we have calculated the minimum, maximum, average, and median scores. Given that each area has a different total possible score, the results have been converted to percentages for consistency. For example, the average overall score in the first study is 66.8 out of a possible 100, which translates to 66.8%. The second study, evaluating the final version, received notably higher scores. Minimum, maximum, mean and median values are significantly higher, highlighting significant progress. To confirm the significance of these improvements, we applied Welch's t-test, which compares two groups' means without assuming equal variances. Tested hypotheses included:

- **H0 (null hypothesis):** The means are equal (no significant difference).
- **H1 (alternative hypothesis):** The means are not equal (a significant difference exists).

The analysis of data from the two user studies shows a clear improvement in user satisfaction and plugin performance. In the second study, all key areas—Interface, Feature, UX, and Overall—had higher minimum, maximum, average, and median scores compared to the first. For instance, the average Interface score increased from 77.2% to 87.2%, while Overall rose from 66.8% to 86.0%. Notably, in the *Feature* category, the minimum score increased from 33.3% to 60.0%, indicating a more consistent and satisfying user experience. Welch's t-test p-values showed statistically significant improvements in Interface, UX, and Overall scores at the 0.01 level. Although *Feature* scores didn't reach this significance, the overall trend suggests meaningful improvements for most users.

Q9 asks users about their likelihood of continuing to use the plugin, with scores ranging from 0 to 10. This represents the Net Promoter Score (NPS), a metric for measuring user satisfaction and enthusiasm [46]. Scores between 0 and 6 are classified as detractors, 7 or 8 as passive users, and 9 or 10 as promoters. In our analysis, these categories are treated uniformly; for example, scores of 0 and 6 are both considered detractors. While NPS results can be overstated in business contexts [32], it is appropriate to use here to gauge overall satisfaction.



(a) NPS score of the first study (b) NPS score of the second study

Figure 5.4: NPS score comparison

The NPS score is calculated as %promoters - %detractors, ranging from -100 to 100, with higher scores being better. More promoters than detractors are indicated by an NPS score higher than 0. The two studies' NPS scores are displayed in Figure 5.4. There were three critics and one promoter in the first research, yielding an NPS of -20. In the second survey, there was a significant improvement in user satisfaction from the prototype to the final version, as seen by the NPS increasing to 33.3 and all six non-passive replies being promoters. All users expressed pleasure with the final version, and even after examining it exclusively, the NPS was still favourable, indicating that they might keep using it for video privacy checks.

Chapter 6

Conclusions

6.1 Summary

In this project, we designed a Chrome browser-based plugin for detecting privacy leaks in YouTube videos to aid video social media creators in identifying potential privacy issues in the content they publish. We adopted agile development methodologies during the plugin development and created a feedback loop with user research to improve the design. We can summarize our work by addressing the research objectives.

RQ1: We successfully implemented a tool capable of detecting various types of privacy leaks, including unintended facial appearances, exposed sensitive text, audio privacy breaches, and explicit content detection.

RQ2: We adopted a front-end and back-end separated architecture, utilizing React and Chrome's Manifest v3 for creating web layouts, and optimized the interface with Material-UI. The backend design is based on the Flask framework as the core, integrated with Google Cloud Storage for data storage, and supported by APIs such as Google Cloud Video Intelligence API, OpenCV, Google NLP API, and YouTube Data API v3.

RQ3: To assess the usability of our tool, we conducted two user studies—one with the prototype and another with the final version. By thoroughly analyzing the results using various methods, we observed significant improvements and received valuable feedback, confirming that the tool is indeed user-friendly and well-suited for practical use.

RQ4: To evaluate plug-in designs, including user experience, functional effectiveness, and detection accuracy, we conducted two user studies (see chapter 5 for details). The first focused on the prototype, and the second on the final version. We used methods such as usability testing, SUS scores, and task performance metrics to assess user interaction, along with accuracy tests for detection effectiveness. By comparing both studies, we identified significant improvements and gathered feedback, ensuring the final design met high standards in usability, effectiveness, and precision.

6.2 Future Work

We received suggestions for new features, such as detecting items in videos, analyzing backgrounds for leakage risks, and applying mosaic protection. While feasible, these have not been implemented yet. Options like embedding detection results for easy download are also possible. However, the current version heavily relies on costly APIs, so future versions may shift to cloud-based machine learning for detection to reduce API dependency.

