Attack of the Clones!

Functional Specification

Conor Lewis – C00246763
Project Supervisor – Christopher Staff

Abstract

Website Cloning is the copying of code from one website and duplicating it on another website, usually with alterations made to suit the cloners need. This comes with a risk for both web developers and internet users.

The goal of this functional specification is to document the methods involved in the developing an application that will identify when a user has accessed a fraudulent cloned website or an authentic original website.

Contents

Introduction

This functional specification document is for my fourth-year project titled Attack of the Clones, the goal of this project is to develop an application that can notify users when the website they have visited is a fraudulent cloned website or an authentic original website.

The application will be able to distinguish between a fraudulent cloned website and an authentic original website by running a series of tests on the information about the website and the information found on the website itself, the results of these tests will then be used to calculate the likelihood that a website is fraudulent.

Project Scope

The objective of this project is to develop an application will be able to distinguish between a fraudulent cloned website and an authentic original website by running a series of tests on the information about the website and the information found on the website itself, the results of these tests will then be used to calculate the likelihood that a website is fraudulent, as well as include an informative section to the application that will educate website developers on ways that they can reduce the risk of their website from being cloned.

Assumptions

This application assumes that the user is running a windows operating system and the user is using the browser Google Chrome. The application also assumes that the user has a reliable internet connection in order to perform certain internet queries as part of the tests the application must conduct.

Risks

There are certain security features that must be considered whilst developing the application in order to ensure that the user is safe whilst utilizing the application.

One key risk in this application is that a bad actor may try to intercept the traffic, once the bad actor has done this, they could either alter the verdict displayed to the user convincing them that the website they have visited is an authentic original website instead of a fraudulent cloned website. The bad actor may choose to just read the contents instead viewing the information displayed to the user.

Another risk that may be present in this application is a bad actor choosing to make a fraudulent clone of the application. The bad actor may place malware within the fraudulent application and trick the user into downloading their version of the application over the authentic original application.

The implementation of these risks as well as their mitigation methods are expanded upon within the application are expanded on within the misuse case diagram.

Deliverables

The goal of the application is to provide the following deliverables from the application:

Core Deliverables

- To provide a browser extension that can be installed on the Google Chrome browser.
- Enable the browser extension to interact with an underlying application that lies on the users' machine.
- Create a pop-up notification on the browser extension to notify the user when the application detects that the user has visited a suspected fraudulent cloned website.
- Secure the transfer of information between the application and the browser extension.
- Allow the user to interact with the browser extension in order to activate and deactivate the browser extension and the underlying application.

Non-Core Deliverables

- Create an informational section on the application that will provide information on reducing the risk of website cloning happening to a website the user is developing

Technologies

Programming Language

Python

The programming language I will be using in order to develop this application is Python. Python is a high-level programming language that I have found to most suitable in the development of this application, this is due to the many libraries that are readily available for use within the python programming such as the Natural Language Toolkit and Gazpacho, which are beneficial in developing this application. The python programming also dynamically typed which means that variables do not need to be declared, allowing for easier integration of various datatypes within the application.

Libraries

Natural Language Toolkit[1]

Natural Language Toolkit is a python library that allows natural language processing to take place within the application. This allows the application to take in human language and process the language as part of the application, such as checking for grammatical errors within a website. I will also be using the Natural Language Toolkit in order to perform Named Entity Recognition allowing me to identify information such as the brand that is associated with the website I am scanning.

[1] NLTK (2009). *Natural Language Toolkit — NLTK 3.4.4 documentation*. [online] Nltk.org. Available at: https://www.nltk.org/.

BeautifulSoup[2]

BeautifulSoup is a python library that allows the application to parse HTML information within the application. It scrapes the website and can take information from the website such as the contents of the website. The BeautifulSoup library is better equipped than many other HTML parsing libraries making it ideal for this program as it deals with a wide range of websites

Flask[3]

The Flask python library allows the use of python in order to develop web applications, in the case of this program the flask program is used in order to hold a server that the browser extension can call upon to engage in inter-process communication with the underlying browser to send messages to the underlying application and receive message from the underlying application.

Flask_CORS[4]

Flask_CORS is a python library used to enable Cross-Origin Resource Sharing for web applications that have been developed with Flask. This enables the permissions required in order for the browser extension in order to communicate with the browser extension.

