PHISH ALERT

Katie Doyle

C00240662

Christopher Staff

Institute of Technology, Carlow

17 December 2021

DOCUMENT HISTORY

Date	Version	Description	Document Author
08/12/21	1.0	Initial Revision	Katie Doyle
23/04/22	1.0	Revision	Katie Doyle

Table of Contents

1 Introduction

A prospective Phishing Detection tool with a coherent level of accuracy in determining the legitimacy of a received Electronic Mail message is welcomed in the present-day battle against Phishing attempts. With this in mind, it is my intention to fulfil this consumer need with an innovative approach to differentiate my prospective Phishing Detection tool from the ones currently in production. The tool will be presented as an Email Client extension which will instinctively classify Electronic Mail messages in real-time as they enter the recipients inbox. The tool will act as a standalone intermediary between the consumer and a Phishing attempt.

1.1 Purpose of the document

A functional specification is a document used to describe a products intended capabilities, appearance, and interactions with users in detail for software developers. The functional specification is a guideline and continuing reference point for the developers (Search Software Quality, 2021).

1.2 Project Scope

To produce a standalone Phishing Detection tool presented as an Email Client extension which will accurately identify and classify the presence of Phishing with the ability to provide user feedback and possess the ability to relocate the intended mail to a dedicated folder for Phishing if necessary.

1.3 Scope of the document

To identify the prospective systems capabilities, appearance, and interactions with system users. The document will act as a guideline and continuing reference point throughout the development phase.

1.4 Related documents

Component	Name (with link to document)	Description
Research Manual	Phish Alert: https://bit.ly/307K4EZ	Phish Alert Research Manual

1.5 Terms/Acronyms and Definitions

Term/Acronym	Definition	Description
Email	Electronic Mail	Information stored on a computer that is exchanged between two users over telecommunications (Computer Hope, 2021).
IDE	Integrated Development Environment	An Integrated Development Environment is software for building applications that

		combines common developer tools into a single graphical user interface (Red Hat, 2019).
CRUD	Create, Read, Update, Delete	CRUD is an acronym that refers to the four functions that are considered necessary to implement a persistent storage application: create, read, update and delete (Sumo Logic, 2021).
API	Application Programming Interface	API is the acronym for Application Programming Interface, which is a software intermediary that allows two applications to talk to each other (Mule Soft, 2021).

1.6 Risks and Assumptions

A number of risks have been identified which could affect the functional design of the system. These include the risk of system penetration, creation of a similar tool for malicious intent, reconnaissance on the underlying algorithm arising a system bypass, and an email being sent to the end-user containing unexpected input to intentionally change the state and/or condition of the system.

2 System / Solution Overview

The application, as an Email Client extension, has unimpeded access to the consumers incoming mail to begin Phishing Detection analysis. The analysis conducted consists of several proven methods of determining the nature of an Electronic Mail message, such as extracting any URLs present in the mail and extracting Header information. By analysing these two divisions the system will be able to derive an accurate classification of the nature of the email.

After successfully and accurately classifying an Electronic Mail message, the system will then return feedback to the consumer which will detail the determined interpretation of the message. If the feedback indicates a message indicative of Phishing, the original email will be moved out of the recipients inbox and into a dedicated Phishing folder to further protect the consumer from accidental exposure to hazard.

Acting as an intermediary between a consumer and a Phishing attempt, the prospective system will prevent successful Phishing attempts delivered through Electronic Mail.

2.1 Context / Misuse Case Diagram

The Context Diagram shown defines and clarifies the boundaries of the proposed system. It identifies the flows of information between my system and external entities. The system is shown as a single process (University of Cape Town, 2011).

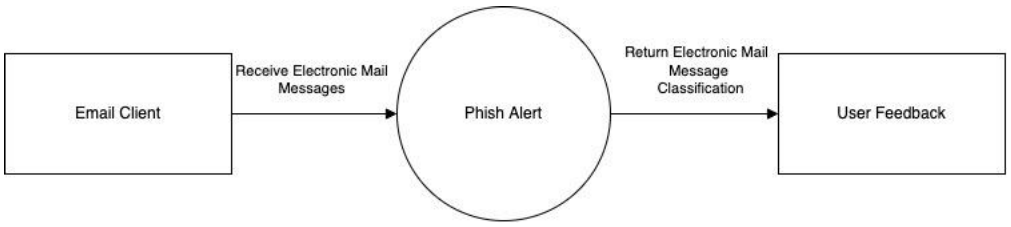
Email Client
Receive Electronic Mail Messages
Phish Alert
Return Electronic Mail Message Classification
User Feedback

Figure 1: Context Diagram

The Misuse Case Diagram shown is combined with a corresponding Use Case Diagram. It shows the systems operations and the associated actors. The inclusion of the Misuse Diagram expands the use cases by adding malicious functions that might affect the system (ToolBox, 2021).

