

Florida International University
Knight Foundation School of Computing and Information Sciences

Cybersecurity Focus

Final Deliverable

Project Title:

Phishing Detection in Chrome Extension

Team Members: Juan Azcuna, Aiyanna Ferguson, Ivan Ortiz

Product Owner(s): Juan Azcuna

Instructor: Masoud Sadjadi

The MIT License (MIT)

Copyright (c) 2016 Florida International University

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Abstract

This document provides an overview of the research and steps taken to develop a Google Chrome extension for the purpose of detected phishing in emails within Gmail. It addresses the common problem of email phishing, also known as spear phishing, and serves to provide users with a phishing score and URL analysis flags to advise users of possible threats. To achieve this, Google's Gemini AI model was incorporated to analyze the email content using a prompt, and the VirusTotal v3 public API was integrated to fetch reports of links from available scanners. In addition, different permissions were integrated in the Chrome extension to ensure the functionality of the email analysis and link analysis feature, as well as user security by allowing access only to specific hosts. This document thoroughly outlines the development of PhishGuard, the steps required for usage, and ways the project can be improved for further development.

Table of Contents

Introduction.....	5-6
Current System.....	5-6
Purpose of New System.....	6
User Stories	6-7
Implemented User Stories	7
Project Plan	8
Hardware and Software Resources.....	8
Sprints Plan.....	9-16
System Design.....	17-20
Architectural Patterns	17
System and Subsystem Decomposition.....	17-20
Deployment Diagram.....	20
System Validation.....	21
Glossary.....	22-23
Appendix	24-34
Appendix A -UML Diagram.....	24
Appendix B -User Interface Design	25
Appendix C – Sprint Review Reports.....	25-30
Appendix D – User Manuals, Installation/Maintenance Document, Wishlist.....	31-34
References	35-36

INTRODUCTION

The technological evolution of online networks has led to the overall progression of social engineering. Social engineering is a technique that combines the use of deception and disguise to convince users to reveal sensitive information about themselves or an organization (Varshney et al., 2024). Over the years, social engineering has become the number one cybercrime (Klimburg-Witjes & Wentland, 2021), with phishing as the most common tactic. Phishing poses a significant threat to sensitive data, especially shared via email. With 46.6% of phishing incidents reported for the first half of 2024 highlighting webmail as one of the main target areas (“Phishing Activity Trends Report 1st Quarter 2024”; “Phishing Activity Trends Report 2nd Quarter 2024”), the need for proper application security and user security is vital for the overall safety of company and personal data.

With software becoming an integral part of businesses, application security is crucial to ensure the confidentiality, integrity, and availability of data. Application Programming Interfaces (APIs) play an important role in developing software. APIs create the foundation on which people can effectively utilize applications and systems (Ofoeda et al., 2019). When it comes to the use of web applications, browser extension APIs are also important to provide different functionalities to a webpage. The PhishGuard capstone project incorporates the use of Google’s built-in Gemini AI model and VirusTotal’s v3 public API to determine the security of emails and their embedded URLs within the Gmail web application.

Current System

Many applications exist that cater to phishing detection including platforms like VirusTotal, PhishTank and extensions like AntiPhish, DOMAntiPhish, LinkGuard, and many more (Bhopen Singh and Tahbildar). Some of these extensions utilize industry security algorithms like the

Symmetric Data Encryption Standard (DES) algorithm while others utilize their own to ensure data privacy and security. Both VirusTotal and PhishTank platforms have a public database with request limitations along with subscription options for users who are in need of quicker, more efficient results.

Purpose of New System

PhishGuard is a Chrome extension that incorporates modern, efficient tools capable of providing users with real-time phishing detection for emails within their inboxes. The use of Google Gemini makes it possible to develop the product with privacy and security in mind – allowing the data to be stored locally and not sent to external services. PhishGuard has the capacity to analyze the different aspects of an email and apply common phishing detection principles to determine the overall safety of emails present on Gmail. Furthermore, PhishGuard aims to further user awareness and knowledge of spear phishing and contribute to the overall decrease in successful phishing incidents per year. To achieve this, the extension was developed over the course of six sprints with two key features – one for email analyses, and the other for link analyses.

User Stories

The following section provides the user stories that were implemented during this iteration of the Phishing Detection in Chrome Extension project. These user stories provided below served as the basis for the implementation of the project's features.

Implemented User Stories

User Story 1: Recently, a business contacted us to voice their concerns about a recent attack they had that led to a breach in one of their databases. The attack was executed via email because an administrator clicked a link that led to downloading a virus in his computer. They have already more strict training for their employees but are looking for more options that could add an extra layer of security to their infrastructure.