Urllib.parse

Urllib.parse library offers many functions when operating with URLs. As part of this program, we are using this library for its ability to extract just the domain from a URL that is

[2] beautiful-soup-4.readthedocs.io. (n.d.). *Beautiful Soup Documentation — Beautiful Soup 4.4.0 documentation*. [online] Available at: https://beautiful-soup-4.readthedocs.io/en/latest/.

[3] flask.palletsprojects.com. (n.d.). *Welcome to Flask — Flask Documentation (2.2.x)*. [online] Available at: https://flask.palletsprojects.com/en/2.2.x/.

[4] Readthedocs.io. (2013). *Flask-CORS — Flask-Cors 3.0.7 documentation*. [online] Available at: https://flask-cors.readthedocs.io/en/latest/.

docs.python.org. (n.d.). *urllib.parse — Parse URLs into components — Python 3.8.3 documentation*. [online] Available at: https://docs.python.org/3/library/urllib.parse.html.

passed to the underlying python application from the browser extension, providing a layer of security for the users.

sklearn[5]

Sklearn is a python library that provides a number of packages that offer an array of functions, in terms of this project we are interested in the package cosine_similarity as this allows for calculating the similarity between two separate pieces of text in this case, we are checking for the similarity between the contents of two websites.

Duckduckgo_search[6]

Duckduckgo_search is a python library that enables a user to make internet search queries using the Duckduckgo search engine. This library allows for more consecutive searches to be made that other search engine python libraries at a time, which is ideal for this project.

WhoIs[7]

The WhoIs python library allows a python program to gain WhoIs metadata of a website and be able to extract import details all within the python program. The library does this by communicating back and forth with WhoIs servers and then storing this information in a way that application can see specific pieces of information extracted.

[5] Scikit-learn (2019). *scikit-learn: machine learning in Python — scikit-learn 0.20.3 documentation*. [online] Scikit-learn.org. Available at: https://scikit-learn.org/stable/index.html.

[6] PyPI. (n.d.). *duckduckgo-search: Search for words, documents, images, news, maps and text translation using the DuckDuckGo.com search engine.* [online] Available at: https://pypi.org/project/duckduckgo-search/.

[7] Penman, R. (n.d.). *python-whois: Whois querying and parsing of domain registration information.* [online] PyPI. Available at: https://pypi.org/project/python-whois/.

Json[8]

The Json python library allows for the python application to use Json format on data within the application. In this application a json file is used so the json library is used to read the information present within the json file.

Socket[9]

The socket python library is used by python applications when they wish to perform socket connections in order to communicate with other machines over the internet, in this example we must use a socket connection to a webserver

SSL[10]

SSL python library enables the underlying connection to create a secure SSL/TLS connection to the webserver it has a socket connection to, doing this we can confirm that there is a SSL certificate for the website we have connected to and view the SSL certificate information.

datetime[11]

Datetime library is used when using dates and times within the python application in this case we use datetime to ensure that range the SSL certificate is falls time that is declared acceptable time stated on the certificate.

[8] docs.python.org. (n.d.). *json — JSON encoder and decoder — Python 3.8.3rc1 documentation*. [online] Available at: https://docs.python.org/3/library/json.html.

[9] Python.org. (2020). *socket — Low-level networking interface — Python 3.8.1 documentation*. [online] Available at: https://docs.python.org/3/library/socket.html.

[10] Python.org. (2018). *ssl — TLS/SSL wrapper for socket objects — Python 3.7.2 documentation*. [online] Available at: https://docs.python.org/3/library/ssl.html.

[11] Python Software Foundation (2002). *Datetime — Basic Date and Time Types — Python 3.7.2 Documentation*. [online] Python.org. Available at: https://docs.python.org/3/library/datetime.html.

Requests[12]

The requests library is used in order to perform HTTP requests from within a python application. In this python application we are required to make requests to webpages at multiple stages within the application in order to harvest information from within the website.

Re[13]

The re python library is used in order to check for patterns within data. This allows the HTML parser to remove any HTML code that would be present on the website enabling higher accuracy when the data undergoes natural language processing as the special characters will no longer throw off results.

WordNinja[14]

The wordninja library enables the splitting of words within a string of text, preventing unintentional or unintended catenation occurring in strings of text. Removing the unintended concatenation leads to higher accuracy for any natural language processing that occurs on the information present on the webpage

[12] Reitz, K. (n.d.). *requests: Python HTTP for Humans.* [online] PyPI. Available at: https://pypi.org/project/requests/.