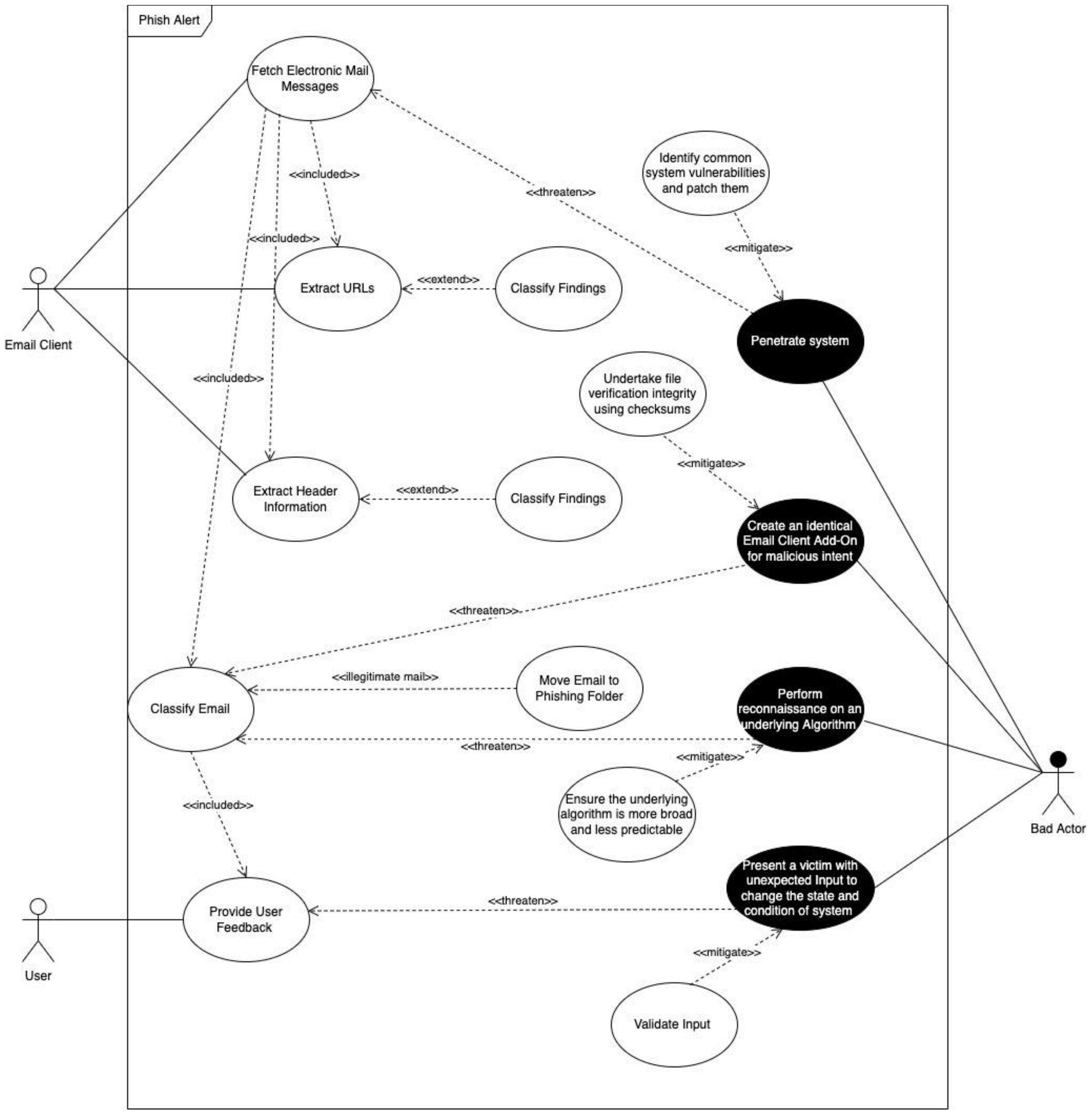
Phish Alert
Fetch Electronic Mail Messages
<<included>>
<<included>>
<<included>>
<<threaten>>
Identify common system vulnerabilities and patch them
<<mitigate>>
Email Client
Extract URLs
<<extend>>
Classify Findings
Penetrate system
Undertake file verification integrity using checksums
<<mitigate>>
Extract Header Information
<<extend>>
Classify Findings
Create an identical Email Client Add-On for malicious intent
<<threaten>>
<<illegitimate mail>>
Move Email to Phishing Folder
Classify Email
Perform reconnaissance on an underlying Algorithm
<<threaten>>
<<mitigate>>
Ensure the underlying algorithm is more broad and less predictable
<<included>>
Bad Actor
Present a victim with unexpected Input to change the state and condition of system
<<threaten>>
Provide User Feedback
User
<<mitigate>>
Validate Input