User Story 5: As a project manager, I want to delegate tasks to each of my team members to further develop and test the Java, HTML, and CSS codes for the implementation of the API.

User Story 6: Adding a feature to the extension that allows you to right-click on a URL in the email, you get the option to analyze the link for malicious URLs, probably using an API or DB.

User Story 7: As a developer I want to improve the phishing score system and the whole structure of the main feature of the extension.

User Story 8: As a developer I want to improve the look of the extension using CSS and improve the HTML popup.

User Story 9: As a developer I want to design an icon that encompasses the main goal of this security feature: phishing detection via malicious links.

User Story 10: As a developer I want to design integrate the link analysis feature as developed in User Story 6 to provide a full-working product.

PROJECT PLAN

This section describes the planning that went into the realization of this project and the hardware and software resources used. This project incorporated agile development techniques, such as sprint planning meetings that are also detailed in this section. Some meetings have listed the same user stories as some user stories required more than one sprint to complete.

Hardware and Software Resources

The following is a list of all hardware and software resources that were used in this project:

- **Google Gemini AI Model**
- **Virus Total v3 public API**
- **Google Developers platform**
- **Visual Studio Code**
- **GitHub**
- **Gmail**
- **HTML**
- **JavaScript**
- **Cascading Style Sheets (CSS)**
- **Laptop Computers**

Sprints Plan

Sprint 1: Laying the Foundation

User Story 1: Recently, a business contacted us to voice their concerns about a recent attack they had that led to a breach in one of their databases. The attack was executed via email because an administrator clicked a link that led to downloading a virus in his computer. They have already more strict training for their employees but are looking for more options that could add an extra layer of security to their infrastructure.

User Tasks:

- Conduct research on the following areas:
 - Basic web development (HTML/CSS/JavaScript)
 - Chrome extension development
 - API integration
 - Security and privacy concepts (Knowledge of phishing, AI processing on-device)
 - Git knowledge
 - Unit testing
 - Introduction to machine learning language (MLL)
- Construct a business proposal using what was gathered from the research
- Present to the team the findings

User Story 2: As a project manager I want to make a discord for organizing tasks and improving communication in the team.

User Tasks:

- Create Discord server and manage (roles, channels, content) for the team
- Conduct daily meetings and collect user feedback
- Use feedback to improve server in performance for optimizing the development

User Story 3: As a project manager I want to specify the project roles and the people that will be using GitHub setting up the environment on their computers and letting them create PRs.

User Tasks:

- Define project roles
- Create/assign roles in GitHub repo
- Setup the local env for everyone working on the project
- Set scopes for the next tasks

User Story 4: As a team member I want to create a spreadsheet calendar with availability(short-term/long-term) for everyone to know eachother's available days to meet and at what times.

User Tasks:

- Create shared spreadsheet
- Add my time in the spreadsheet
- Decide as a group what are the best meeting dates and times

Sprint 2

User Story 1: Recently, a business contacted us to voice their concerns about a recent attack they had that led to a breach in one of their databases. The attack was executed via email because an administrator clicked a link that led to downloading a virus in his computer. They have already more strict training for their employees but are looking for more options that could add an extra layer of security to their infrastructure.

User Story 2: As a project manager I want to make a Discord group chat for organizing tasks and improving communication in the team.

User Story 5: As a project manager, I want to delegate tasks to each of my team members to further develop and test the Java, HTML, and CSS codes for the implementation of the API.

User Tasks:

- Set a weekly meeting schedule to begin working on the codes together
- Share individual progress with teammates to determine how the code is coming along
- Test codes and check on the progress via GitHub's Pull Requests

User Story 6: Adding a feature to the extension that allows you to right-click on a URL in the email, you get the option to analyze the link for malicious URLs, probably using an API or DB.

User Tasks:

- Research what is needed to execute this feature (Technology-wise, integration-wise).
- Develop a document with steps to use as a reference
- Begin to develop the feature and update teammates about the progress

User Story 7: As a developer I want to improve the phishing score system and the whole structure of the main feature of the extension.

User Tasks:

- Research what is needed to execute this feature correctly.
- Define the different types of DOMS in the Outlook app program.
- Define the prompts for Gemini-in-device Continue to develop the feature and update teammates about the progress

User Story 8: As a developer I want to improve the look of the extension using CSS and improve the HTML popup.

User Tasks:

- Research what is needed to execute this feature (Technology-wise, integration-wise).
- Develop a document with steps to use as a reference
- Begin to develop the feature and update teammates about the progress

Sprint 3

User Story 1: Recently, a business contacted us to voice their concerns about a recent attack they had that led to a breach in one of their databases. The attack was executed via email because an administrator clicked a link that led to downloading a virus in his computer. They have already more strict training for their employees but are looking for more options that could add an extra layer of security to their infrastructure.

