**Project Report: Phishing Website Detection System**

**Introduction**

Phishing websites are a major cybersecurity threat, tricking users into revealing sensitive information such as login credentials and financial details. This project is designed to detect phishing websites using a **browser extension** and a **Flask-based backend server** that employs **machine learning models** for classification. The project integrates **URL-based analysis, webpage text content analysis, SSL certificate validation, and behavioural analysis** to improve phishing detection accuracy.

**Project Components**

The project consists of two main parts:

1. **AI Model (Backend - Flask Server)**
2. **Browser Extension (Frontend - Chrome Extension)**

**1. AI Model (Backend - Flask Server)**

The backend is built using Flask and includes various machine learning components to analyse URLs, webpage content, SSL validity, and user behaviour.

**Key Python Modules in the AI Model**

**(a) app.py (Main Flask Server)**

This is the central script that loads machine learning models and handles API requests from the browser extension. It performs the following tasks:

* Loads trained **URL-based phishing detection model**, trained using model.py (best\_model.pkl).
* Loads **text-based phishing detection model** (text\_model.pkl) and the corresponding **vectorizer** (vectorizer.pkl).
* Integrates **SSL certificate validation** (ssl\_check.py).
* Extracts webpage text content using content\_analysis.py.
* Performs **behavioural analysis** using behavioural\_analysis.py.
* Defines API endpoints:
  + /analyze-url – Classifies the URL based on machine learning features.
  + /analyze-behaviour – Detects suspicious behaviour from user interactions.
  + Handles errors with appropriate responses (400, 404, 500).

**(b) content\_analysis.py**

* Extracts visible text from a webpage using BeautifulSoup.
* Removes unnecessary elements like scripts and styles.
* Cleans the text by removing extra spaces and special characters.

**(c) ssl\_check.py**

* Checks the SSL certificate expiration of a given domain.
* Returns the **expiry date** and the **number of days left** before expiration.
* Helps identify phishing websites that often have expired or invalid SSL certificates.

**(d) behavioural\_analysis.py**

* Detects suspicious user behaviour such as:
  + **Rapid mouse movements**.
  + **Excessive clicks**.
  + **Fast keystrokes**.
* If any of these behaviours exceed predefined thresholds, the site is flagged as potentially malicious.

**(e) model.py & model2.py**

* These contain scripts for training the phishing detection models
* model.py focuses on **URL feature extraction** and training an XGBoost-based classifier
* model2.py might contain an alternative or improved version of the model
* Multi-layer security using model.py and model2.py

**(f) Data Files (Phishing\_Websites\_Data.csv & webpage\_text\_dataset.csv)**

* Contain datasets used for training and testing the models.
* Include features like **URL structure**, **keyword presence**, and **text content of phishing sites**.

**2. Browser Extension (Frontend - Chrome Extension)**

The frontend is a Chrome extension that captures user interactions and sends website data to the backend for analysis.

**Key Files in the Browser Extension**

**(a) manifest.json**

* The **configuration file** for the Chrome extension.
* Specifies permissions, background scripts, and content scripts.

**(b) background.js**

* Handles the core logic of the extension.
* Listens for events (such as URL visits) and sends the URL to the backend.
* Receives analysis results and displays warnings if a phishing site is detected.

**(c) content.js**

* Runs inside webpages to collect user behavior data.
* Captures **mouse movements, clicks, and keystrokes**.
* Sends this behavioral data to the backend for phishing detection.

**(d) popup.html & popup.js**

* Define the **user interface** of the extension.
* Displays results from backend analysis, warning users about potential phishing threats.

**How the System Works**

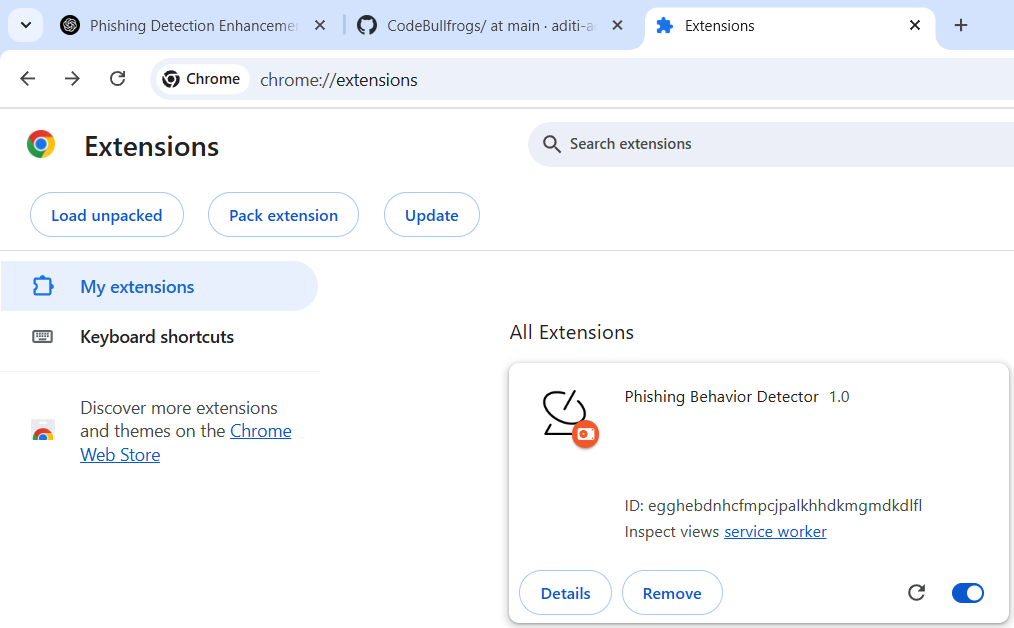
1. **User visits a website** → The browser extension captures the URL and user interactions.
2. **The extension sends data to the backend** → The backend analyzes:
   * URL structure using the **URL-based ML model**.
   * Webpage text content using the **text-based ML model**.
   * SSL certificate validity.
   * User behavior (mouse, click, keystrokes).
3. **Backend returns results** → If a phishing attempt is detected, the extension alerts the user.