Figure 2: Misuse Case Diagram

2.2 System Actors

2.2.1 User Roles and Responsibilities / Authority Requirements

User/Role	Example	Frequency of Use	Security/Access, Features Used	Additional Notes
Email Client	Provides email metadata	Frequent	Extract URLs, Extract Header Information, Classify Email	Not Applicable
User	Receives feedback from the system	Occasional	Receive user feedback	Not Applicable
Bad Actor	Poses a threat to the system	Rare	Abuses features	Not Applicable

3 Functional Specifications

A number of specifications relate to the overall prospective system. They can be classified under two headings, Core and Non-Core, as shown below.

Name	Classification
Fetch Electronic Mail Messages	Core
Extract URLs	Core
Extract Header Information	Core
Classify Email	Core
Static or Dynamic (Inbox)	Core
Provide User Feedback	Core

Due to the nature of the prospective system, no Non-Core specifications have yet been identified.

3.1 Fetch Electronic Mail Messages

3.1.1 Purpose / Description

This specification provides the email metadata needed to perform the detection techniques on. This specification is first required to enable the use of all other core specifications.

3.1.2 Use Case

UC-1	Fetch Electronic Mail Messages
Primary Actor(s)	Email Client
Stakeholders and Interest	User and Bad Actor
Trigger	Incoming mail
Pre-conditions	Email is received
Post-conditions	Extract URLs, Extract Header Information, Classify Email, Provide User Feedback
Main Success Scenario	Incoming mail is detected and received
Extensions	Not Applicable
Priority	High

Special Requirements	Not Applicable
Open Questions	Not Applicable

3.1.3 Functional Requirements

Email Client extension which will be triggered upon receiving incoming mail.

3.2 Extract URLs

3.2.1 Purpose / Description

This specification will extract any URLs present in the email presented for analysis by the predecessor.

3.2.2 Use Case

UC-2	**Extract URLs**
Primary Actor(s)	Email Client
Stakeholders and Interest	User and Bad Actor
Trigger	Received mail
Pre-conditions	Email is fetched
Post-conditions	Extract Header Information, Classify Email, Provide User Feedback
Main Success Scenario	Presented mail is analysed and any URLs present are extracted
Extensions	Not Applicable
Priority	High
Special Requirements	Not Applicable
Open Questions	Not Applicable

3.2.3 Functional Requirements

Email is fetched by the system and provided for analysis.

3.3 Extract Header Information

3.3.1 Purpose / Description

This specification will extract any Header Information present in the email message source supplied for analysis by the predecessor.

3.3.2 Use Case

UC-3	**Extract Header Information**
Primary Actor(s)	Email Client
Stakeholders and Interest	User and Bad Actor
Trigger	Received mail
Pre-conditions	Email is fetched
Post-conditions	Classify Email, Provide User Feedback

Main Success Scenario	Presented mail is analysed and any Header Information present is extracted
Extensions	Not Applicable
Priority	High
Special Requirements	Not Applicable
Open Questions	Not Applicable

3.3.3 Functional Requirements

Email is fetched by the system and provided for analysis.