User Story 5: As a project manager, I want to delegate tasks to each of my team members to further develop and test the Java, HTML, and CSS codes for the implementation of the API.

User Story 6: Adding a feature to the extension that allows you to right-click on a URL in the email, you get the option to analyze the link for malicious URLs, probably using an API or DB.

User Story 7: As a developer I want to improve the phishing score system and the whole structure of the main feature of the extension.

User Story 8: As a developer I want to improve the look of the extension using CSS and improve the HTML popup.

User Story 9: As a developer I want to design an icon that encompasses the main goal of this security feature: phishing detection via malicious links.

User Task:

- Design a draft for the icon that includes elements of this main idea Share the design with the team and gain feedback Make changes based off feedback and share with team one again. Once it is approved, mark the task as completed. Requirements for this icon is a deliverable of a 16x16, 32x32, 48x48, 128x128 in png file type

Sprint 4

User Story 1: Recently, a business contacted us to voice their concerns about a recent attack they had that led to a breach in one of their databases. The attack was executed via email because

an administrator clicked a link that led to downloading a virus in his computer. They have already more strict training for their employees but are looking for more options that could add an extra layer of security to their infrastructure.

User Story 5: As a project manager, I want to delegate tasks to each of my team members to further develop and test the Java, HTML, and CSS codes for the implementation of the API.

User Story 6: Adding a feature to the extension that allows you to right-click on a URL in the email, you get the option to analyze the link for malicious URLs, probably using an API or DB.

User Story 7: As a developer I want to improve the phishing score system and the whole structure of the main feature of the extension.

User Story 8: As a developer I want to improve the look of the extension using CSS and improve the HTML popup.

User Story 9: As a developer I want to design an icon that encompasses the main goal of this security feature: phishing detection via malicious links.

Sprint 5

User Story 1: Recently, a business contacted us to voice their concerns about a recent attack they had that led to a breach in one of their databases. The attack was executed via email because an administrator clicked a link that led to downloading a virus in his computer. They have already more strict training for their employees but are looking for more options that could add an extra layer of security to their infrastructure.

User Story 5: As a project manager, I want to delegate tasks to each of my team members to further develop and test the Java, HTML, and CSS codes for the implementation of the API.

User Story 6: Adding a feature to the extension that allows you to right-click on a URL in the email, you get the option to analyze the link for malicious URLs, probably using an API or DB.

User Story 7: As a developer I want to improve the phishing score system and the whole structure of the main feature of the extension.

User Story 8: As a developer I want to improve the look of the extension using CSS and improve the HTML popup.

User Story 9: As a developer I want to design an icon that encompasses the main goal of this security feature: phishing detection via malicious links.

Sprint 6

User Story 5: As a project manager, I want to delegate tasks to each of my team members to further develop and test the Java, HTML, and CSS codes for the implementation of the API.

User Story 6: Adding a feature to the extension that allows you to right-click on a URL in the email, you get the option to analyze the link for malicious URLs, probably using an API or DB.

User Story 7: As a developer I want to improve the phishing score system and the whole structure of the main feature of the extension.

User Story 8: As a developer I want to improve the look of the extension using CSS and improve the HTML popup.

User Story 9: As a developer I want to design an icon that encompasses the main goal of this security feature: phishing detection via malicious links.

Sprint 7

User Story 5: As a project manager, I want to delegate tasks to each of my team members to further develop and test the Java, HTML, and CSS codes for the implementation of the API.

User Story 6: Adding a feature to the extension that allows you to right-click on a URL in the email, you get the option to analyze the link for malicious URLs, probably using an API or DB.

User Story 7: As a developer I want to improve the phishing score system and the whole structure of the main feature of the extension.

User Story 8: As a developer I want to improve the look of the extension using CSS and improve the HTML popup.