Additionally, the plugin needs to be uploaded to the Google Chrome Extensions platform, which requires increased attention to user security, such as using MySQL instead of Google Cloud Storage and implementing gateway filtering between the front and back end. The current plug-in hasn't addressed high concurrency scenarios, so the back-end architecture needs further optimization. I'm also experimenting with distributing API calls across multiple accounts to evenly spread the load and reduce costs.

Finally, extensive testing for different video types and scenarios to ensure the stability and accuracy of the new features. Currently only supports Chrome plug-ins, the future can be applied to different browsers to meet user requirements.

6.3 Reflections

Through this project, my first full-stack experience, I gained valuable knowledge in front-end web design and rapid development with frameworks, ultimately creating an interface I'm satisfied with. However, there are areas for improvement, such as refining the logic and enhancing the user experience to meet enterprise-level standards. I'm also concerned about potential excessive API consumption and the associated costs once the plugin is released, which will require further research for optimization.

Despite these challenges, the project was a valuable learning experience, allowing me to go through the entire software development lifecycle—from development to testing and usability analysis. This has significantly improved my programming and product design skills, which will be essential for my future work.

Bibliography

[1] Alessandro Acquisti, Laura Brandimarte, and George Loewenstein. Privacy and human behavior in the age of information. *Science*, 347(6221):509–514, 2015.

[2] Sanchit Aggarwal et al. Modern web-development using reactjs. *International Journal of Recent Research Aspects*, 5(1):133–137, 2018.

[3] Ross Anderson. *Security engineering: a guide to building dependable distributed systems*. John Wiley & Sons, 2020.

[4] Jane Arthurs, Sophia Drakopoulou, and Alessandro Gandini. Researching youtube, 2018.

[5] Ghazaleh Beigi and Huan Liu. A survey on privacy in social media: Identification, mitigation, and applications. *ACM Transactions on Data Science*, 1(1):1–38, 2020.

[6] Adam Boduch. *React material-ui cookbook: build captivating user experiences using react and material-ui*. Packt Publishing Ltd, 2019.

[7] Michael Boyle, Christopher Edwards, and Saul Greenberg. The effects of filtered video on awareness and privacy. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, pages 1–10, 2000.

[8] John Brooke et al. Sus-a quick and dirty usability scale. *Usability evaluation in industry*, 189(194):4–7, 1996.

[9] Jean Burgess and Joshua Green. *YouTube: Online video and participatory culture*. John Wiley & Sons, 2018.

[10] Lan Cao, Kannan Mohan, Peng Xu, and Balasubramaniam Ramesh. A framework for adapting agile development methodologies. *European Journal of Information Systems*, 18(4):332–343, 2009.

[11] Nicholas Carlini, Adrienne Porter Felt, and David Wagner. An evaluation of the google chrome extension security architecture. In *21st USENIX Security Symposium (USENIX Security 12)*, pages 97–111, 2012.

[12] Google Help Center. Google photos help - edit your photos, 2024. Accessed: 2024-08-18.

[13] Jay Pil Choi, Doh-Shin Jeon, and Byung-Cheol Kim. Privacy and personal data collection with information externalities. *Journal of Public Economics*, 173:113–124, 2019.

[14] Google Cloud. Google cloud natural language api documentation, 2024. Accessed: 2024-08-18.

[15] Google Cloud. Google cloud storage documentation, 2024. Accessed: 2024-08-18.

[16] Google Cloud. Google cloud video intelligence api documentation, 2024. Accessed: 2024-08-18.

[17] Lorrie F Cranor. A framework for reasoning about the human in the loop. 2008.

[18] Jasmine DeHart, Makya Stell, and Christan Grant. Social media and the scourge of visual privacy. *Information*, 11(2):57, 2020.

[19] Cory Dolphin and Flask-CORS contributors. Flask-cors documentation, 2024. Accessed: 2024-08-18.

[20] Mamarajabov Odil Elmurzayevich and Butayev Otabek. Amazon, google va microsoft platformalarini solishtirish. *Journal of Academic Research and Trends in Educational Sciences*, pages 301–306, 2024.

[21] GERALD FRIEDLAND and JAEYOUNG CHOI. Semantic computing and privacy: A case study using inferred geo-location. *International Journal of Semantic Computing*, 2011.

[22] Gerald Friedland, Robin Sommer, et al. Cybercasing the joint: On the privacy implications of {Geo-Tagging}. In *5th USENIX workshop on Hot Topics in Security (HotSec 10)*, 2010.