3.4 Classify Email

3.4.1 Purpose / Description

This specification will make a classification based on the results on the URL extraction and Header Information extraction specifications. This classification will determine whether the email will remain static in the recipients inbox or be moved and determine the feedback presented to the user.

3.4.2 Use Case

UC-4	Classify Email
Primary Actor(s)	Email Client
Stakeholders and Interest	User and Bad Actor
Trigger	Received mail
Pre-conditions	Email is fetched
Post-conditions	Provide User Feedback
Main Success Scenario	An accurate classification is made based on the results of the Extract URL and Extract Header Information specifications
Extensions	Not Applicable
Priority	High
Special Requirements	Not Applicable
Open Questions	Not Applicable

3.4.3 Functional Requirements

Email that was fetched at the beginning of the system flow has now been analysed by the previous use cases, determining an appropriate use case.

3.5 Static or Dynamic (Inbox)

3.5.1 Purpose / Description

This specification will act on the classification determined in the predecessor. If the classification is indicative of Phishing the email determined to be hazardous will be relocated to a designated folder for Phishing, to further protect the end-user.

3.5.2 Use Case

UC-5	Static or Dynamic (Inbox)
Primary Actor(s)	Email Client
Stakeholders and Interest	User and Bad Actor
Trigger	Email classification
Pre-conditions	Email is classified
Post-conditions	Email remains static, email is relocated
Main Success Scenario	A classification indicative of Phishing results in the intended email being relocated to a designated folder for Phishing
Extensions	Not Applicable
Priority	High
Special Requirements	Not Applicable
Open Questions	Not Applicable

3.5.3 Functional Requirements

The classification determined by the predecessor is now used to determine if it is necessary to relocate an email to a dedicated Phishing folder.

3.6 Provide User Feedback

3.6.1 Purpose / Description

This specification will act on the classification determined in the predecessor. If the classification is indicative of Phishing the email determined to be hazardous will be relocated to a designated folder for Phishing, to further protect the end-user.

3.6.2 Use Case

UC-6	Provide User Feedback
Primary Actor(s)	User
Stakeholders and Interest	Email Client and Bad Actor
Trigger	Email classification
Pre-conditions	Email is classified
Post-conditions	Not Applicable
Main Success Scenario	Detailed, accurate feedback is presented to the end-user, indicating the intentions of a received email
Extensions	Not Applicable
Priority	Medium
Special Requirements	Not Applicable
Open Questions	Not Applicable

3.6.3 Functional Requirements

The classification determined by the predecessor is now elaborated on for the purposes of the end-user.

4 System Configurations

An overview of all the steps required to configure the prospective system.

Step	Description
1.	Configure in an IDE an executable which has the ability to perform basic CRUD operations on an email using the Java Mail API.
2.	Once mail manipulation is possible in the executable create the functions which will deal with URL extraction and Header Information extraction. This can be done using built-in libraries such as JNDI.
3.	Taking into account the results of the URL extraction and Header Information extraction functions, make an informed classification of the nature of the email.
4.	Once a basic implementation exists, introduce the use of a specific dataset containing mass amounts of Phishing and Non-Phishing emails to gauge the accuracy of the system.
5.	Undertake any systematic reviews depending on the results of the predecessor.
6.	Once a satisfactory accuracy level is obtained begin the creation of the Email Client extension which will replace the need for the dataset and Java Mail API.
7.	Email Client extension is now instinctively classifying emails in real-time at a high level of accuracy.
8.	Introduce a function which will take the results of the classification and if it is indicative of Phishing, relocate the email in question to a designated Phishing folder.
9.	Introduce the User Feedback function which will detail the findings to the end-user.

5 Other System Requirements / Non-Functional Requirements

5.1 Non-Functional Requirements

Name	Description
Availability	System is available to work as required when it is required.
Reliability	System will perform the task(s) it was designed or intended to do.
Performance	System will perform tasks in a fashion that complies with predetermined criteria.
Security	System will protect all data manipulated internally from unauthorized access and threats.
Scalability	System will appropriately handle increasing and decreasing workloads.
Usability	System is easy to configure and is efficient in carrying out user tasks.
Maintainability	System is can be easily supported, changed, enhanced or restructured over time.

5.2 Metrics

Key metrics for this proposed system will heavily depend on the dataset or repository of Phishing emails used from the beginning to gauge the accuracy of the system.