**Conclusion**

This project combines **machine learning, cybersecurity, and browser automation** to create an advanced phishing detection system. It enhances security by leveraging **multi-modal analysis** (URL-based, text-based, SSL validation, and behavioral monitoring) to improve accuracy and reduce false positives. The **Flask backend** processes the data efficiently, while the **browser extension** provides a seamless user experience.

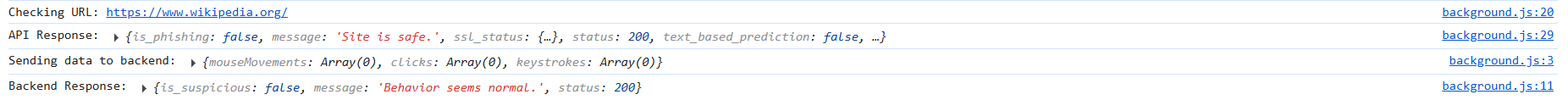
By integrating multiple detection techniques, this system offers a **robust and comprehensive approach** to detecting phishing websites, making the internet safer for users.

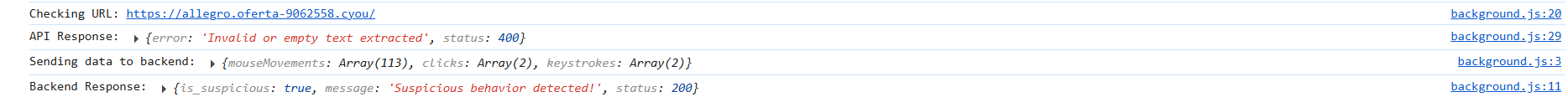
**Images for chrome extension (folder browser\_extension):**

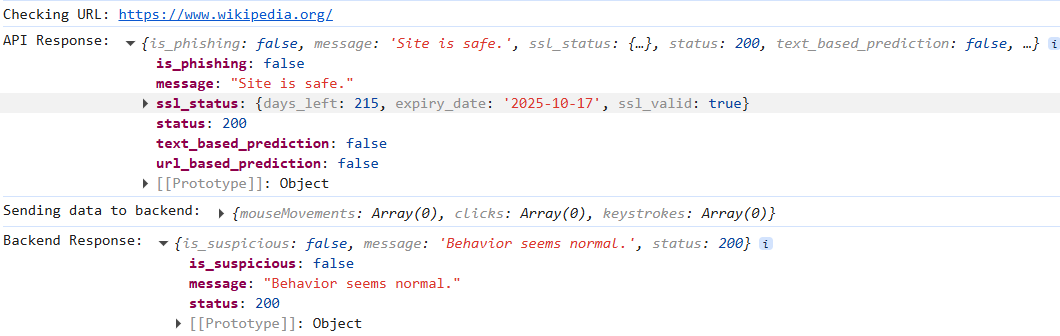


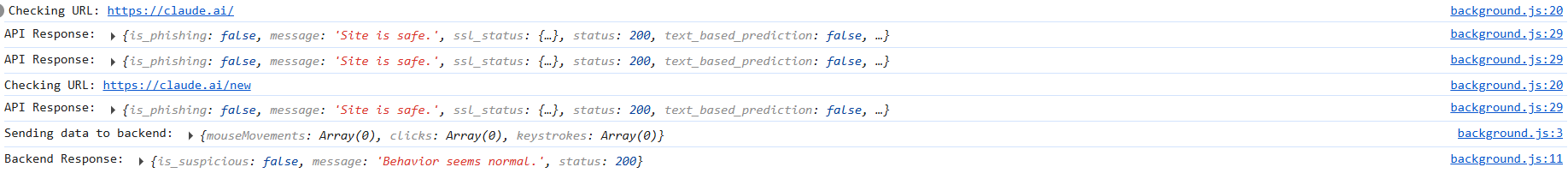
With developer mode “ON”

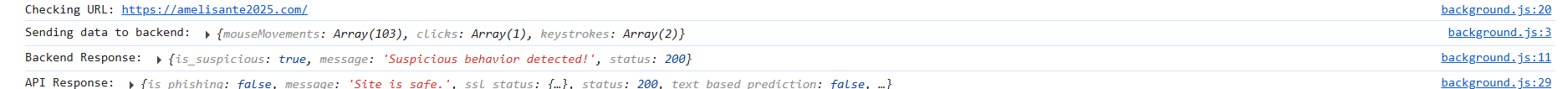
**Output from service worker:**



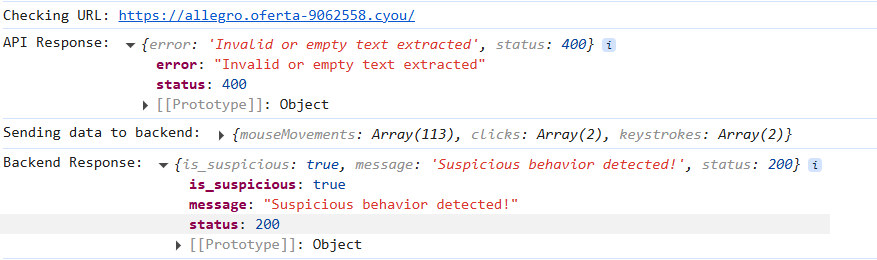








Sometimes it shows that it has suspicious behaviour but does not mark as phishing



Phishing sites normally show invalid text extraction and suspicious behaviour