[23] Cory Gackenheimer. *Introduction to React*. Apress, 2015.

[24] Tom Gerety. Redefining privacy. *Harv. CR-CLL Rev.*, 12:233, 1977.

[25] Javad Ghofrani and Daniel Lübke. Challenges of microservices architecture: A survey on the state of the practice. *ZEUS*, 2018:1–8, 2018.

[26] Google. What is manifest v3, 2024. Accessed: 2024-08-18.

[27] Google. Youtube data api v3 documentation, 2024. Accessed: 2024-08-18.

[28] Miguel Grinberg. *Flask web development.* " O'Reilly Media, Inc.", 2018.

[29] Adrian Holovaty and Jacob Kaplan-Moss. *The definitive guide to Django: Web development done right.* Apress, 2009.

[30] Ankit Kumar Jain, Somya Ranjan Sahoo, and Jyoti Kaubiyal. Online social networks security and privacy: comprehensive review and analysis. *Complex & Intelligent Systems*, 7(5):2157–2177, 2021.

[31] Ankur Joshi, Saket Kale, Satish Chandel, and D Kumar Pal. Likert scale: Explored and explained. *British journal of applied science & technology*, 7(4):396–403, 2015.

[32] Timothy L Keiningham, Bruce Cooil, Tor Wallin Andreassen, and Lerzan Aksoy. A longitudinal examination of net promoter and firm revenue growth. *Journal of Marketing*, 71(3):39–51, 2007.

[33] Maxime Lamothe, Yann-Gaël Guéhéneuc, and Weiyi Shang. A systematic review of api evolution literature. *ACM Computing Surveys (CSUR)*, 54(8):1–36, 2021.

[34] Patricia G Lange. Publicly private and privately public: Social networking on youtube. *Journal of computer-mediated communication*, 13(1):361–380, 2007.

[35] Lei Liu, Xinwen Zhang, Guanhua Yan, Songqing Chen, et al. Chrome extensions: Threat analysis and countermeasures. In *NDSS*. Citeseer, 2012.

[36] Alice Marwick and Danah Boyd. To see and be seen: Celebrity practice on twitter. *Convergence*, 17(2):139–158, 2011.

[37] Alice E Marwick and Danah Boyd. I tweet honestly, i tweet passionately: Twitter users, context collapse, and the imagined audience. *New media & society*, 13(1):114–133, 2011.

[38] Alice E Marwick and Danah Boyd. Networked privacy: How teenagers negotiate context in social media. *New media & society*, 16(7):1051–1067, 2014.

[39] Microsoft. Learn about azure ai video indexer, 2024. Accessed: 2024-08-18.

[40] Nacos. What is nacos?, 2024. Accessed: 2024-08-18.

[41] Jakob Nielsen. How to conduct a heuristic evaluation. *Nielsen Norman Group*, 1(1):8, 1995.

[42] OpenCV. Opencv documentation, 2024. Accessed: 2024-08-18.

[43] Leysia Palen and Paul Dourish. Unpacking" privacy" for a networked world. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 129–136, 2003.

[44] Nikolaos Pantelaios and Alexandros Kapravelos. Manifest v3 unveiled: Navigating the new era of browser extensions. *arXiv preprint arXiv:2404.08310*, 2024.

[45] Arvind Ravulavaru. *Google Cloud AI Services Quick Start Guide: Build Intelligent Applications with Google Cloud AI Services*. Packt Publishing Ltd, 2018.

[46] Frederick F Reichheld. The one number you need to grow. *Harvard business review*, 81(12):46–55, 2003.

[47] Martin P Robillard and Robert DeLine. A field study of api learning obstacles. *Empirical Software Engineering*, 16:703–732, 2011.

[48] Megan Sauer. Some tiktok users are receiving $167 checks over data privacy violations—and google and snapchat could be next, 2022. Accessed: 2022-10-28.

[49] Aliaksandra Shutsko. User-generated short video content in social media. a case study of tiktok. In *Social Computing and Social Media. Participation, User Experience, Consumer Experience, and Applications of Social Computing: 12th International Conference, SCSM 2020, Held as Part of the 22nd HCI International Conference, HCII 2020, Copenhagen, Denmark, July 19–24, 2020, Proceedings, Part II 22*, pages 108–125. Springer, 2020.

[50] Agrima Srivastava and G Geethakumari. Measuring privacy leaks in online social networks. In *2013 International Conference on Advances in Computing, Communications and Informatics (ICACCI)*, pages 2095–2100. IEEE, 2013.