In order to gauge if this system is successful in detecting Phishing attempts, the level of accuracy will be determined based on the number of Phishing and Non-Phishing emails present in the dataset, and the number of positive and negative classifications derived from the system. The closer the accuracy reading is to one hundred percent, the more successful the system will be.

These figures will be supplied with the final report.

The metrics to be used to gauge the success of the security of the system is as follows:

- The system is not vulnerable to penetration,
- A checksum is provided upon installation to verify file integrity to rule out an installation of malicious intent,
- The underlying algorithm is not vulnerable to reconnaissance attempts,
- Input is validated.

5.3 Precedent

A similar implementation of these deliverables has been conducted before, basing the underlying technology on an ensemble of techniques, including the use of Machine Learning algorithms such as Bagging, AdaBoost, Random Forest and Gradient boosting.

The difference between this previous implementation and my proposed system is the underlying techniques used or to be used. A Machine Learning system was implemented in the precedent, and I am proposing a system Conventionally Programmed where I will be manually formulating the rules and logic.

The precedent produced high variations in accuracy ratings between each algorithm used, although all algorithms were trained from the same dataset. The highest accuracy documented is 96.15% and lowest being 54.31%. The dataset used contained 11,000 malicious URL's (National College of Ireland, 2020).

I intend to respond to the various limitations associated with the precedent with a Conventionally Programmed implementation which will allow me to formulate an algorithm to deal with a generic image of Phishing, improving the accuracy figures mentioned above, and subsequently allowing me to be innovative due to the lack of published papers introducing a system of this nature.

5.4 External Tools / Libraries

The following external tools/systems are required for a successful implementation of the deliverables expected:

- Java Mail API – allows for mail manipulation using basic CRUD (create, read, update, delete) operations which will assist in the initial development stage.
- Java Naming and Directory Interface (JNDI) – allows for requesting functions such as DNS (Domain Name Service) which will assist the URL Extraction function to classify an email.
- Eclipse IDE – an integrated development environment application which facilitates software developing.
- Gmail – a free Email Client which will host the prospective Extension.
- Phish Tank – an online repository of Phishing and Non-Phishing emails. AN API is also available for developers to access the dataset from an application.
- Google Apps Script – online JavaScript Platform used in the development of Gmail Extensions.

6 Project Plan

6.1 Hardware / Software Requirements

For the successful implementation of this project, the following Software and Hardware requirements must be met:

Software
MacOS Operating System
Eclipse IDE
Java Mail API
Java Naming and Directory Interface (JNDI)
Google Apps Script
Google Chrome
Gmail (Web)

Hardware
Apple Device Hardware
RAM: At least 128 MB
Disk Space: 124 MB for JRE, 2 MB for Java Update
Processor: Minimum Pentium 2 266 MHz processor
Modem

6.2 Project Milestones

Major project milestones are shown below in tabular form and documented in a Gantt chart in chronological order.

Task	**Start Date**	**End Date**	**Duration (Days)**
Research	04-Oct	26-Nov	53
Functional Specification	26-Nov	17-Dec	21
Begin Implementation	17-Dec	28-Mar	101
Fetch Email Function	17-Dec	31-Dec	14
Extract URL Function	03-Jan	20-Jan	17
Extract Header Information Function	21-Jan	04-Feb	14
Classify Email Function	05-Feb	20-Feb	15
User Feedback Function	21-Feb	07-Mar	14
Move Email Function	08-Mar	17-Mar	9
Create Email Client Extension	18-Mar	28-Mar	10
System Security Review	28-Mar	31-Mar	3
Final Changes	01-Apr	04-Apr	3