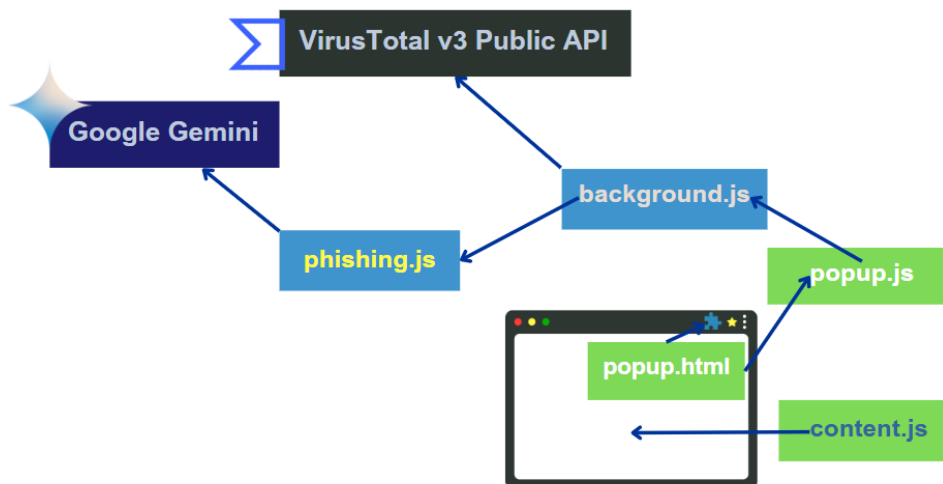
User Story 9: As a developer I want to design an icon that encompasses the main goal of this security feature: phishing detection via malicious links.

SYSTEM DESIGN

This section contains information on the design decisions that went into this project. The architecture patterns are outlined and explained. The entire system is shown in a package diagram and the subsystems are explained. Finally, the design patterns used in the project are discussed.

Architectural Patterns

Architectural Pattern



System and Subsystem Decomposition

To manage the requirements for the extension, two phases were undergone:

Phase One: Email Analysis (Main Function)

The following provides a code breakdown that describes how the different JavaScripts work together to analyze the emails within active tabs in Gmail.

Content JavaScript (content.js)

The content script is comprised of multiple functions that handle extracting the details from the email. The details that are extracted include the sender email address, the email's subject line, the time it was sent, and the email body.

Once this data is extracted it is sent to the background script.

Background Script (background.js)

The background script, also known as the service worker for the extension manages the background processes necessary for it to operate as intended. Once the extracted data is sent to this script, it utilizes the phishing.js script to begin the email analysis using Google Gemini.

Phishing Script (phishing.js)

This script provides the built-in AI model with a well-defined prompt to assess phishing indicators to determine whether or not the sender is legitimate and generate a phishing score from 0-100. A detailed explanation is also provided. The popup interface scripts are then used to display this information to the user.

Popup Interface Scripts (popup.html & popup.js)

The popup.html script entails what is printed on the screen. Both the popup.html and popup.js scripts define how the results are printed to the user. These scripts provide toggle options, and additional buttons – one that users can click on to report phishing and another to learn more about the extension.

Phase Two: Link Analysis

Describe here the code flow for the link analysis [aiyanna]

Several functions were defined in both the background and popup JavaScripts that allow the link analysis feature to function alongside the general email analysis.

Content Script (content.js) & Background Script (background.js)

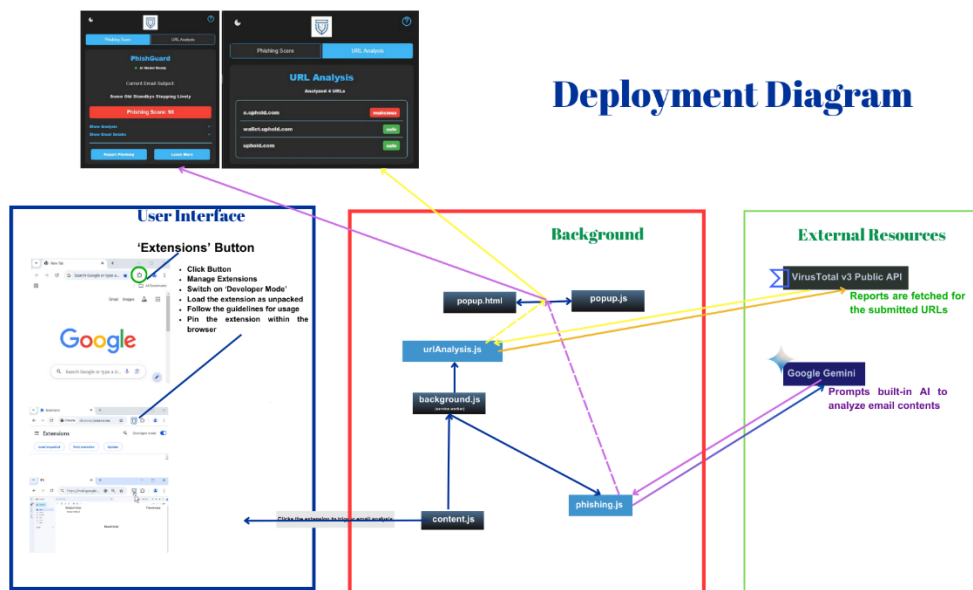
Upon the extraction of the email details described above, a link array is also formulated and sent to the background script. The background script sends the link array to the urlAnalysis script.

