** **

**A NOVEL PHISHING WEBSITE PREDICTION MODEL WITH CROWDSOURCING**

**A PROJECT REPORT**

***Submitted by***

**BALAMURUGAN.P (820617104008)**

**HARISH.A (820617104017)**

**PERINBAPERUMAL.S.A (820617104024)**

***in partial fulfillment for the award of the degree***

***Of***

**BACHELOR OF ENGINEERING**

**IN**

**COMPUTER SCIENCE & ENGINEERING**

**ARASU ENGINEERING COLLEGE, KUMBAKONAM**

**ANNA UNIVERSITY: CHENNAI 600 025**

**APRIL 2021**

**DECLARATION**

We hereby declare that the Project work entitled **“A NOVEL PHISHING WEBSITE PREDICTION MODEL WITH CROWDSOURCING”** is submitted in partial fulfillment of requirement for the award of the degree in B.E., Anna University Chennai, is a record of our own work carried out by us during the academic year 2020-2021 under the supervision and guidance of Dr. Kalaimani Shanmugam M.Tech., Ph.D., Professor and Head, Department in Computer Science and Engineering, Arasu Engineering College , Kumbakonam. The extent and source of information are derived from the existing literature and have been indicatsed through the dissertation at appropriate places. The matter embodied in this work is original and has not been submitted for the award of any other degree, either in this or any other university.

**Name Register number Signature**

Balamurugan.P 820617104008

Harish.A 820617104017

Perinbaperumal.S.A 820617104024

I certify that the declaration made above by the candidate is true.

Signature of the Supervisor

**Dr. Kalaimani Shanmugam M. Tech., Ph.D.,**

Professor and Head, Department of CSE

**ANNA UNIVERSITY : CHENNAI 600 025**

**BONAFIDE CERTIFICATE**

Certified that this project report **“A NOVEL PHISHING WEBSITE PREDICTION MODEL WITH CROWDSOURCING”** is the bonafide work of **“BALAMURUGAN.P (820617104008), HARISH.A (820617104017), PERINBAPERUMAL.S.A (820617104024) ”** who carried out the project work under my supervision.

**SIGNATURE**   **SIGNATURE**

Dr. KALAIMANI SHANMUGAM M. Tech., Dr. KALAIMANI SHANMUGAM M. Tech.,

Ph.D., Ph.D.,

**PROFESSOR AND HEAD**   **SUPERVISOR**

Department of Professor and Head

Computer Science and Engineering, Department of

Arasu Engineering College, Computer Science and Engineering,

Kumbakonam-612501 Arasu Engineering College,

Kumbakonam-612501.

**Submitted for Project Viva-Voce examination held on: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**INTERNAL EXAMINER EXTERNAL EXAMINER**

**ACKNOWLEDGEMENT**

Apart from the efforts of us, the success of our project depends largely on the encouragement and guidelines of many others. It is our pleasure to give our special thanks to **ANNA UNVIERSITY**.

Also we thank to **ARASU ENGINEERING COLLEGE** especially to our Chairman **Mr. R. THIRUNAVUKARASU, M. A.,** and other management people for permitted us to carry out this project work.

We like to express our deep gratitude to our beloved Principal,

**Dr. T. BALAMURUGAN M.E., Ph.D.,** for providing us all necessary facilities to carry out our project successfully.

Our profound sense of gratitude is due to our Head of Computer Science & Engineering department, **Dr. KALAIMANI SHANMUGAM M. Tech., Ph.D.,** for her constant encouragement and valuable guidance.

Our heartfelt gratitude to our project coordinator **Mr. M. ANANDA KUMAR M.E.,** Assistant Professor of CSE department who continuously supported us in every possible way, from initial advice to encouragement till now.

We would like to show my greatest appreciation to our project guide **Dr. KALAIMANI SHANMUGAM M. Tech., Ph.D.,** Professor and Head of CSE department. We feel motivated and encouraged every time when we attended his meeting. Without his encouragement and guidance this project would not have materialized.

The guidance and support received from all the **Faculty members** who contributed to this project, was vital for the success of the project. We are grateful for their constant support and help.

Last but not least my special thanks to the all the supporting **staffs and lab technicians** at the institute who directly or indirectly helped us in completing our project.

**ABSTRACT**

Nowadays the web and online services have revolutionized the modern world. Many people use browser to perform various activities like online shopping, online bill payment, online mobile recharge and online banking transaction. Due to the dependence on online services, the online security threats like cybercrime are also increasing rapidly. One of the most common online security threats is Phishing attack, which is used to mimic a legitimate website such as online banking, e-commerce or social networking website in order to obtain sensitive data such as user-names, passwords, financial and health-related information from potential victims. There are several anti-phishing approaches such as Blacklist or Whitelist, heuristic and visual similarity-based methods for phishing website detection are available, but the main drawbacks are poor accuracy and low adaptability to new phishing links. In this project, a novel phishing website prediction model is developed using a more powerful machine learning algorithm called Random Forest thereby helping the user to distinguish between Legitimate website and phishing website. In this proposed model, feature selection and crowdsourcing technique are employed to make the prediction in efficient manner. The proposed model is evaluated using Phishtank dataset and better accuracy is achieved**.**

.