[13] Python (2009). *re — Regular expression operations — Python 3.7.2 documentation*. [online] Python.org. Available at: https://docs.python.org/3/library/re.html.

[14] Anderson, D. (2023). *Word Ninja*. [online] GitHub. Available at: https://github.com/keredson/wordninja.

Software

Google Chrome[15]

Google Chrome is required in the development of this application as this is the browser that the browser extension is being developed for. Google Chrome makes creating custom browser extensions easy to integrate due to their developer mode which makes testing your own browser extension rather simple.

Context Diagram

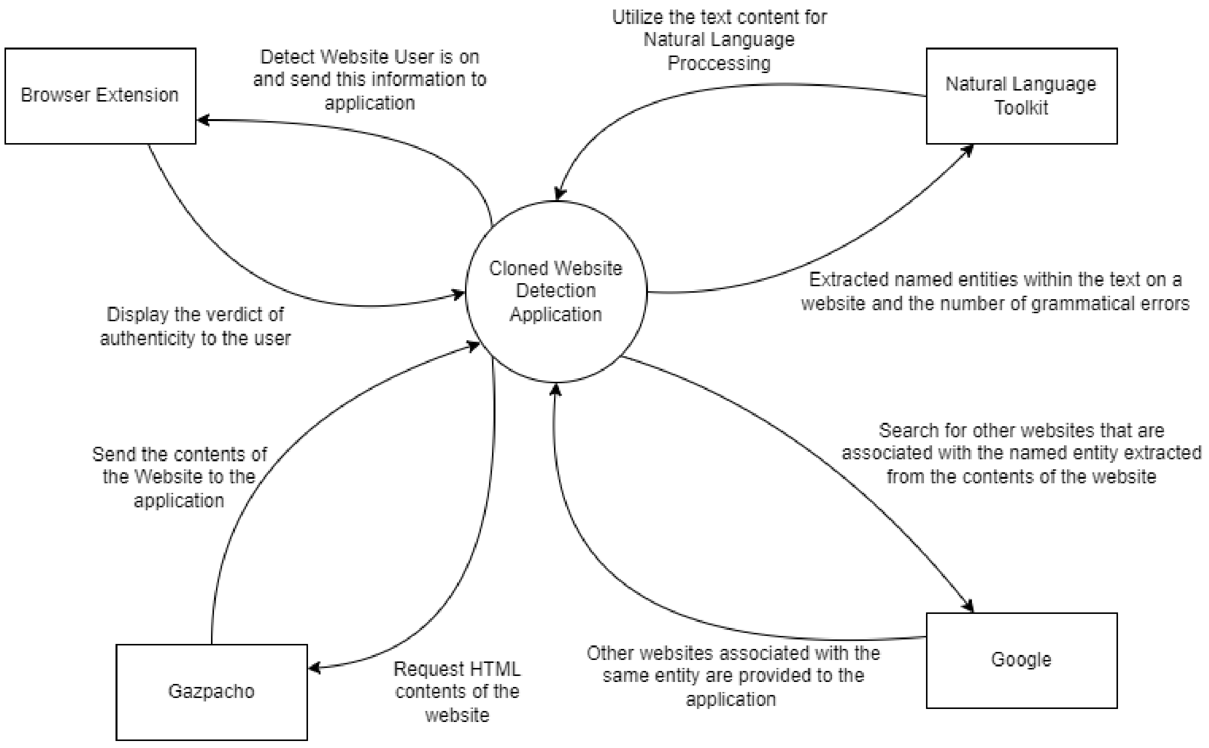
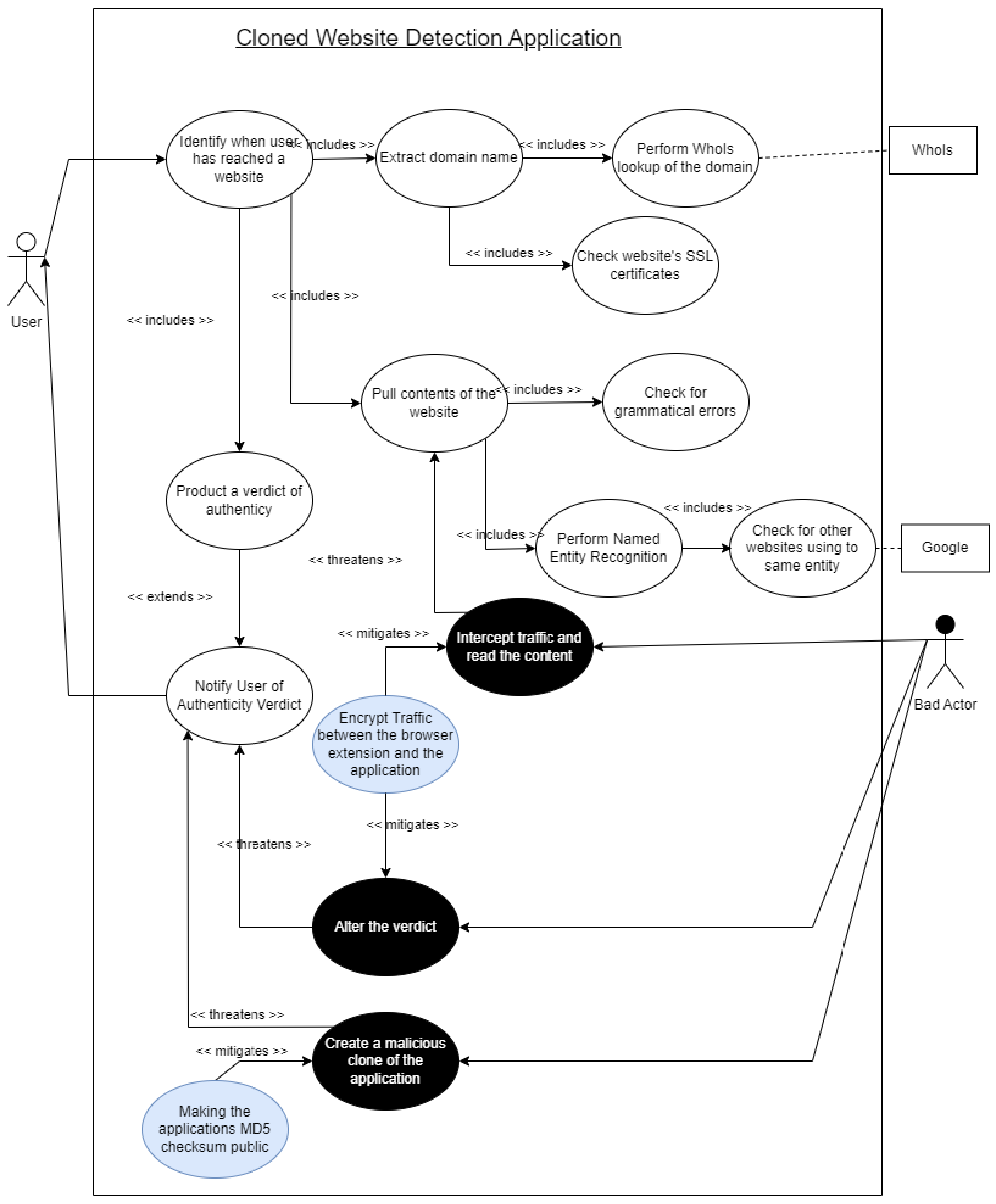