[51] Frederic D Stutzman, Ralph Gross, and Alessandro Acquisti. Silent listeners: The evolution of privacy and disclosure on facebook. *Journal of privacy and confidentiality*, 4(2):2, 2013.

[52] Mike Thelwall, Pardeep Sud, and Farida Vis. Commenting on youtube videos: From guatemalan rock to el big bang. *Journal of the American society for information science and technology*, 63(3):616–629, 2012.

[53] Jai Prakash Verma, Smita Agrawal, Bankim Patel, and Atul Patel. Big data analytics: Challenges and applications for text, audio, video, and social media data. *International Journal on Soft Computing, Artificial Intelligence and Applications (IJSCAI)*, 5(1):41–51, 2016.

[54] Craig Walls. *Spring in action.* Simon and Schuster, 2022.

[55] Zhuo Wei, Yongdong Wu, Yanjiang Yang, Zheng Yan, Qingqi Pei, Yajuan Xie, and Jian Weng. Autoprivacy: Automatic privacy protection and tagging suggestion for mobile social photo. *Computers & Security*, 76:341–353, 2018.

[56] Alan F Westin. Privacy and freedom. *Washington and Lee Law Review*, 25(1):166, 1968.

[57] Jehan Wickramasuriya, Mahesh Datt, Sharad Mehrotra, and Nalini Venkatasubramanian. Privacy protecting data collection in media spaces. In *Proceedings of the 12th annual ACM international conference on Multimedia*, pages 48–55, 2004.

[58] Jon Yablonski. *Laws of UX.* " O'Reilly Media, Inc.", 2024.

Appendix A

Pre-Survey form

A survey for a Privacy Tool for Detecting and Preventing Youtube Data Leaks

* Required

Privacy leak Detector

The following sections are to be filled while/after using the application.
The survey will take approximately 10 minutes to complete.

Please note: After completing the Pre-survey section, you must use the plugin before proceeding to the remaining sections.

Pre-Survey

Please rate the below Privacy Preferences on a scale of 1-10 (10 being most important)

1. Face - If there are few people in the video and some faces are very clear, it may lead to the leakage of people's privacy.

1	2	3	4	5	6	7	8	9	10

2. Text information - For example, if there is information with text in the background, such as a restaurant sign, the privacy of a person's location will be compromised.

1	2	3	4	5	6	7	8	9	10

3. Private Places - Breach occurs if the image is captured in a private setting like places of work or home.

1	2	3	4	5	6	7	8	9	10

4. Pet - This breach occurs if the video captures any of the animals (mainly dogs and cats)

1	2	3	4	5	6	7	8	9	10

5. Audio - such as audio in a video, could reveal location or identity information

1	2	3	4	5	6	7	8	9	10

6. Explicit Content - There are some explicit information in the video, which may be your privacy

1	2	3	4	5	6	7	8	9	10

Video #1

Answer the leak related questions, only if the image leaked your privacy settings.

7. Please choose the Video ID

- ○ Video 1
- ○ Video 2
- ○ Video 3
- ○ Video 4
- ○ Video 5
- ○ Video 6
- ○ Video 7
- ○ Video 8

8. Did the video violate your privacy? *

- ○ Yes
- ○ No
- ○ Maybe

9. What was your overall breach score? *

1	2	3	4	5	6	7	8	9	10

10. Which privacy preference had the highest breach score? (eg face or logo *

- ☐ Face
- ☐ Voice
- ☐ Text
- ☐ Explicit Content
- ☐ Background
- ☐ Other

11. Would you still post this video? *

- ◯ Yes
- ◯ No
- ◯ Maybe

Video #2

Answer the leak related questions, only if the image leaked your privacy settings.

12. Please choose the Video ID

- ◯ Video 1
- ◯ Video 2
- ◯ Video 3
- ◯ Video 4
- ◯ Video 5
- ◯ Video 6
- ◯ Video 7
- ◯ Video 8

13. Did the video violate your privacy?

- ◯ Yes
- ◯ No
- ◯ Maybe

14. What was your overall breach score? *

1	2	3	4	5	6	7	8	9	10

15. Which privacy preference had the highest breach score? (eg face or logo *

- ☐ Face
- ☐ Voice
- ☐ Text
- ☐ Explicit Content
- ☐ Background
- ☐ Other

16. Would you still post this video?

- ○ Yes
- ○ No
- ○ Maybe

Video #3

Answer the leak related questions, only if the image leaked your privacy settings.