**TABLE OF CONTENT**

|  |  |  |
| --- | --- | --- |
| **CHAPTER NO** | **TITLE** | **PAGE NO** |
|  | **ABSTRACT**  **LIST OF FIGURES** | V  ix |
| 1 | **INTRODUCTION** | 1 |
| 2 | **LITERATURE SURVEY** | 4 |
| 3 | **PROBLEM DEFINITIONS**  3.1 Existing System  3.2 Proposed System | 9  9  10 |
| 4 | **SYSTEM STUDY**  4.1 FEASIBILITY STUDY  4.1.1 Technical Feasibility  4.1.2 Operational Feasibility  4.1.3 Economical Feasibility | 12  12  13  13  13 |
| 5 | **SYSTEM REQUIREMENTS**  5.1 HARDWARE REQUIREMENTS  5.2 SOFTWARE REQUIREMENTS  5.3 SOFTWARE DESCRIPTION  5.3.1 Android  5.3.2 Python | 15  15  15  15  15  18 |
| 6 | **SYSTEM DESIGN**  6.1 SYSTEM ARCHITECTURE  6.2 DATA FLOW DIAGRAM  6.3 UML DIAGRAM  6.3.1 Use Case Diagram  6.3.2 Class Diagram  6.3.3 Sequence Diagram  6.3.4 Collaboration Diagram | 21  21  22  24  24  25  26  27 |
| 7 | **SYSTEM IMPLEMENTATION**  7.1 SYSTEM DESCRIPTION  7.2 MODULES  7.2.1 Accessibility module  7.2.2 Parsing module  7.2.3 API module  7.2.4 Predicting module  7.2.5 Floating widget module  7.2.6 Logging module  7.2.7 Crowdsourcing module | 28  28  28  28  28  29  29  29  29  30 |
| 8 | **SYSTEM TESTING**  8.1 Introduction    8.2 Testcases | 31  32  34 |
| 9 | **SAMPLE CODE** | 35 |
| 10 | **SCREEN SHOTS** | 71 |
| 11 | **CONCLUSION AND FUTURE ENHANCEMENT** | 75 |
| 12 | **REFERENCES** | 76 |

**LIST OF FIGURES**

|  |  |  |
| --- | --- | --- |
| **FIGURE NO** | **TITLE** | **PAGE NO** |
| 6.1 | SYSTEM ARCHITECTURE | 20 |
| 6.2 | DATA FLOW DIAGRAM | 21 |
| 6.3 | USE CASE DIAGRAM | 24 |
| 6.4 | CLASS DIAGRAM | 25 |
| 6.5 | SEQUENCE DIAGRAM | 26 |
| 6.6 | COLLABORATION DIAGRAM | 27 |
|  |  |  |

**CHAPTER 1**

# 1. INTRODUCTION

Nowadays the web and online services have revolutionized the modern world. Many people use browser to perform various activities like online shopping, online bill payment, online mobile recharge and online banking transaction. Due to the dependence on online services, the online security threats like cybercrime are also increasing rapidly. One of the most common online security threats is Phishing attack, which is used to mimic a legitimate website such as online banking, e-commerce or social networking website in order to obtain sensitive data such as user-names, passwords, financial and health-related information from potential victims. From industries such as healthcare and education to individuals playing games online, the impact of phishing attack is widely felt.

One of the real phishing attacks is Tirumala Tirupathi temple fake darshan ticket. Another example is phishing attack through online payment Paypal. The scammers could send out an attack email that instructs recipients to click on a link in order to rectify a discrepancy with their account. In actuality, the link redirects to a website designed to impersonate PayPal’s login page. That website collects login credentials from the victim when they try to authenticate themselves and sends that data to the attackers.

There are several anti-phishing approaches such as Blacklist or Whitelist, heuristic and visual similarity-based methods for phishing website. But the main drawbacks are poor accuracy and low adaptability to new phishing links. In this project, a novel Phishing website prediction model is developed using a more powerful machine learning algorithm called Random Forest thereby helping the user to distinguish between legitimate website and Phishing website.

**MOTIVATION**

## The motivations for this projects are the recent phishing attacks on temple reservation sites. Fraudsters created fake domains similar to the temple body’s official website to lure devotees and made them transfer money to get arjitha seva tickets. It’s only after the unsuspecting devotees reach Tirumala with the tokens that they realize that they have been taken for a ride. It is very essential to prevent these kinds of attacks in future.

# OVERVIEW

Phishing is a threat in a cyber-world. It has grown and evolved over the years. Phishers are getting creative in planning and executing the attacks. Users could not identity such malicious sites without any tool and lose precious information and money to the phishers. This solution can provide the user with the information about a website whether it is a legitimate website or not before losing any kind of information. Crowdsourcing techniques are used to identify recent phishing websites by the contribution of group of people who report any new unidentified phishing site. The mobile application will scan for all the URL used by the application and runs it on a Random forest model with future selection method within a fraction of the second to predict the URL information and warns the user if any malicious URL is detected.

* 1. **OBJECTIVE**

To provide a knowledge and insight to the inexperienced web users in identifying the phishing URLs. To propose the appropriate feature selection method with Phishtank dataset. To apply crowdsourcing method to include newly emerging phishing websites. To develop a machine learning model for efficiently predict the phishing website. To design and develop user friendly and secured Android based mobile application to automatically detect and block phishing web links.

.

# 

# 

**CHAPTER 2**

**LITERATURE SURVEY**

**MOHITH GOWDA et al., (2020)** to develop aPhishing is a technique under Social Engineering attacks which is most widely used to get user sensitive information, such as login credentials and credit and debit card information, etc. It is carried out by a person masquerading as an authentic individual. To protect web users from these attacks, various anti-phishing techniques are developed, but they fail to protect the user from these attacks in various ways. In this paper, we propose a novel technique to identify phishing websites effortlessly on the client side by proposing a novel browser architecture. In this system, we use the rule of extraction framework to extract the properties or features of a website using the URL only. This list consists of 30 different properties of a URL, which will later be used by the Random Forest Classification machine learning model to detect the authenticity of the website. A dataset consisting of 11,055 tuples is used to train the model. These processes are carried out on the client-side with the help of a redesigned browser architecture. Today Researches have come up with machine learning frameworks to detect phishing sites, but they are not in a state to be used by individuals having no technical knowledge. To make sure that these tools are accessible to every individual, we have improvised and introduced detection methods into the browser architecture named as ‘Embedded Phishing Detection Browser’ (EPDB), which is a novel method to preserve the existing user experience while improving the security. The newly designed browser architecture introduces a special segment to perform phishing detection operations in real-time. We have prototyped this technique to ensure maximum security, better accuracy of 99.36% in the identification of phishing websites in real time.

