Sri Lanka Institute of Information Technology

IE3092 - Information Security Project

Project Proposal

Browser Extension for Preventing Sensitive Data Leaks.

Batch ID: Y3.S2.WE.CS

Group No: ISP-33

August 2025

PROJECT PROPOSAL REPORT

- Official name of the degree - Bachelor of Science in Information Technology specializing in cyber security

- Official name of the department of the university - Department of Computer Systems Engineering

- The month and year of submission- August 2025

Student Name	Student ID	Email
DILSHARA N P G R	IT22064790	it22064790@my.sliit.lk
DULANJANA E J A T	IT22186874	it22186874@my.sliit.lk

DECLARATION

We declare that this is our own work, and this proposal does not incorporate without acknowledgement any material previously submitted for a degree or diploma in any other university or Institute of higher learning and to the best of our knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Student ID	Student Name	Signature
IT22064790	DILSHARA N P G R	[illegible]
IT22186874	DULANJANA E J A T	[illegible]

The supervisor/s should certify the proposal report with the following declaration.

The above candidates are carrying out research for the undergraduate Dissertation under my supervision.

………………………….. ………………….

Signature of the supervisor Date

ABSTRACT.

The risk of sensitive data breaches has increased rapidly with the use of online forms and AI tools. Insecure text boxes are frequently used by users to type or paste bank account numbers, trade secrets, or personal information. Financial loss and severe privacy issues can be induced by the leakage of such data. Because the intrusion occurs at the browser before data has a chance to go out traditional security measures like firewalls and anti-virus are unable to stop it.

This project will create a browser extension that protects users from such mistakes. The plugin will check user input in real time against AI systems or forms. It will use patterns matching with regular expressions, keyword searches and lastly machine learning to identify sensitive information like login credentials, credit card numbers and personal identification numbers. The system will offer a simple pop-up notice when a risk is identified. The user will be able to halt, edit, or proceed with the submission.

All scanning will be done in the browser, so no personally identifiable data will exit the user's PC. This 'privacy by design' methodology helps with adherence to regulations such as GDPR and HIPAA. The extension will be developed in JavaScript, HTML and CSS, and first released on the Chrome Web Store. It will then leverage user feedback to develop the product.

This project offers a simple yet effective solution to prevent unintentional data leakage by bridging the gap between user behavior and server security. It has the potential to protect individuals and businesses alike, particularly with the growing adoption of AI services. Thus, our project aims to facilitate greater trust in digital services and human AI collaboration and reduce risk.

Keywords: Data Leak Prevention, Browser Security, Privacy Extension, PII Protection, Cybersecurity

Table of Contents

LIST OF ABBREVIATIONS

PII	Personally Identifiable Information
LLM	Large Language Model
GDPR	General Data Protection Regulation
HIPAA	Health Insurance Portability and Accountability Act
API	Application Programming Interface
UI	User Interface
ML	Machine Learning
NLP	Natural Language Processing
Regex	Regular Expression

INTRODUCTION.

Large Language Models (LLMs) such as GPT, Gemini, and LLaMA have changed online space through the ability to provide intelligent automation in customer service, healthcare, education, and finance. There are, however, more security and privacy risks due to this rapid growth, especially the unintentional leakage of personal data. Since they rely on user input and training over large corporal LLMs are vulnerable to new attack vectors such as data leakage, adversarial manipulation and prompt injections. Attackers can design inputs to circumvent system controls, pilfer personal data, or create malicious outputs. These vulnerabilities, paired with LLMs' proclivity towards memory, increase the likelihood for confidential or sensitive information to leak out.

Apart from AI-related issues, most of the leaks are the fault of the users themselves. Individuals tend to type in login credentials, company information, or personally identifiable information (PII) in AI chat windows and web forms. Older enterprise technology like network monitoring or endpoint protection cannot identify this because it happens right in the browser. The severity of the problem is evidenced by the March 2023 ChatGPT data breach in which customers' behavior and system vulnerabilities exposed customers' names, emails, and payments.

By insisting on more robust data protection, requirements like the General Data Protection Regulation (GDPR) point to such threats. However, existing privacy techniques like federated learning and anonymization only address the models that train AI and not the protection of users in real-time. The browser level, where data initially is entered, is where a solution must be achieved.