URL Analysis Script (urlAnalysis.js)

Once this script receives the link arrays, the domain which includes the protocol (https), and the site name – for instance *https://google.com*, is extracted from each URL. Considering the limitations of the public API, the domain name is compared to each domain included in the URLs. During the comparison, only the unique domain names (not matching the primary one, like “google”) will be extracted and sent to the API.

The responses from VirusTotal are received for each request. The status, positives value, and total amount of scans are then extracted, and the script sends a message to the popup interface. The interface listens for the message from the urlAnalysis.js script and sends these values to the popup interface. Using these values, along with the links – the users are provided with the number of URLs analyzed, the domain for the website, and a flag including ‘malicious’ and ‘safe’ depending on the report received from VirusTotal.

Deployment Diagram



SYSTEM VALIDATION

To ensure PhishGuard could process the email content extraction and analysis along with the URL safety analysis, the extension's development was carried out in two separate environments.

The **email analysis** portion was developed and debugged using the Google Developer Console. This involved coding the main scripts for the extension and integrating the built-in Gemini environment to facilitate testing. Bulk testing was then conducted to identify the errors in running the code to refine it and ensure efficient performance.

The **link analysis** portion was developed using Visual Studio Code alongside the Google Console, and 'Manage Extensions' interface. These tools helped to identify and improve upon issues within the code. However, the initial development of this feature faced challenges and was unable to yield the desired results due to "failed to fetch" errors and CORS policy restrictions.

To address these challenges, the link analysis features and elements of the code were integrated into the main extension's codebase. This required the creation of an additional JavaScript, *urlAnalysis.js*. This script streamlined scan and fetch requests to the VirusTotal API, effectively overcoming earlier errors and aiding in the overall functionality of the URL analysis feature.

By carrying out preliminary steps to develop the code, extensive bulk-testing, and implementing an additional code, PhishGuard was capable of analyzing both email content and URLs for safety – producing a fully functional extension.

GLOSSARY

Phishing is a form of cybercrime that involves deceiving individuals into revealing sensitive information, such as passwords, credit card numbers, or personal identification information.

API (Application Programming Interfaces) integration is the process of connecting two or more software applications or systems using them to enable seamless communication and data exchange.

Machine learning focuses on developing algorithms and statistical models that allow computers to perform tasks by learning patterns from data, rather than following pre-programmed rules.

Unit testing is a software testing technique that focuses on evaluating the smallest testable parts of an application, typically individual functions, methods, or classes, to ensure they work correctly in isolation.

GitHub allows users to store, manage, and track changes to their code repositories using Git, an open-source version control system & facilitates collaboration between other developers.

Programming languages such as HTML, Java, and CSS are used to create instructions that computers can execute, enabling the development of software, applications, and systems.

Document Object Models (DOM) allows developers to programmatically access, modify, and manipulate the content, structure, and style of web pages using programming languages.

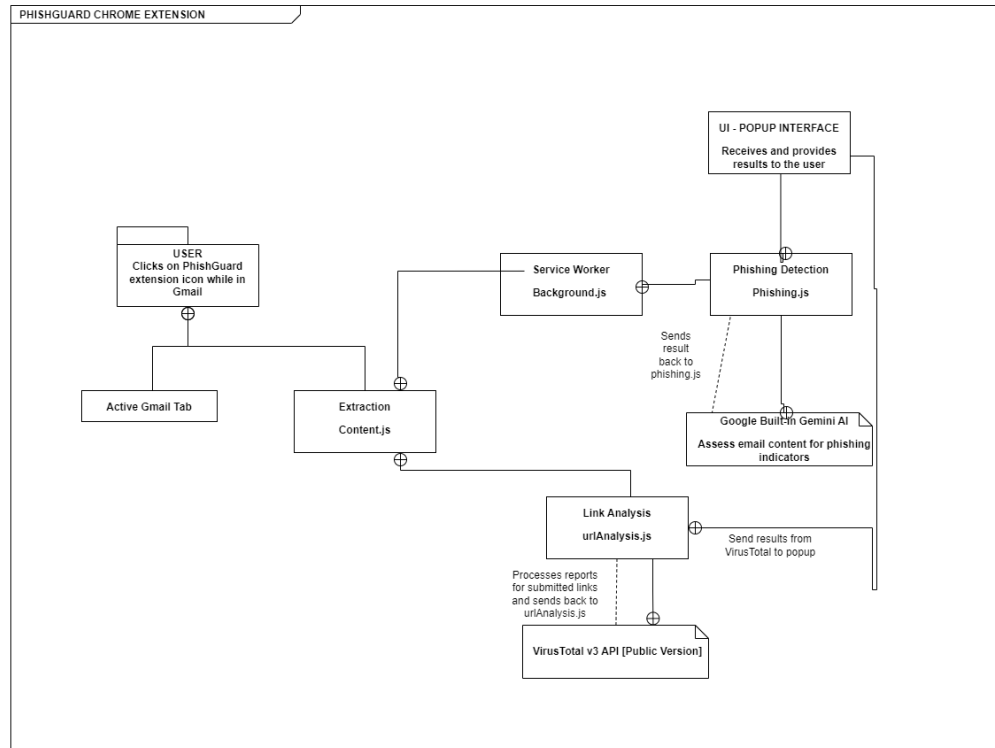
Gemini, developed by Google, is a generative artificial intelligence chatbot and large language model (LLM) that serves as a personal AI assistant.