**RAGHAV KAULT et al., 2019)** presents An increasing number of services, including banking and social networking, are being integrated with world wide web in recent years. The crux of this increasing dependence on the internet is the rise of different kinds of cyberattacks on unsuspecting users. One such attack is phishing, which aims at stealing user information via deceptive websites. The primary defense against phishing consists of maintaining a black list of the phishing URLs. However, a black list approach is reactive and cannot defend against new phishing websites. For this reason, a number of research have been done on using machine learning techniques to detect previously unseen phishing URLs. While they show promising results, any such implementation is yet to be seen. This is because 1) little work has been done on developing a complete end-to-end framework for phishing URL detection 2) it is prohibitively slow to detect phishing URLs using machine learning algorithms. In this work we address these two issues by formulating a robust framework for fast and automated detection of phishing URLs. We have validated our framework with a real dataset achieving 87% accuracy in a real-time setup.

**ISHIKESH MAHAJAN et al., (2020)** presents Phishing attack is a simplest way to obtain sensitive information from innocent users. Aim of the phishers is to acquire critical information like username, password and bank account details. Cyber security persons are now looking for trustworthy and steady detection techniques for phishing websites detection. This paper deals with machine learning technology for detection of phishing URLs by extracting and analyzing various features of legitimate and phishing URLs. Decision Tree, random forest and Support vector machine algorithms are used to detect phishing websites. Aim of the paper is to detect phishing URLs as well as narrow down to best machine learning algorithm by comparing accuracy rate, false positive and false negative rate of each algorithm.

**A. MAHALAKSHMI et al., (2020)** to present Phishing is a deceitful attempt for obtaining the sensitive information like credit card details, user names and passwords. It is one of the social engineering methods that gathers personal information through websites such as malicious websites and deceptive e-mail to canvass personal information from a company or an individual by prance as a trustworthy entity or organization. Phishing often attacks email by using as a vehicle and even sending messages by email to users that represent a part of a company or an institution who perform business such as financial institution, banking etc. Phishing is becoming more malicious day by day and its detection is very important. In cyberspace, phishing is motivating the researchers to develop the model through which we can develop more security towards the safe services provided by the web. Here we discuss types of phishing and conflicts due to it.

**YAZAN AHMAD ALSARIERA et al., (2019)** The damaging of phishing is traumatizing as attackers or hackers execute theft of sensitive information from users subtly for inappropriate or unauthorized usage. In the light of curbing phishing, blacklisting of websites proved as the deployment of phishing websites are rampantly increasing and often short-lived. Hence, machine learning (ML) methods are seen as viable measures and used to develop deplorable models that can detect a phishing website. ML methods are fast gaining attention and acceptance in detecting phishing websites as they can cope with the dynamism of phishing websites and attackers. However, ML methods still some shortcomings in terms of low detection accuracy, high false alarm rate (FAR) and induced bias of developed ML solutions. In addition, with the evolving nature of phishing attacks, there is a continuing imperative need for novel and ML-based methods for detecting phishing websites. This study proposed 3 meta-learner models based on Forest Penalizing Attributes (ForestPA) algorithm. ForestPA uses a weight assignment and weight increment strategy to build highly decision trees by exploiting the prowess of all attributes (non-class inclusive) in a given dataset. From the experimental results, the proposed meta-learners (ForestPA-PWDM, Bagged-ForestPAPWDM, and Adab-ForestPA-PWDM) are highly with the least accuracy of 96.26%, 0.004 FAR, and 0.994 ROC value. Further, with the superiority of the proposed models over other existing methods, we recommend the development and adoption of meta-learners based on ForestPA for phishing website detection and other cyber-security attacks.

It is observed from the literature survey that phishing websites detection systems have used selective sampling, support vector machine, decision tree algorithm, classification, regression tree and embedded phishing detection browser. The most recent technology today which works better for phishing websites prediction system is support vector machine algorithm, which is an extension of the Machine learning algorithm. Hence in this research, irrespective of the being old model, the Random forest algorithm is studied in depth and applied to phishing websites prediction. The Random forest algorithm is a supervised Machine learning algorithm. It is well established and well proven algorithm for feature selection and crowd sourcing techniques.

**CHAPTER 3**

**PROBLEM DEFINITION**

**3.1 PROBLEM STATEMENT**

With the rise of the Internet as a major mode for economic transactions and communication, phishing attackers make web users believe that they are communicating with the trusted entity for the purpose of stealing account information and login details. Phishing detection techniques do suffer low detection accuracy and the user cannot spend time to search the information about the websites. This application is created to automatically detect malicious websites and prevent the user from interacting with it.

**3.2 EXISTING SYSTEM**

Phishing websites detection has a main focus on URL detection and URL feature extraction using feature selection method. A URL feature is classified into four groups such as Address-bar based features, Abnormal features, HTML and JavaScript based features and Domain based features. The Phishtank dataset provides the collection of data needed by the feature extraction model to extract the features. Features are extracted through collaborative approach and large number of features are analyzed to detect the phishing website. Random Forest will be used to determine whether a URL is a phished or legitimate one. Once the URL is fed to the system, the system extracts features such as number of visitors, number of pages hosted by them. These features give a brief overview on category of URL thereby increasing the response time of the system. A simple interface to access parsed Whois data for a given URL’s domain, provides the ability to extract data for all the popular TLD’s by querying a Whois server directly instead of going through an intermediate web service. The Crowdsourcing technique helps to use features that are extracted through collaborative approach allowing much larger number of features to be analyzed to detect recent phishing websites.

**DRAWBACKS**

* Can’t be deployed as application.
* Poor accuracy.
* Needs more computational power.
* Although, it protects user from entering into phishing sites on Chrome Browser, it fails to do so on other browsers.
* It cannot detect newly created phishing sites (Someone has to report it).
* Doesn’t work in other mobile browsers.
* Doesn’t scan all the links opened by the device (Mobile).
* Cannot intercept all network traffic like GET requests.