The purpose of this project is to create a browser extension that stops leaks in advance. Using regular expressions, keyword spotting, and ultimately machine learning, it will be constantly scanning text. Before submission, the user will be warned if sensitive information is found. The solution will be context-aware, unlike conventional filters, to minimize false positives and assist in protecting AI tools from prompt injection attacks.

The initiative seeks to solve this issue in a way that will deter both accidental and intentional leaks. It offers an expandable, deployable, and easy-to-use solution that favors individuals and businesses in safeguarding their data and establishing trust in online business.

How to Use It.

Google Chrome and other Chromium browsers carry the extension. After installation, it operates on its own. The plugin checks the text while the user is typing in a chat session or form. The submission is made normally if no sensitive data is detected. The user is presented with three options in a popup if sensitive data is detected: cancel, edit, or proceed.

Project Importance and Benefits.

- prevents unintended disclosure of business and personal information.
- safeguards users from AI hazards such as rapid injections.
- brings awareness through alerting people before they disclose personal data.
- supports compliance with data protection regulations such as HIPAA and GDPR.
- lightweight, simple to use, and doesn't slow down the browser.
- As it can be published on the Chrome Web Store, it is of commercial value.

PROBLEMS

Why Implement This Project?

There are more individuals than ever relying on internet services and AI applications. In the process, many tend to paste or type personal data, bank information or company data into chat windows and forms without regard for the risk. Modern security software including endpoint security and firewalls only come into play after data is carried from the browser. The most common place where leakage happens is the browser itself is left exposed. It is accomplished to fill that gap by giving users an easy medium which protects their sensitive data before releasing it out.

Existing Problems That Require This Solution.

- Browser leaks

  The majority of the leaks happen when data is input in forms or chat windows. Traditional security tools do not monitor this stage, which leaves it as a weakness for attackers or unintentional exposure.

- Prompt injection attacks

  Attackers can craft malicious inputs that use Large Language Models (LLMs) to breach their security rules. It results in them revealing confidential information or creating perilous outputs.

- User errors

  Most of the leaks are made by users themselves. Employees, for example, in the Samsung Electronics case, duplicated confidential company data into ChatGPT and compromised trade secrets. Personal data, logins and financial credentials, or company identifiers are similarly leaked by users without their knowledge.

- Third-party plugin risks

  The majority of LLM platforms today use external plugins. These usually collect extra information with no set rules on usage and storage, increasing the risk of privacy violation and legal issues under regulations like GDPR and HIPAA.

- Legal and compliance issues

  Laws like GDPR and HIPAA require strong safeguarding of personal data. With the available solutions not protecting users at the input level, most organizations comply with danger.

- Blind faith in AI

There are some users who blindly accept LLM output as fully accurate and safe. They re-post or replicate this information without verification, potentially leading to secondary disclosure or disinformation.

- Overgeneralized protections

Present privacy measures anonymize all information, whether sensitive or not. This decreases usability and produces false positives but still misses some actual threats.

OBJECTIVES.

This project seeks to establish a strong, user-oriented defense against unintended sensitive information disclosures on browsers, including AI-driven interfaces. The aims are framed to progressively address the focuses of the limitation statement, ending in a working, tested, and public tool.

Main Objectives.

To build and empirically evaluate a browser extension that accurately detects and avoids accidental submission of sensitive information across traditional web forms and LLM chat interfaces.

Specific Objectives.

- To conduct a full analysis and specification of the technical patterns and surrounding context indicators of different types of sensitive data (E.g. PII, financial data (Credit Card numbers, bank accounts), login credentials, API keys, Corporate specific confidential identifiers).
- For developing a real time form input analysis and monitor detection engine with both high performance and low latency. This engine will use multi-layered filters including pattern matching (Regular Expressions), keyword search, and heuristic checks to identify some data leakage.
- Develop an easy-to-use intuitive alert interface to display a clear unobtrusive warning to users when they have submitted sensitive data and give them the choice and ability to cancel or continue with their submission.

- To conduct a thorough and comprehensive validation of the extension by means of a well-controlled test plan. This will include testing significant things like detection accuracy (maximizing true positives) false positive/negative rates and impacts on browser performance and system resources for a great user experience.