17. Please choose the Video ID

- ○ Video 1
- ○ Video 2
- ○ Video 3
- ○ Video 4
- ○ Video 5
- ○ Video 6
- ○ Video 7
- ○ Video 8

18. Did the video violate your privacy

- ○ Yes
- ○ No
- ○ Maybe

19. What was your overall breach score? *

1	2	3	4	5	6	7	8	9	10

20. Which privacy preference had the highest breach score? (eg face or logo *

- ☐ Face
- ☐ Voice
- ☐ Text
- ☐ Explicit Content
- ☐ Background
- ☐ Other

21. Would you still post this video?

- ○ Yes
- ○ No
- ○ Maybe

Appendix B

Survey

User Experience Questionaire

22. **How frequently do you watch short videos (e.g., Youtube Short, TikTok, Instagram Reels)?**

- ○ Never
- ○ Rarely
- ○ Sometimes
- ○ Often
- ○ Always
- ○ Prefer not to say

23. **How seriously do you take video privacy leaks**

- ○ Extremely seriously
- ○ Very seriously
- ○ Moderately seriously
- ○ Slightly seriously
- ○ Not seriously at all
- ○ Prefer not to say

24. **Have you used any similar detection software before?**

- ○ Yes
- ○ No

25. **If yes, which software have you used?** (Open-ended)

26. **Please rate the overall of the plugin by the following indicators.**

	Strongly Agree	Agree	Neutral	Disagree	Strongly Di
It is easy to use this plugin for the first time.	◯	◯	◯	◯	◯
The interface information on this plugin (such as content materials, icon recognition, is easy to understand.	◯	◯	◯	◯	◯
I am interested in the solutions provided on the plugin.	◯	◯	◯	◯	◯
I can quickly find the function which I need on plugin.	◯	◯	◯	◯	◯
I find the plugin to be very smooth to use.	◯	◯	◯	◯	◯

27. **Please rate the interface of the plugin by the following indicators.**

	Strongly Agree	Agree	Neutral	Disagree	Strongly Di
The use of words in the interface is easy to understand	◯	◯	◯	◯	◯
The use of icons in the interface is easy to understand.	◯	◯	◯	◯	◯
I know how to use plugin without reading documentati on.	◯	◯	◯	◯	◯
The interface design is intuitive.	◯	◯	◯	◯	◯
The font colour is accessible to me.	◯	◯	◯	◯	◯

28. **Please rate the features of plugin by the following indicators.**

	Strongly Agree	Agree	Neutral	Disagree	Strongly Di
The settings page is easy to understand.	○	○	○	○	○
The settings options satisfy my needs.	○	○	○	○	○
The information on suggestion cards is clear and helpful.	○	○	○	○	○

29. **Please rate your user experience by the following indicators.**

	Strongly Agree	Agree	Neutral	Disagree	Strongly Di
I didn't find any bugs when using this plugin.	○	○	○	○	○
The plugin works well in browser.	○	○	○	○	○
The plugin accurately identifies privacy leaks.	○	○	○	○	○

30. **How likely will you keep using the plugin in the future?**

1	2	3	4	5	6	7	8	9	10

31. **If you could improve one thing about the plugin, what would it be?**

32. **Is there any feature you think the plugin should add?**

Appendix C

Participants’ information sheet

Participant Information Sheet

Project title:	A Privacy Tool for Detecting and Preventing Youtube Data Leaks	
Principal investigator:	Nadin Kokciyan	
Researcher collecting data:	Jiaxi Chen	
Funder (if applicable):		

This study was certified according to the Informatics Research Ethics Process, reference number 963374. Please take time to read the following information carefully. You should keep this page for your records.

Who are the researchers?

The research is being carried out as part of the MSc final project of Jiaxi Chen at The University of Edinburgh. This project is supervised by Nadin Kokciyan.

What is the purpose of the study?

As a Youtube privacy control tool, the tool will provide users with an instant detection tool that automatically detects whether the content uploaded by the user is at risk of a privacy breach and provides a reminder. the purpose of the experiment is to evaluate the quality of the privacy information provided by the plugin, as well as to evaluate the plugin itself.

Why have I been asked to take part?

You regularly browse the internet and actively use social media.

Do I have to take part?