**3.3 PROPOSED SYSTEM**

The proposed phishing websites detection system helps to detect any phishing website from all over the device’s traffic to provide a more robust and reliable security against all kinds of phishing attacks. The proposed system will give exact information to the user about whether a website is legitimate or not potentially preventing the user from interacting with any kind of phishing site. The proposed system offers very minimal impact on the performance since all the prediction happens on a server thus providing high speed detection along with minimal performance impact. The proposed system uses Crowdsourcing technique to detect any newly created phishing sites thus providing complete protection against phishing attacks.

**FEATURES OF PROPOSED SYSTEM:**

1. **FUNCTIONAL CAPABILITIES:**    The ultimate aim of this project is to detect phishing attacks in real-time. The app in the client device keeps on checking with machine learning server for any maliciousness in the accessed site/app.
2. **PERFORMANCE LEVEL:**   The current Machine learning server is setup on 1GB RAM Virtual Private Server. It gives the response within 500-1000ms. At the client side, it takes 1-2 seconds to detect whether a site is phishing or not.
3. **DATA STRUCTURES:**  The data in this project are maintained in the CSV form. It provides easy access to the user.
4. **SAFETY:**No data loss occurs in this system.
5. **RELIABILITY:** It is assured that the project is completely authenticated in order to enhance security and corruptions of database as well as the software. The person is given access only if he/she has a valid username and password.

**CHAPTER 4**

**SYSTEM STUDY**

**4.1 FEASIBILITY STUDY**

This project is feasible provided given unlimited resources and infinite time. Unfortunately, the development of a computer-based system is more likely to be plagued by resource scarcity and stringent schedules. It is both necessary and prudent to evaluate the feasibility of a project at earliest possible time. Wastage of manpower and financial resources and untold professional embarrassment can be avoided if an ill-conceived system is recognized early in the development phase. So, a detailed study was carried out to check the workability of the proposed system. Feasibility study is a test of system proposal regarding its workability, impact on the organization, ability to meet user needs and effective use of resources. Thus, when an application is proposed, it is normally going through a feasibility study before it is approved for development.

Feasibility and risk analysis are related in many ways. If project risk is great, the feasibility of producing quality is reduced. Thus, during feasibility analysis for this project, following three primary areas for interest was considered very carefully. There are several types of feasibility. Three key consideration involved in the feasibility analysis are,

1) Technical Feasibility

2) Operational Feasibility

3) Economic Feasibility

**4.1.1 TECHNICAL FEASIBILITY**

This study is carried out to check the technical feasibility; this is the technical requirements of the system. Any system developed must not have a importance to consider the monetary factors also it might happen that developing a particular system may be technically possible. This will lead to high demands being placed on the client. The developed system must have modest requirements, as only minimal or null changes are required for implementing this system.

In this technical feasibility the following issues are taken into considerations. Once the technical feasibility is established, it is important to consider the monetary factors also. Since it might happen that developing a particular system may be technically possible but it may require huge investment and benefits may be less. For evaluating this, economic feasibility of the proposed system is carried out.

**4.1.2 OPERATIONAL FEASIBILITY**

The proposed system normally solves the problem and takes advantages of the opportunities identified during scope definitions; it satisfies the requirements identified in the requirement analysis phase of system development. Since the statistical figures are stored in a certain format in the computer, reduce the manual work and enhance the standard of presentation also.

Nonoperational feasibility assesses the extent to which the required software performs a series of steps to solve business problems and user requirements. This measures how well your company will be able to solve problems and take advantage of opportunities that are presented during the course of the project.

**4.1.3 ECONOMIC FEASIBILITY**

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

In economic feasibility, cost benefits analysis is done in which expected cost and benefits are evaluated. Economic analysis is used for evaluating the effectiveness of the proposed system. The developed system is economical when compared to the existing system job done manual system. So the proposed system is so fast that planning can be made easily.

**CHAPTER 5**

**SYSTEM REQUIREMENTS AND SPECIFICATION**

**5.1 HARDWARE REQUIREMENTS**

Processor : Intel i3 or higher

RAM : 8GB

**5.2 SOFTWARE REQUIREMENTS**

Operating system : Android Marshmallow (API 23) or above

Font-End : Android

Back-End : Flask API

Framework : Flask, Android

Language : Java, Python, XML

Server : Heroku (Free plan)

**5.3 SOFTWARE DESCRIPTION**

**5.3.1 Android**

**Platform Architecture**

Android is an open source, Linux-based software stack created for a wide array of devices and form factors. The following diagram shows the major components of the Android platform.

**The Linux Kernel**

The foundation of the Android platform is the Linux kernel. For example, the Android Runtime (ART) relies on the Linux kernel for underlying functionalities such as threading and low-level memory management.

Using a Linux kernel allows Android to take advantage of key security features and allows device manufacturers to develop hardware drivers for a well-known kernel.

**Hardware Abstraction Layer (HAL)**

The hardware abstraction layer (HAL) provides standard interfaces that expose device hardware capabilities to the higher-level Java API framework. The HAL consists of multiple library modules, each of which implements an interface for a specific type of hardware component, such as the camera or Bluetooth module. When a framework API makes a call to access device hardware, the Android system loads the library module for that hardware component.

**Android Runtime**

For devices running Android version 5.0 (API level 21) or higher, each app runs in its own process and with its own instance of the Android Runtime (ART). ART is written to run multiple virtual machines on low-memory devices by executing DEX files, a bytecode format designed especially for Android that's optimized for minimal memory footprint. Build tools, such as d8, compile Java sources into DEX bytecode, which can run on the Android platform.