Browser Extension
Detect Website User is on and send this information to application
Utilize the text content for Natural Language Proccessing
Natural Language Toolkit
Cloned Website Detection Application
Extracted named entities within the text on a website and the number of grammatical errors
Display the verdict of authenticity to the user
Send the contents of the Website to the application
Search for other websites that are associated with the named entity extracted from the contents of the website
Gazpacho
Request HTML contents of the website
Other websites associated with the same entity are provided to the application
Google

Figure 1: Context Diagram

[15] Google.com. (2017). *Google Chrome - The Fast, Simple and Secure Browser from Google*. [online] Available at: https://www.google.com/chrome/.

Misuse Case Diagram



Cloned Website Detection Application
User
Identify when user has reached a website
<< includes >>
Extract domain name
<< includes >>
Perform WhoIs lookup of the domain
WhoIs
<< includes >>
Check website's SSL certificates
<< includes >>
<< includes >>
Pull contents of the website
<< includes >>
Check for grammatical errors
Product a verdict of authenticity
<< includes >>
Perform Named Entity Recognition
<< includes >>
Check for other websites using to same entity
Google
<< threatens >>
<< extends >>
<< mitigates >>
Intercept traffic and read the content
Bad Actor
Notify User of Authenticity Verdict
Encrypt Traffic between the browser extension and the application
<< mitigates >>
<< threatens >>
Alter the verdict
<< threatens >>
<< mitigates >>
Create a malicious clone of the application
Making the applications MD5 checksum public

Figure 2: Misuse Case Diagram

Use Cases

Identify when a user has reached a website

The browser will identify when the user has reached a new website and it will send this information to the application installed on the user's machine.