Project Report	05-Apr	12-Apr	7
Project Demo	13-Apr	30-Apr	17

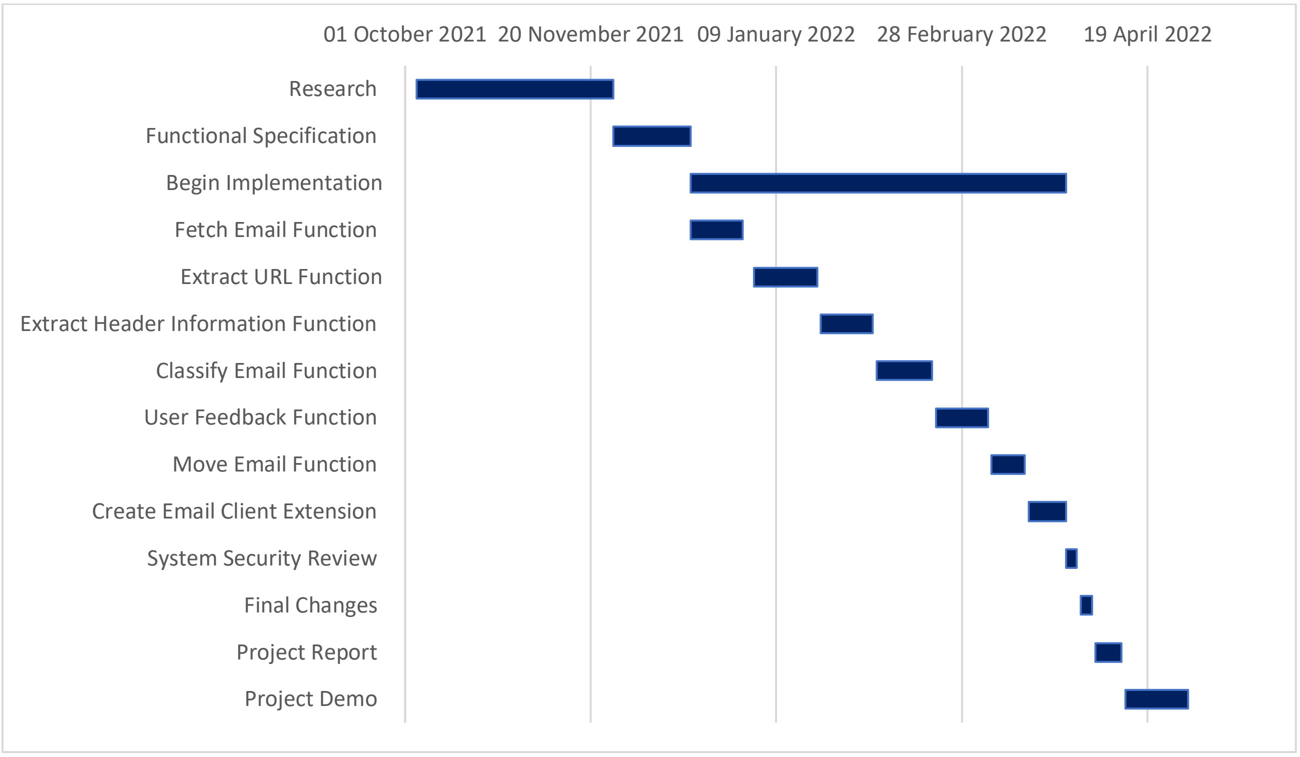

Figure 3: Gantt Chart

7 Bibliography

Computer Hope (2021) *Email*. Available at: https://www.computerhope.com/jargon/e/email.htm#:~:text=Short%20for%20electronic%20mail%2C%20e,individual%20or%20group%20of%20individuals. (Accessed: 8 December 2021)

Mule Soft (2021) *What is an API?* Available at: https://www.mulesoft.com/resources/api/what-is-an-api
(Accessed: 8 December 2021)

National College of Ireland (2020) *Detection of Phishing URL using Ensemble Learning Techniques.* Available at: http://norma.ncirl.ie/4509/1/sharadrajendraparmar.pdf
(Accessed: 25 October 2021)

Red Hat (2019) *What is an IDE?* Available at: https://www.redhat.com/en/topics/middleware/what-is-ide#:~:text=An%20integrated%20development%20environment%20(IDE,graphical%20user%20interface%20(GUI).(Accessed: 8 December 2021)

Sumo Logic (2021) *DevOps and Security Glossary Terms*. Available at: https://www.academia.edu/38006035/Functional_Specification_Document_Template
(Accessed: 8 December 2021)

ToolBox (2021) *Using UML Misuse Diagrams to Secure Systems From Threat Actors*. Available at: https://www.toolbox.com/tech/it-strategy/articles/using-misuse-case-diagrams/(Accessed: 7 December 2021)

University of Cape Town (2011) *Chapter 6. Data-Flow Diagrams*. Available at: https://www.cs.uct.ac.za/mit_notes/software/htmls/ch06s06.html
(Accessed: 7 December 2021)