Google Chrome extension development is built using web technologies such as HTML, CSS, and JavaScript, and can interact with web pages, modify the browser's user interface, and access various Chrome APIs to perform specific tasks

APPENDIX

Appendix A - UML Diagram

UML DIAGRAM



Appendix B - User Interface Design



Appendix C - Sprint Review Reports

Sprint Review 1

Attendees: Juan Azcuna, Aiyanna Ferguson, Ivan Ortiz

Start time: 4:00 PM [09/13/2024]

End time: 4:30PM [09/13/2024]

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

12/05/2024

Page 25 of 36

- **User Story 2:** As a project manager I want to make a discord for organizing tasks and improving communication in the team.
- **User Story 4:** As a team member I want to create a spreadsheet calendar with availability(short-term/long-term) for everyone to know each other's available days to meet and at what times. The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting:
- **User Story 1:** Recently, a business contacted us to voice their concerns about a recent attack they had that led to a breach in one of their databases. The attack was executed via email because an administrator clicked a link that led to downloading a virus in his computer. They already implemented a more strict training to their employees but are looking for more options that could add an extra layer of security to their infrastructure.

User Tasks: Conduct research on the following areas: ○ Basic web development (HTML/CSS and JavaScript) ○ Chrome extension development ○ API integration ○ Security and privacy concepts(Knowledge of phishing, ai processing on device) ○ Git knowledge ○ Unit testing ○ Introduction to machine learning language(MLL)

- **User Story 3:** As a project manager I want to specify the project roles and the people that will be using GitHub setting up the environment on their computers and letting them create PRs. User Tasks: Define project roles---- Create/assign roles in GitHub repo Setup the local env for everyone working on the project Set scopes for the next task.

Sprint Review 2

Project: Phishing Chrome Extension

Attendees: Juan Azcuna, Aiyanna Ferguson, Ivan Ortiz

Start time: 6:15 PM [09/27/2024]

End time: 7:15 PM [09/27/2024]

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- During the spring review meeting and considering our progress we found that more time would have to be contributed to different aspects of the project both theoretically and technically. With that, some of the user stories were moved to the backlog to be assigned to the future sprint.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting:

- **User Story 1:** Recently, a business contacted us to voice their concerns about a recent attack they had that led to a breach in one of their databases. The attack was executed via email because an administrator clicked a link that led to downloading a virus in his computer. They already implemented a more strict training to their employees but are looking for more options that could add an extra layer of security to their infrastructure.
- **User Stories 6, 7, 8:** ○ We believe because of the complexity of these user stories these will be about 3 sprints long to complete as per the difficulty and longevity of the tests that have to be done on the product

Sprint Review 3

Project: Phishing Chrome Extension

Attendees: Juan Azcuna, Aiyanna Ferguson, Ivan Ortiz

Start time: 8:30 PM[10/11/2024]

End time: 9:00PM [10/11/2024]

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- During the spring review meeting and considering our progress we found that more time would have to be contributed to different aspects of the project both theoretically and

technically. With that, some of the user stories were moved to the backlog to be assigned to the future sprint.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting:

- **User Story 1:** Recently, a business contacted us to voice their concerns about a recent attack they had that led to a breach in one of their databases. The attack was executed via email because an administrator clicked a link that led to downloading a virus in his computer. They already implemented a more strict training to their employees but are looking for more options that could add an extra layer of security to their infrastructure.
- **User Story 6, 7, 8:** ○ We believe because of the complexity of these user stories these will be about 3 sprints long to complete as per the difficulty and longevity of the tests that have to be done on the product

Sprint Review 4

Project: Phishing Chrome Extension

Attendees: Juan Azcuna, Aiyanna Ferguson, Ivan Ortiz

Start time: 6:00PM[10/27/2024]

End time: 6:30PM [10/27/2024]

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- During the spring review meeting and considering our progress we found that more time would have to be contributed to different aspects of the project both theoretically and technically. With that, some of the user stories were moved to the backlog to be assigned to the future sprint.