Some of the major features of ART include the following:

* Ahead-of-time (AOT) and just-in-time (JIT) compilation
* Optimized garbage collection (GC)

On Android 9 (API level 28) and higher, conversion of an app package's Dalvik Executable format (DEX) files to more compact machine code.

Better debugging support, including a dedicated sampling profiler, detailed diagnostic exceptions and crash reporting, and the ability to set watchpoints to monitor specific fields

Prior to Android version 5.0 (API level 21), Dalvik was the Android runtime. If your app runs well on ART, then it should work on Dalvik as well, but the reverse may not be true.

Android also includes a set of core runtime libraries that provide most of the functionality of the Java programming language, including some Java 8 language features, that the Java API framework uses.

**Native C/C++ Libraries**

Many core Android system components and services, such as ART and HAL, are built from native code that require native libraries written in C and C++. The Android platform provides Java framework APIs to expose the functionality of some of these native libraries to apps. For example, you can access OpenGL ES through the Android framework’s Java OpenGL API to add support for drawing and manipulating 2D and 3D graphics in your app.

If you are developing an app that requires C or C++ code, you can use the Android NDK to access some of these native platform libraries directly from your native code.

**Java API Framework**

The entire feature-set of the Android OS is available to you through APIs written in the Java language. These APIs form the building blocks you need to create Android apps by simplifying the reuse of core, modular system components and services, which include the following:

* A rich and extensible View System you can use to build an app’s UI, including lists, grids, text boxes, buttons, and even an embeddable web browser.
* A Resource Manager, providing access to non-code resources such as localized strings, graphics, and layout files.
* A Notification Manager that enables all apps to display custom alerts in the status bar.
* An Activity Manager that manages the lifecycle of apps and provides a common navigation back stack.
* Content Providers that enable apps to access data from other apps, such as the Contacts app, or to share their own data.
* Developers have full access to the same framework APIs that Android system apps use.

**System Apps**

Android comes with a set of core apps for email, SMS messaging, calendars, internet browsing, contacts, and more. Apps included with the platform have no special status among the apps the user chooses to install. So a third-party app can become the user's default web browser, SMS messenger, or even the default keyboard (some exceptions apply, such as the system's Settings app).

The system apps function both as apps for users and to provide key capabilities that developers can access from their own app. For example, if your app would like to deliver an SMS message, you don't need to build that functionality yourself. You can instead invoke whichever SMS app is already installed to deliver a message to the recipient you specify.

**5.3.2 PYTHON**

* Python is a high-level, interpreted, interactive and object-oriented scripting Language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages.
* Python is Interpreted − Python is processed at runtime by the interpreter.

You do not need to compile your program before executing it. This is similar to PERL and PHP.

* Python is Interactive − You can actually sit at a Python prompt and interact

with the interpreter directly to write your programs.

* Python is object-Oriented − Python supports Object-Oriented style or

technique of programming that encapsulates code within objects.

* Python is a Beginner's Language − Python is a great language for the

beginner- level programmers and supports the development of a wide range of applications from simple text processing to WWW browsers to games.

**Assigning Values to Variables**

**Locating Modules**

* When you import a module, the Python interpreter searches for the module in the following sequences −
* The current directory.
* If the module isn't found, Python then searches each directory in the shell variable PYTHONPATH.
* If all else fails, Python checks the default path. On UNIX, this default path is normally /usr/local/lib/python/.
* Python variables do not need explicit declaration to reserve memory space. The declaration happens automatically when you assign a value to a variable. The equal sign (=) is used to assign values to variables.
* The operand to the left of the = operator is the name of the variable and the operand to the right of the = operator is the value stored in the variable.
* Python has been an object-oriented language since it existed. Because of this, creating and using classes and objects are downright easy. This chapter helps you become an expert in using Python's object-oriented programming support.
* If you do not have any previous experience with object-oriented (OO) programming, you may want to consult an introductory course on it or at least a tutorial of some sort so that you have a grasp of the basic concepts.
* However, here is small introduction of Object-Oriented Programming (OOP) to bring you at speed.

**Overview of OOP Terminology**

Class − A user-defined prototype for an object that defines a set of attributes that characterize any object of the class. The attributes are data members (class variables and instance variables) and methods, accessed via dot notation.

Class variable − A variable that is shared by all instances of a class. Class variables are defined within a class but outside any of the class's methods. Class variables are not used as frequently as instance variables are.

Data member − A class variable or instance variable that holds data associated with a class and its objects.

Function overloading − The assignment of more than one behavior to a particular function. The operation performed varies by the types of objects or arguments involved.

Instance variable − A variable that is defined inside a method and belongs only to the current instance of a class.

Inheritance − The transfer of the characteristics of a class to other classes that are derived from it.

Instance − An individual object of a certain class. An object obj that belongs to a class Circle, for example, is an instance of the class Circle.

Instantiation − The creation of an instance of a class.

Method  - A special kind of function that is defined in a class definition.

Object − A unique instance of a data structure that's defined by its class. An object comprises both data members (class variables and instance variables) and methods.

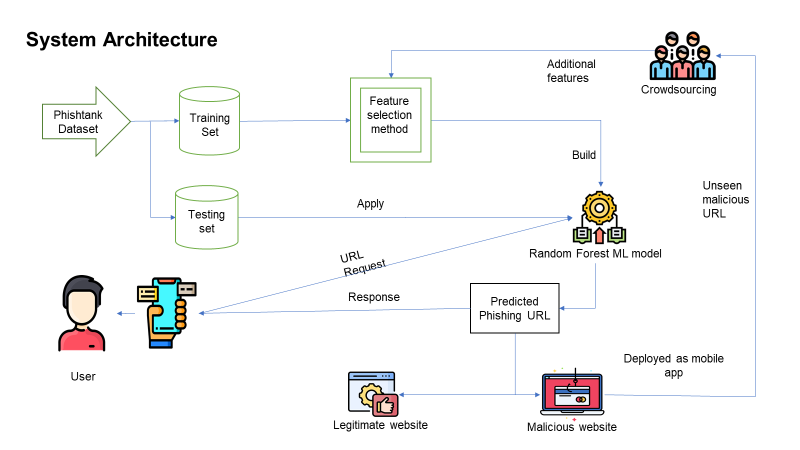
Operator overloading – The assignment of more than one function to a particular operator.