Primary Actor: User.

Stakeholder: User.

Preconditions: The user has both the browser extension and the application installed on their system and currently running on their system.

Triggers: The user has visited a website.

Expected Outcomes: The browser extension has been alerted to the actions of the user.

Extract Domain Name

The domain name of the website has been extracted by the browser extension and sent to the underlying application for testing

Primary Actor: User.

Stakeholder: User.

Preconditions: The user has visited a website.

Triggers: The browser extension has detected the user has accessed a new website.

Expected Outcomes: The browser extension will take the domain name of the website the user has just visited and will send this information to the underlying application that resides on the user's machine.

Perform WhoIs lookup

The underlying application takes in the domain name and searches the information in order to detect flags that the website is a fraudulent cloned website.

Primary Actor: User.

Stakeholder: User.

Preconditions: The domain name has been sent to the underlying application and the user still has suitable internet connection in order to gain information from WhoIs

Triggers: The domain name has been sent to the application.

Expected Outcomes: WhoIs information has been gathered, and the application can check for information such as who owns the domain. This information can then be used to identify discrepancies between a legitimate website and a fraudulent cloned website.

Check website's SSL certificates

The browser will identify when the user has reached a new website and it will send this information to the application installed on the user's machine.

Primary Actor: User.

Stakeholder: User.

Preconditions: The domain name has been sent to the underlying application and the user still has suitable internet connection in order to view website's SSL certificates.

Triggers: The domain name has been sent to the application.

Expected Outcomes: Can check whether SSL certificates match the authentic original website or if they contain indicators of being a fraudulent clone. The result of this will be used in the conclusion of authenticity.

Pull the website's contents

The application will gather information from the website itself such as the text contents of the website in order to process in order to conduct tests on the information found within the website itself.

Primary Actor: User.

Stakeholder: User, Bad Actor.

Preconditions: The browser extension and the application are running on the machine and the user still has internet connection so the application can communicate with the website.

Triggers: Browser extension has identified the user has accessed a website and its information has been sent to the underlying application.

Expected Outcomes: The information found will be placed into the application in a form that it can be processed by the application for future testing.

Check for grammatical errors

The application will use the Natural Language Toolkit library in order to detect how many grammatical errors are found within a website, a key indicator of a fraudulent cloned website.

Primary Actor: User.

Stakeholder: User.

Preconditions: The user has the application running on the user's machine.

Triggers: The websites contents have been parsed into the application.

Expected Outcomes: The number of grammatical errors has been calculated by the application.

Perform Names Entity Recognition

The application has detected any named entities that have occurred within the application in order to identify any organization that is affiliated with the website.

Primary Actor: User.

Stakeholder: User.

Preconditions: The user has the application running on the user's machine.

Triggers: The websites contents have been parsed into the application.

Expected Outcomes: Named entities within the website's contents has been detected are prepared for future testing within the application.

Check for other websites using the same entity

The application will perform google searching in order to identify other websites that are affiliated with the entity that was detected in the previous use case.

Primary Actor: User.

Stakeholder: User.

Preconditions: The user has the application running on the user's machine and a reliable internet connection in order to perform the google searches.

Triggers: Named Entity Recognition extraction has been performed on the website's contents

Expected Outcomes: If there are other websites affiliated with the same organization this information has been gathered to see if the current website is a fraudulent clone of these websites or if this is the authentic original website.

Produce a verdict of authenticity

The application takes in the results of all tests performed in earlier use cases in order to conclude the authenticity of the website.

Primary Actor: User.

Stakeholder: User.

Preconditions: All tests on the website have been performed by the application.

Triggers: The final test has been performed.

Expected Outcomes: The application uses the information gathered during all tests in order to calculate the likelihood the website the user visited is an authentic original website or a fraudulent cloned website.

Notify User of Authenticity

Using the verdict, the application has concluded the user is notified if the website has been deemed a fraudulent clone.

Primary Actor: User.

Stakeholder: User, Bad Actor.

Preconditions: The browser extension and the underlying application are both running on the user's machine.

Triggers: The underlying application has produced a verdict stating the website is a guaranteed fraudulent cloned or likely a fraudulent clone.