No – participation in this study is entirely up to you. You can withdraw from the study at any time, without giving a reason. After this point, personal data will be deleted and anonymised data will be combined such that it is impossible to remove individual information from the analysis. Your rights will not be affected. If you wish to withdraw, contact the PI. We will keep copies of your original consent, and of your withdrawal request.

What will happen if I decide to take part?

Firstly, you will be given a laptop with the browser add-on installed. You will then be instructed to log into your social media accounts and browse the website as usual, noting the information displayed in the browser extension. Subsequently, you will be instructed to use the text box of the browser extension to create your own text posts based on the character and information of the fictitious profile you have given or created yourself. Afterwards, you will be asked to fill out an online questionnaire to help evaluate the usability of the plugin and the quality of the information provided about the privacy settings you have chosen. The duration of this session should not exceed 30 minutes. No audio or video will be recorded of any participants. This study will be conducted once at a time that suits you best, in a student space like Appleton Tower or Main Library.

Are there any risks associated with taking part?

There are no significant risks associated with participation.

Are there any benefits associated with taking part?

No.

What will happen to the results of this study?
The results of this study may be summarised in published articles, reports and presentations. Quotes or key findings will be anonymized: We will remove any information that could, in our assessment, allow anyone to identify you. With your consent, information can also be used for future research. Your data may be archived for a maximum of 2 years. All potentially identifiable data will be deleted within this timeframe if it has not already been deleted as part of anonymization.

Data protection and confidentiality.
Your data will be processed in accordance with Data Protection Law. All information collected about you will be kept strictly confidential. Your data will be referred to by a

unique participant number rather than by name. Your data will only be viewed by the researcher/research team Jiaxi Chen and Nadin Kokciyan.

All electronic data will be stored on a password-protected encrypted computer, on the School of Informatics' secure file servers, or on the University's secure encrypted cloud storage services (DataShare, ownCloud, or Sharepoint) and all paper records will be stored in a locked filing cabinet in the PI's office. Your consent information will be kept separately from your responses in order to minimise risk.

What are my data protection rights?
The University of Edinburgh is a Data Controller for the information you provide. You have the right to access information held about you. Your right of access can be exercised in accordance Data Protection Law. You also have other rights including rights of correction, erasure and objection. For more details, including the right to lodge a complaint with the Information Commissioner's Office, please visit www.ico.org.uk. Questions, comments and requests about your personal data can also be sent to the University Data Protection Officer at dpo@ed.ac.uk.

Who can I contact?
If you have any further questions about the study, please contact the lead researcher, Jiaxi Chen <s2530822@ed.ac.uk>.
If you wish to make a complaint about the study, please contact inf-ethics@inf.ed.ac.uk. When you contact us, please provide the study title and detail the nature of your complaint.
Updated information.
If the research project changes in any way, an updated Participant Information Sheet will be made available on http://web.inf.ed.ac.uk/infweb/research/study-updates.

Alternative formats.
To request this document in an alternative format, such as large print or on coloured paper, please contact Jiaxi Chen <s2530822@ed.ac.uk>.
General information.
For general information about how we use your data, go to: edin.ac/privacy-research

THE UNIVERSITY of EDINBURGH
informatics

Appendix D

Participants' consent form

Participant number:________________________

Participant Consent Form

Project title:	A Privacy Tool for Detecting and Preventing Youtube Data Leaks
Principal investigator (PI):	Nadin Kokciyan
Researcher:	Jiaxi Chen
PI contact details:	nadin.kokciyan@ed.ac.uk

By participating in the study you agree that:

- I have read and understood the Participant Information Sheet for the above study, that I have had the opportunity to ask questions, and that any questions I had were answered to my satisfaction.
- My participation is voluntary, and that I can withdraw at any time without giving a reason. Withdrawing will not affect any of my rights.
- I consent to my anonymised data being used in academic publications and presentations.
- I understand that my anonymised data will be stored for the duration outlined in the Participant Information Sheet.

Please tick yes or no for each of these statements.

1.	I agree to being audio recorded.	**Yes** ☐	**No** ☐
2.	I agree to being video recorded.	**Yes** ☐	**No** ☐
3.	I allow my data to be used in future ethically approved research.	**Yes** ☐	**No** ☐
4.	I agree to take part in this study.	**Yes** ☐	**No** ☐

Name of person giving consent	Date dd/mm/yy	Signature
Name of person taking consent	Date dd/mm/yy	Signature

THE UNIVERSITY *of* EDINBURGH informatics