- Deploy final extension publicly publish on significant browser marketplace (eg. Chrome Web Store) and develop a framework for soliciting and analyzing user feedback to understand the actual usability, efficacy and marketability of the concept.

LITERATURE REVIEW

Technique Used	Approach	Author and Publications	Published Year
Machine Learning (ML) for Anomaly Detection & User Profiling	Proposed ML-based system to detect insider data leakage by analyzing user behavior patterns. Used SMOTE for class imbalance and OneHot encoding for preprocessing.	R. Ramachandiran et al., Data Leakage Detection Using ML	2023
Fine-tuned BERT for Named Entity Recognition (NER)	Used a fine-tuned BERTFor TokenClassification model for PII detection in documents, leveraging contextual embeddings.	Shraddha Patel & Dr. Parul Verma, Detecting Sensitive Information From Documents	2025
Comprehensive Survey of Privacy Threats in LLMs	Categorized privacy risks (prompt injection, membership inference, model inversion) and protections like Differential Privacy and Federated Learning.	Biwei Yan et al., On Protecting the Data Privacy of LLMs: A Survey	2024
Privacy-Preserving Hash & EmbeddingBased Auditing	Segmented user prompts into chunks, stored only cryptographic hashes and embeddings to detect prompt injections without storing raw data	Guozhi Hao & Jun Wu, Prompt Injection Detection for LLMs	2024

Prompt Injection Detection in LLMs	Developed a framework with probes, scanners, and evaluation metrics to detect adversarial prompt injections in LLMs.	Joseph et al., Prompt Injection in Large Language Model Exploitation: A Security Perspective (IEEE)	2025
Intelligent Sensitive Data Recognition in Long Text	Proposed an NLP + ML framework with multimodal features and user behavior analysis for long text data recognition	Hua Fu et al., Sensitive Data Recognition in Long Text (IEEE	2024
Textual Differential Privacy for LLMs	Introduced Differential Embedding Hash for anonymizing PII in context-aware reasoning while maintaining inference accuracy	Junwei Yu et al., Textual Differential Privacy for LLMs (IEEE)	2024

METHODOLOGY

This chapter outlines the systematic approach for developing the "**Browser Extension for RealTime Prevention of Sensitive Data Leaks**." The methodology encompasses the technology stack, development lifecycle, the core detection engine's architecture, a comprehensive testing plan and the personnel and facilities required for successful project execution

1. Implementation Approach

This project will adopt a systematic, agile and user-centric development approach. The core philosophy is to build a robust lightweight browser extension that acts as a proactive safeguard intercepting sensitive data at the point of entry the browser. The approach is multi-faceted.

Agile and Iterative Development: The project will follow an Agile methodology organized into two-week sprints. This allows for incremental development of features (e.g., regex engine, popup UI) continuous testing, and the incorporation of feedback, ensuring a high-quality and adaptable final product.

Multi-Layered Detection Engine: The technical heart of the solution is a dual-layer detection engine. This combines rule-based pattern matching (using Regular Expressions for structured data like credit cards and IDs) with keyword scanning (for unstructured confidential terms) to provide comprehensive coverage. This design prioritizes immediate effectiveness and real-time performance, with a defined pathway for integrating a machine learning module for future context-aware analysis.

Privacy by Design: The extension will operate entirely client-side on the user's machine. All input analysis occurs locally within the browser no sensitive user data is transmitted to external servers, ensuring privacy and compliance with data protection regulations like GDPR from the ground up.

User Empowerment Over Blocking: Instead of outright blocking user actions the extension is designed to warn and empower the user. By providing clear alert and actionable choices (Cancel, Edit, Proceed) it enhances security without sacrificing usability, making it a practical tool for daily use.

This approach ensures the development of a practical, scalable, and effective tool that directly addresses the critical gap in end-user, real-time data leak prevention identified in the problem statement.

2. Main Steps of the Implementation Process

We will follow an agile methodology. The task will be divided into small pieces and perfected step by step. The focus is to keep the extension simple and light, and it should run completely inside the browser.

- Identify and decide what types of sensitive information should be detected.
- Design the system workflow and popup alerts.
- Build the detection engine (regex + keyword + ML later).
- Create the popup interface for alerts.
- Test for usability, speed, and accuracy.