**CHAPTER 6**

# SYSTEM DESIGN

**6.1 SYSTEM ARCHITECTURE**

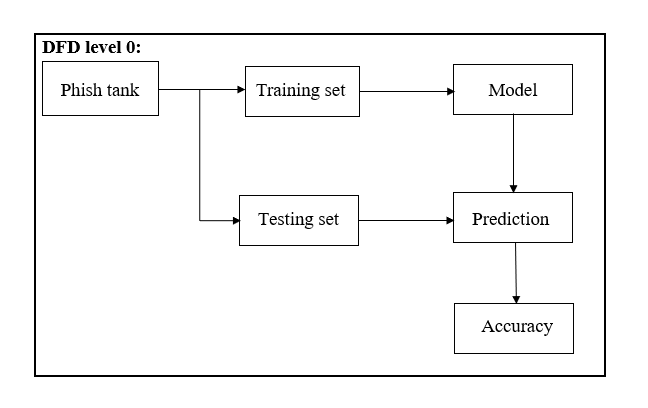
The architecture diagram describes the overall function of the project.

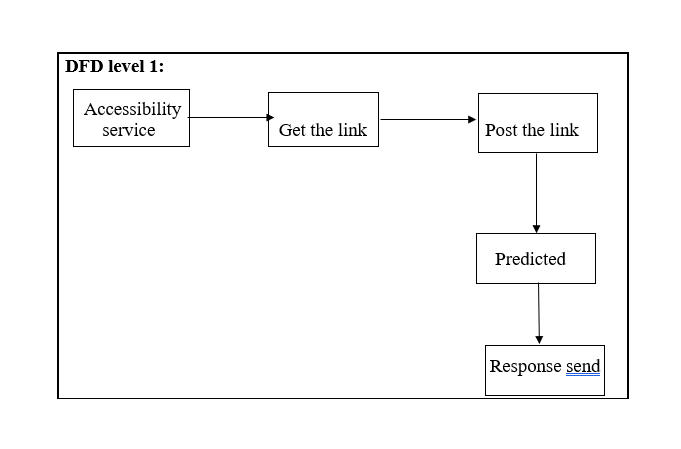
****

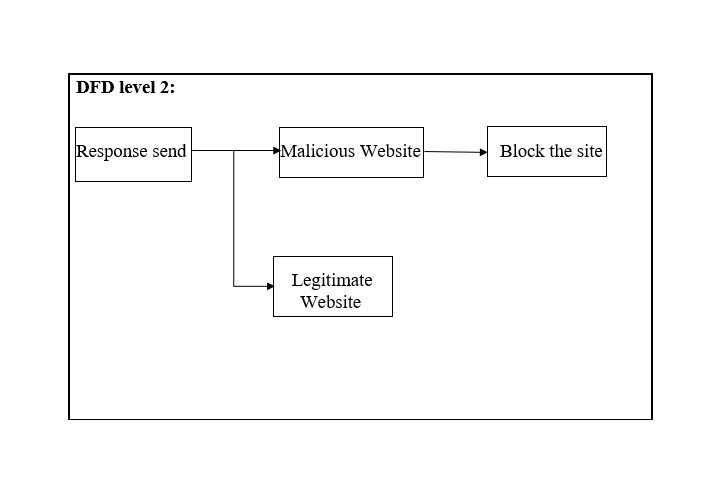
**Fig 6.1 System Architecture**

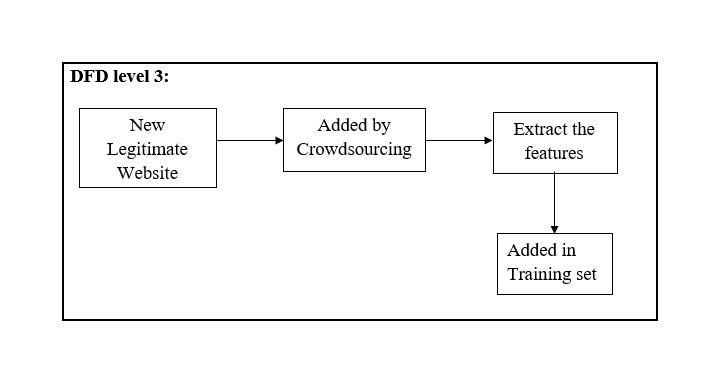
**6.2 DATAFLOW DIAGRAM**

A data flow diagram is a graphical representation of the flow of data through an information system, modeling its process aspects. DFDs can also be used for the visualization of data processing (structured design).