Expected Outcomes: If the application has concluded that the website the user has visited a likely or guaranteed fraudulent clone then the browser extension will provide a pop-up to warn the user to take care of the website they are visiting.

Intercept Traffic and Read their Contents

One goal of a bad actor who intents to cause harm using the application may be to intercept traffic between the application and the browser extension so they can read any information present on the website the user is visiting.

Primary Actor: Bad Actor.

Stakeholder: User, Bad Actor.

Preconditions: The application and the browser extension are running on the users' machine and communicating in an unsafe manner.

Triggers: The browser extension has sent the domain name to the application.

Expected Outcomes: The bad actor can view all information present on the website.

Alter the Verdict

The bad actor may wish to convince the user that their website is authentic, to do so they would have to alter the verdict that the application has concluded to say their website is authentic.

Primary Actor: Bad Actor.

Stakeholder: User, Bad Actor.

Preconditions: The application and the browser extension are running on the users' machine and communicating in an unsafe manner.

Triggers: The application is notifying the user of authenticity verdict.

Expected Outcomes: The bad actor can change a fraudulent clone verdict to a authentic original verdict.

Create a Malicious Clone of the Application

The bad actor may create a similar application but imbed malware within the clone and trick the user to download their clone of the application rather than the original safe software.

Primary Actor: Bad actor.

Stakeholder: User, Bad actor.

Preconditions: The bad actor has identified the cloned website detection application.

Triggers: The bad actor has developed a clone of the fraudulent website detection tool

Expected Outcomes: The bad actor can convince the user to install a clone of the fraudulent clone detection tool that has malware within it. Once the application is installing the malware has free reign over the user's PC.

Encrypt Traffic Between the Browser Extension and The Application

Whilst in the data is in transit between the browser extension and the underlying application the data is encrypted making it much more difficult for bad actor to alter data in transit

Primary Actor: User.

Stakeholder: User, Bad actor.

Preconditions: The user has both the browser extension and the application installed on their system and currently running on their system.

Triggers: The bad actor tries to intercept traffic and read the contents, or the bad actor tries to alter the verdict.

Expected Outcomes: The bad actor cannot intercept traffic between the browser extension and the application in order to read or alter the contents within.

Making the MD5 Checksum Public

The MD5 checksum of the application is made public knowledge so the user can confirm the file they have downloaded is the same as the one that has been uploaded by the developer of the application.

Primary Actor: User.

Stakeholder: User, Bad actor.

Preconditions: The application has been completed by the developer and has the Checksum is updated with each update of the application.

Triggers: The developer has calculated the MD5 checksum of the application file.

Expected Outcomes: The user can perform a check that the checksum of the file they have downloaded is the same as the public MD5 checksum as the one made publicly available by the application developer to make an additional layer of security for the user.

FURPS+

Functionality

The functionality of this project is to be able to identify when a user has entered a fraudulent cloned website over an authentic original website. The application can do this by performing a number of various tests on the data about the website and the data that can be found within the website. The results of these tests are then used to calculate the likelihood of a website being a fraudulent clone.

Usability

For an individual to use this application that will be produced as part of this project, the user must install a Chrome browser extension and an application that will reside on the user's machine. The browser extension will identify when the user has reached a website. The information regarding the website will be sent to the underlying application where the tests will be performed. The browser extension will then display the verdict of authenticity that the application has come to.

Reliability

Given the wide range of factors involved in identifying a fraudulent cloned website, the application will provide its verdict of authenticity, this will allow greater reliability within the application as one test will not completely screw the verdict and the other tests will provide more accurate feedback the user.

Performance

The application will require internet access in order to perform some tests such as checking for other domains that have the same entity within them, this will cause changes in the application performance as it relies on the user's internet connection in order to perform the tests. The application itself has been made with performance in mind as I am using lightweight and faster libraries where possible such as using Gazpacho library in order to perform HTML parsing instead of libraries such as BeautifulSoup which require more resources in order to conduct HTML parsing.

Supportability

This application will be supported by Windows operating system and the browser extension will be supported by the browser Google Chrome. The user has to install the browser extension through Google Chrome and then need to separately install the underlying application that must communicate with the browser extension. The application and browser extension will be run locally on the user's device.

Metrics

I would consider this project successful if the following has occurred:

- The application can successfully categorise fraudulent cloned website.
- The application and the browser extension are able to communicate information between each other.
- Information between the application and the browser extension are transferred securely.
- The browser extension is able to successfully display information to the user.
- All project deadlines have been met.