3. Software Requirements

- Development Tools.
    - **Code Editor** → VS Code.
    - **Node.js + npm/yarn** → for dependency management and bundling if using modern frameworks.
    - **Browser Developer Tools** → Chrome DevTools / Firefox DevTools for debugging.
- Programming Languages & APIs.

- **HTML, CSS, JavaScript/TypeScript** (core for browser extension).

- **Manifest v3 (for Chrome) / WebExtension API (for Firefox)**.

- **Regex / NLP libraries** (e.g., xregexp, natural.js, compromise.js) for sensitive data detection

➢ Python & ML Stack:

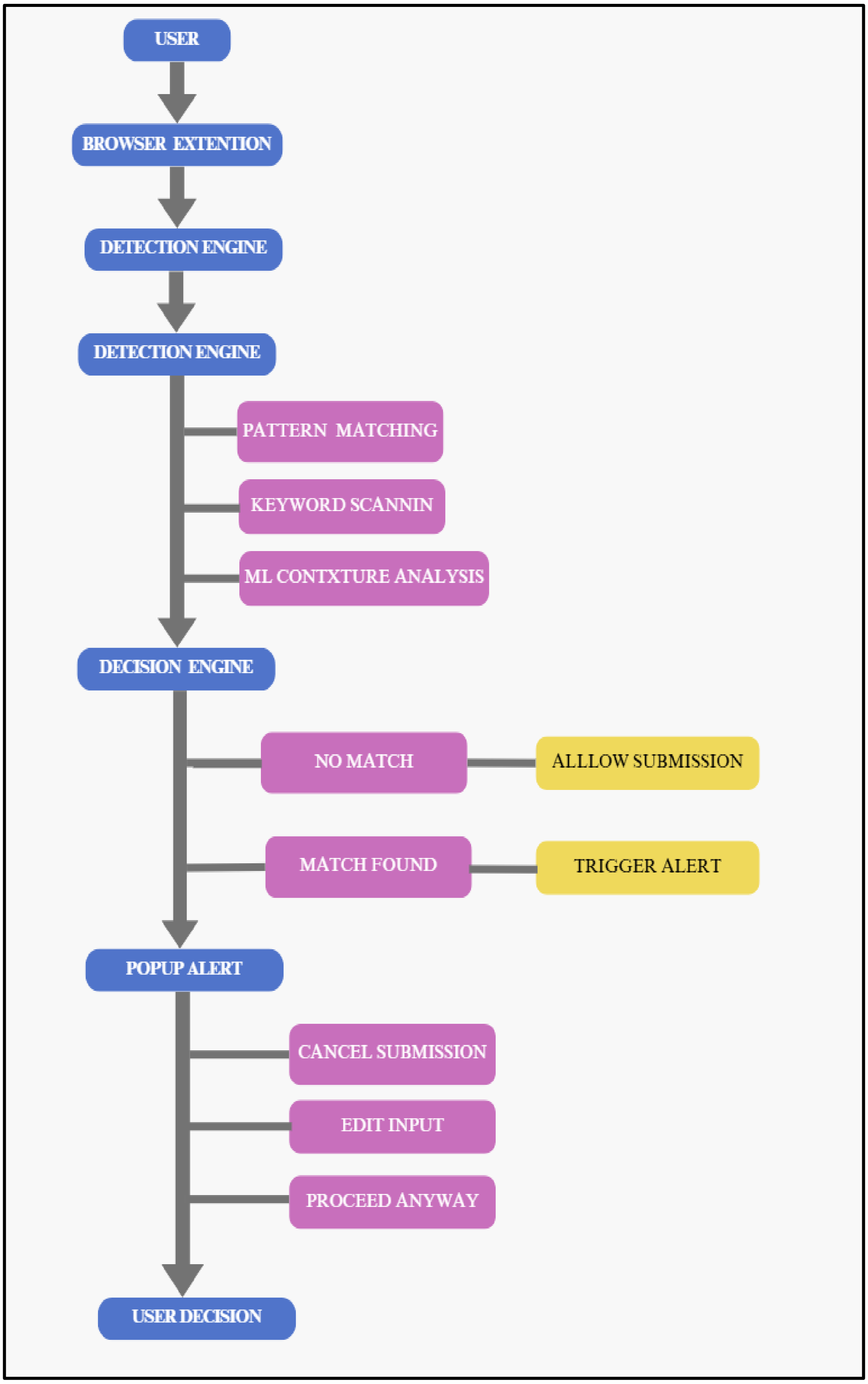
- **Python 3.9+** (recommended version for most modern ML libraries).
- **ML/NLP Libraries**:

  - scikit-learn → Traditional ML (Naive Bayes, SVMs, etc.).
  - spaCy or NLTK → NLP pipelines for detecting PII/sensitive data.
  - transformers (HuggingFace) → For LLM/advanced text classification.
  - pandas, numpy → Data processing

4. Hardware Requirements.

Hardware Component	Specification
CPU	Dual-core processor (Intel i3 / AMD equivalent).
RAM	4GB
Storage	500 MB of free space.

5. Big Picture of the Implementation Process.


USER
BROWSER EXTENTION
DETECTION ENGINE
DETECTION ENGINE
PATTERN MATCHING
KEYWORD SCANNIN
ML CONTXTURE ANALYSIS
DECISION ENGINE
NO MATCH
ALLLOW SUBMISSION
MATCH FOUND
TRIGGER ALERT
POPUP ALERT
CANCEL SUBMISSION
EDIT INPUT
PROCEED ANYWAY
USER DECISION

6. Project Workflow and Architecture.

The proposed system introduces a browser extension for sensitive information leak detection to Large Language Models (LLMs) and other web platforms. The workflow offers real-time monitoring, intelligent detection, and user-led decision making to decrease the risk of inadvertent disclosures. The entire architecture is shown in Figure X (the provided diagram).

1. User Interaction
The procedure begins whenever the user types in or attempts to post content (e.g., text, search requests, or form data) into a web application. The browser extension captures this content, acting as the monitor layer.

2. Detection Engine
Once the content has been captured, it is analyzed by the Detection Engine, which integrates a number of detection techniques to identify potential sensitive data:
Pattern Matching: Rule-based identification against pre-established patterns such as email patterns, phone number patterns, or credit card patterns.
Keyword Scanning: Identification of sensitive keywords such as "password," "API key," or "confidential."
Machine Learning Contextual Analysis: Advanced module powered by trained Python-based ML models. This module is beyond keyword and pattern analysis since it scans the semantic and contextual meaning of text to identify hidden sensitive information.

3. Decision Engine
The findings of the detection engine are passed on to the Decision Engine, which defines the appropriate action:
No Match: When no sensitive information is found, the submission is passed on without any hesitation.
Match Found: When sensitive content is found, the system raises an alert to notify the user.

4. Popup Alert
When a suspected data leak is detected, a popup alert is activated in the user's direction. The alert provides actionable choices:
Cancel Submission: Halt the action to avoid disclosure of sensitive data.
Edit Input: Allow the user to edit their content in order to delete or modify sensitive information.
Proceed Anyway: Permit submission despite warning, useful for false alarms or scheduled disclosures.

5. User Decision

The final step allows the user to have the final say. Through the addition of automatic discovery coupled with human insight, the system provides flexibility, user consciousness, and reduced opportunities for accidental information spills

REFERENCES.

[1] *Data leakage detection using ML*. (2023, November 17). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/10395027/authors#authors

[2] *Detecting Sensitive Information from Documents*. (2025, June 25). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/11089654

[3] *On Protecting the data privacy of Large Language Models (LLMs): a survey*. (2024, June 20). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/11062758

[4] *Privacy Leak Detection in LLM Interactions with a User-Centric Approach*. (2024, December 17). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/10945196

[5] *Privacy-Preserving healthcare data security using large language models and adaptive access control*. (2025, May 28). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/11105296

[6] *Privacy-Preserving prompt injection detection for smart Cloud-Deployed large language models*. (2025, May 9). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/11006851

[7*Prompt Injection in large language model Exploitation: A Security Perspective*. (2025, May 1). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/11064209

[8] *Research on intelligent sensitive data recognition and annotation technology based on long text*. (2024, May 24). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/10627741

[9] *Textual Differential Privacy for Context-Aware Reasoning with Large Language Model*. (2024, July 2). IEEE Conference Publication | IEEE Xplore. https://ieeexplore.ieee.org/document/10633584