The following ones were rejected and moved back to the product backlog to be assigned to a future sprint at a future Sprint Planning meeting:

- **User Story 1:** Recently, a business contacted us to voice their concerns about a recent attack they had that led to a breach in one of their databases. The attack was executed via email because an administrator clicked a link that led to downloading a virus in his computer. They already implemented a more strict training to their employees but are looking for more options that could add an extra layer of security to their infrastructure.
- **User Story 6, 7, 8:** ○ We believe because of the complexity of these user stories these will be about 3 sprints long to complete as per the difficulty and longevity of the tests that have to be done on the product

Sprint Review 5

Project: Phishing Chrome Extension

Attendees: Juan Azcuna, Aiyanna Ferguson, Ivan Ortiz

Start time: 7:30 PM[11/08/2024]

End time: 8:00PM [11/08/2024]

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- Some of the user stories were moved to the backlog to continue working on in a future sprint considering more time is needed for those tasks.

The following ones were rejected and moved back to the product backlog to continue working on it in a future sprint:

- **User Story 6, 7, 8:** ○ Due to the complexity of these user stories these will be about multiple sprints to complete as per the difficulty and longevity of the tests that have to be done on the product

Sprint Review 6

Project: Phishing Chrome Extension

Attendees: Juan Azcuna, Aiyanna Ferguson, Ivan Ortiz

Start time: 6:30 PM[11/22/2024]

End time: 7:00PM [11/22/2024]

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- The remaining user stories [5-9] were moved to the backlog to continue working on considering more time is needed for those tasks

Sprint Review 7

Project: Phishing Chrome Extension

Attendees: Juan Azcuna, Aiyanna Ferguson, Ivan Ortiz

Start time: 7:30 PM [12/07/2024]

End time: 8:00PM [12/07/2024]

After a show and tell presentation, the implementation of the following user stories were accepted by the product owners:

- The implementation of user stories 1-9 were accepted by the product owners

Appendix D - User Manuals, Installation/Maintenance Document, Shortcomings/Wishlist Document and other documents

Installation

1. Clone the repository: git clone https://github.com/Dalosuuu/GmailPhishGuardian_A.git
2. Open Chrome and navigate to chrome://extensions/
3. Enable "Developer mode" if it's not already enabled.
4. Click on "Load unpacked" and select the cloned repository folder.
5. The extension should now be installed and active.
6. Follow the instructions below from official Google documentation to install built-in-AI and make sure the extension will work correctly.

Requirements

To try it out, you need to:

- be on Mac or Windows
- **Very important:** Make sure you have the required disk space.
- be on Chrome dev / canary (version ≥ 127)
- enable chrome://flags/#optimization-guide-on-device-model (set to **Enabled BypassPerfRequirement**)
- enable chrome://flags/#prompt-api-for-gemini-nano (set to **Enabled**)
- re-launch Chrome and revisit page
- Inspect the page and go to the console tab and type (await ai.languageModel.capabilities()).available;
 - If it returns readily then the model was installed correctly.
 - If it returns after-download is either still downloading or the model is not downloading. Go to Debug after-download
 - If it returns no then the model is not installed.

- Debug after-download/Check for the current version, try chrome://components and look for **Optimization Guide On Device Model**
 - this should have a version which is not 0.0.0.0. If it is, then the model is not downloading or currently downloading. You can try to click on the update button but don't click too many times.
 - eventually if still not working as of suggestion from the documentation wait a few minutes with the browser open to wait for the this will have the right value (after downloading the model), then you'll need to relaunch chrome and try again

For any other questions please refer to the Google official documentation on the gemini on device API:

https://docs.google.com/document/d/1VG8HIyz361zGduWgNG7R_R8Xkv0OOJ8b5C9QKeCjU0c/edit?tab=t.0

Requirements

Our Built-in AI program is currently focused on desktop platforms. In addition, the following conditions are required for Chrome to download and run Gemini Nano.