****

****

****

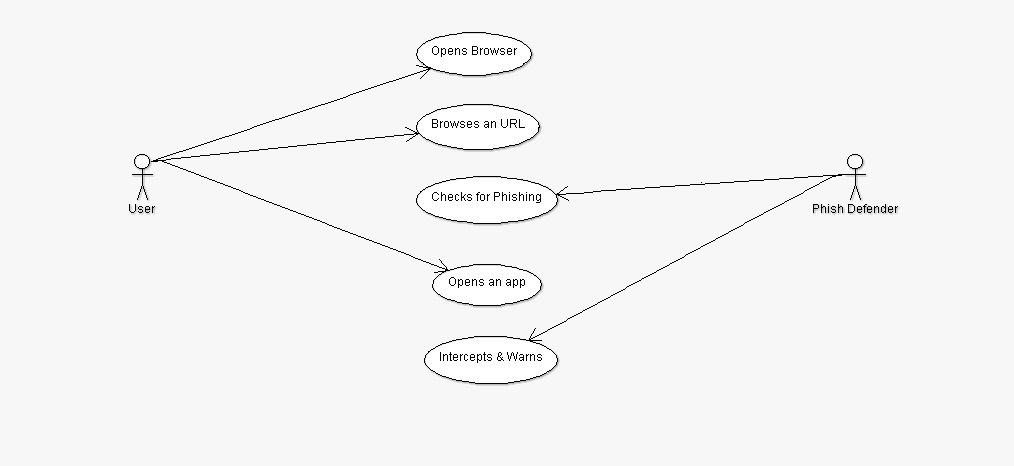
****

**Fig 6.2 Dataflow diagram**

**6.3 UML DIAGRAM**

**6.3.1 USECASE DIAGRAM**

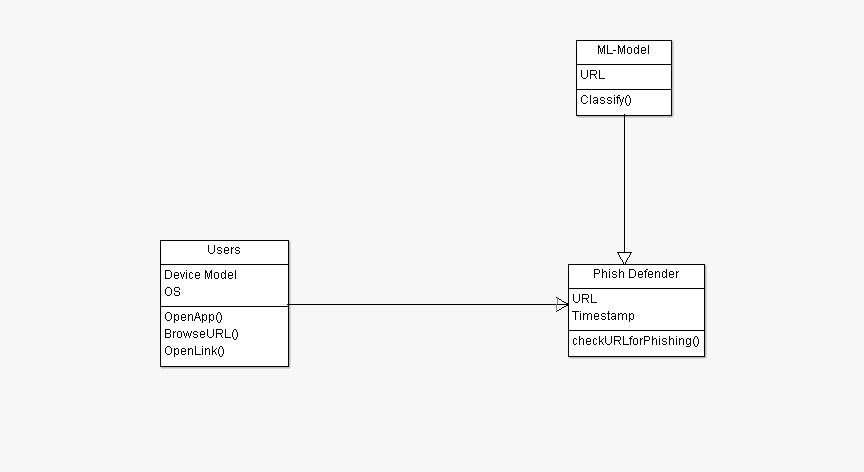
Use case diagrams are referred to as behaviors diagrams used to describe a set of actions (use cases) that some system or systems (subject) should or can perform in collaboration with one or more external users of the system (actors).A use case is a methodology used in system analysis to identify, clarify, and organize system requirements



**Fig 6.3.1 Use Case Diagram.**

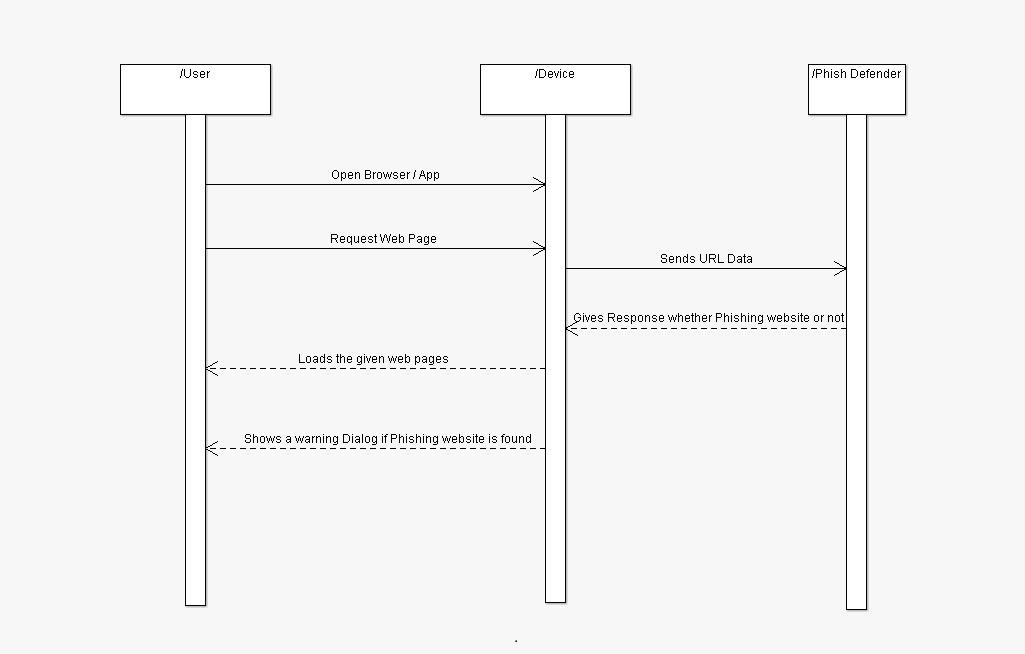
**6.3.2 CLASS DIAGRAM**

A class diagram in the Unified Modeling Language (UML) is a type of static structure diagram that describes the structure of a system by showing a system’s classes, their attributes operations (or methods), and the relationships among objects. The class diagram is the main building block of object-oriented modeling of the systematic of the application, and for detailed modeling translating the models into programming code. Class diagrams can also be used for data modeling. The classes in a class diagram represent boy the main elements, interactions in the applications, and the classes to be programmed