Aspect	Windows	MacOS	Linux
OS version	10, 11	≥ 13 (Ventura)	Not specified
Storage	At least 22 GB on the volume that contains your Chrome profile. <i>Note that the model requires a lot less storage. It's just for the sake of having an ample storage margin. If after the download the available storage space falls below 10 GB, the model will be deleted again.</i>		
GPU	Integrated GPU, or discrete GPU (e.g. video card).		
Video RAM	4 GB (minimum)		
Network connection	A non-metered connection		

	These are not necessarily the final requirements for Gemini Nano in Chrome.
	We are interested in feedback about performance aspects to further improve the user experience, and adjust the requirements.
	Not yet supported: • Chrome for Android • Chrome for iOS • Chrome for ChromeOS

Usage

1. Open Gmail in Chrome
 - a. Log in to your Gmail account using Google Chrome.
2. Interact with Emails
 - a. Open any email conversation.
 - b. The extension will automatically analyze the email and calculate a phishing score.
3. View Phishing Score
 - a. Click on the PhishGuard AI extension icon next to the address bar.
 - b. The popup will display the email's subject and phishing score.
4. View Email Details and Phishing Analysis
 - a. Click on Show Email Details to expand and view sender information, timestamp, and a snippet of the email body.
 - b. Click on Show Phishing Analysis to view the AI's analysis of the email.
5. Toggle Theme
 - a. Use the theme toggle button in the top-left corner of the popup to switch between light and dark modes.
6. Report Phishing
 - a. If you suspect an email is phishing, click on Report Phishing to notify Google and help protect others.
7. Link Analysis
 - a. To utilize the link analysis, right-click any link within your email and select 'Analyze Link'

Wishlist

Premium VirusTotal 3 API

Utilizing the premium version of the VirusTotal version 3 API will allow us to make more requests per day for each URL, providing users with more feedback regarding those embedded in their emails.

Backend

Backends offer the ability to manage data and allow for the scalability of requests. With that, by having a backend, the API requests can be better managed, and it would also ensure product security and protect the unique API key from exposure.

Gmail API

Implementing the Gmail API would retrieve the metadata from the email to improve the phishing score calculation, providing users with more accurate scores.

PhishGuard ML Model

The creation of a machine learning model compatible with Chrome will allow for more accuracy and less hardware requirements – improving the overall usability and development of the project.

REFERENCES

- Varshney, Gaurav, et al. "Anti-Phishing: A Comprehensive Perspective." *Expert Systems with Applications*, vol. 238, 15 Mar. 2024, p. 122199, doi:10.1016/j.eswa.2023.122199.
- "Protect User Privacy." Chrome for Developers, 2018, developer.chrome.com/docs/extensions/develop/security-privacy/user-privacy.
- "Extensions / Develop." Chrome for Developers, developer.chrome.com/docs/extensions/develop.
- Klimburg-Witjes, N., & Wentland, A. (2021). Hacking humans? Social engineering and the construction of the "deficient user" in cybersecurity discourses. *Science, Technology, & Human Values*, 46(6), 1316–1339. <https://doi.org/10.1177/0162243921992844>
- Ofoeda, J., Boateng, R., & Effah, J. (2019). Application Programming Interface (API) Research. *International Journal of Enterprise Information Systems*, 15(3), 76–95. <https://doi.org/10.4018/ijeis.2019070105>
- "Phishing Activity Trends Report 1st Quarter 2023." APWG, 2 Nov. 2023.
- "Phishing Activity Trends Report 1st Quarter 2024." APWG, 14 May 2024.
- Moustafa, Ahmed A., et al. "The Role of User Behaviour in Improving Cyber Security Management." *Frontiers in Psychology*, vol. 12, no. 12, 18 June 2021, <https://doi.org/10.3389/fpsyg.2021.561011>.
- "Phishing Activity Trends Report 2nd Quarter 2024." APWG, 21 Aug. 2024.
- Qi, Yan, and Rui Xu. "Research on User Interface Design and Interaction Experience: A Case Study from "Duolingo" Platform." *ICST Transactions on Scalable Information Systems*, 4 Apr. 2024. ResearchGate, [www.researchgate.net/publication/379578564_Research_on_User_Interface_Design_and_Interaction_Experience_A_Case_Study_from_Duolingo_Platform](https://doi.org/10.4108/eetsis.5461), <https://doi.org/10.4108/eetsis.5461>.
- Wang, Yan, et al. "Simple = Authentic: The Effect of Visually Simple Package Design on Perceived Brand Authenticity and Brand Choice." *Journal of Business Research*, vol.

166, 1 Nov. 2023, p. 114078. *ScienceDirect*,
www.sciencedirect.com/science/article/abs/pii/S0148296323004368,
<https://doi.org/10.1016/j.jbusres.2023.114078>.

Bhopen Singh, Oinam, and Hitesh Tahbildar. “A Literature Survey on Anti-Phishing Browser Extensions.” *International Journal of Computer Science & Engineering Survey*, vol. 6, no. 4, 31 Aug. 2015, pp. 21–37, <https://doi.org/10.5121/ijcses.2015.6402>.