**Fig 6.3.2 Class Diagram**

**6.3.3 SEQUENCE DIAGRAM**

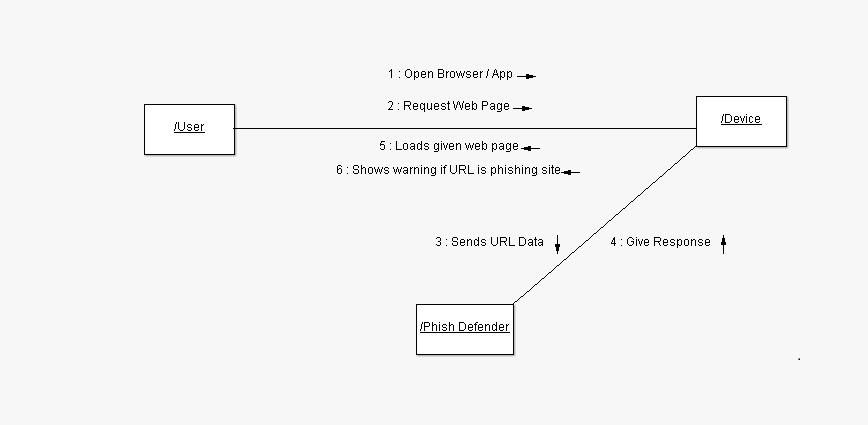
A sequence diagram is an interaction diagram that shows how objects operate with one another and in what order. It is a construct of a message sequence chart. A sequence diagram shows object interactions arranged in time sequence. It depicts the objects and classes involved in the scenario and the sequence of messages exchanged between the objects needed to carry out the functionality of the scenario. Sequence diagrams are typically associated with use case realizations in the logical view of the system under development. Sequence diagrams are sometimes called event diagrams or event scenarios.



**Fig 6.3.3 Sequence Diagram**

* + 1. **COLLABORATION DIAGRAM**

A collaboration diagram, also called a communication diagram or interaction diagram, is an illustration of the relationships and interactions among software objects in the Unified Modeling Language (UML).The concept is more than a decade old although it has been refined as modeling paradigms have evolved.



**Fig 6.3.4 Collaboration diagram**

**CHAPTER 7**

**SYSTEM IMPLEMENTATION**

* 1. **SYSTEM DESCRIPTION**

System implementation is the stage in the project where the theoretical design is turned into a working system. The most critical stage is achieving a successful system and building confidence on the new system for the user that it will work efficiently and effectively.

* 1. **MODULES**
* Accessibility module
* Parsing module
* API module
* Predicting module
* Floating widget module
* Logging module
* Crowdsourcing module

**Accessibility module**

This module will handle all the monitoring activities in the application. In this module, native Accessibility API is used to monitor the specified apps for all the URLs used by it.

The list of Apps which are need to be monitored can be configured for this module. If any link is found, it will notify the parsing module.

**Parsing module**

This module will handle all the parsing works in the app. This listens to the accessibility module for any data. Once, it got any data, it will then parse them as JSON so that it can send them via the internet to our server. Also, it handles all the redundancy, null data, and other errors. It listens to API module to parse the response data to a class object.

**API module**

This module handles all the communications between the Android application and the server via the http. It sends the data to the server as a formatted JSON file and also receives the response as JSON. This notifies the parsing module as soon as the data is received via http.

**Predicting module**

While all other module sits on the Android, this module is hosted on Flask in the server. This gets the data from the Android app via http as JSON. This has its own parsing functions built into it to get the data from the JSON. After getting the data, this will predict whether the link is suspicious or not and send the response via the Flask API.

**Floating widget module**

This module acts as an alert for the user even before the user opens the links. This listens to the data from parsing module. It shows a floating widget as an alert if any malicious site is detected.

**Logging module**

This module gets the response data from server via the parsing module. Upon receiving the data, it will sort the links as phishing and legit. Also, this log can be viewed as a list in Real-time.

**Crowdsourcing module**

This module handles getting the links from the user, stores it in the list and later, it will extract the new features from it. After that using Phishtank API it will send the links to the community powered database for helping others.

**CHAPTER 8**

**TESTING**

Testing is the process of detecting errors. Testing performs a very critical role for quality assurance and for ensuring the reliability of software. The results of testing are used later on during maintenance also.

**PSYCHOLOGY OF TESTING**

The aim of testing is often to demonstrate that a program works by showing that it has no errors. The basic purpose of testing phase is to detect the errors that may be present in the program. Hence one should not start testing with the intent of showing that a program works, but the intent should be to show that a program doesn’t work.

Testing is the process of executing a program with the intent of finding errors.

**TESTING OBJECTIVES**

The main objective of the testing is to uncover the host of errors, systematically and with minimum effort and time. Stating formally,

Testing is the process of executing a program with the intent of finding errors.

• A successful test is one that uncovers an as yet undiscovered error.

• A good test case is one that has a high probability of finding errors, if exists.

• The tests are adequate to detect possibly present error.

• The software more or less confirms to the quality and reliable standards.

**8.1 INTRODUCTION:**

Application Testing is the process used to help identify the correctness, completeness, security, and quality of developed user application. Testing is a process of technical investigation, performed on behalf of stakeholders, that is intended to reveal quality-related information about the product with respect to the context in which it is intended to operate. This includes, but is not limited to, the process of executing a program or application with the intent of finding errors. Quality is not an absolute; it is value to some person. With that in mind, testing can never completely establish the correctness of arbitrary computer software; testing furnishes a criticism or comparison that compares the state and behavior of the product against a specification. An important point is that software testing should be distinguished from the separate discipline of Software Quality Assurance (SQA), which encompasses all business process areas, not just testing.

**Test levels**

Unit testing tests the minimal software component and sub-component or modules by the programmers.

* Integration testing exposes defects in the interfaces and interaction between integrated components (modules).
* Functional testing tests the product according to programmable work. System testing tests an integrated system to verify/validate that it meets its requirements.
* Acceptance testing can be conducted by the client. It allows the end-user or customer or client to decide whether or not to accept the product. Acceptance testing may be performed after the testing and before the implementation phase. See also Development stage
* Beta testing comes after alpha testing. Versions of the software, known as beta versions, are released to a limited audience outside of the company. The software is released to groups of people so that further testing can ensure the product has few faults or bugs. Sometimes, beta versions are made available to the open public to increase the feedback field to a maximal number of future users.

**Test cases, suites, scripts and scenarios:**

A test case is a software testing document, which consists of event, action, input, output, expected result and actual result. Clinically defined (IEEE 829-1998) a test case is an input and an expected result. This can be as pragmatic as 'for condition x your derived result is y', whereas other test cases described in more detail the input scenario and what results might be expected. It can occasionally be a series of steps (but often steps are contained in a separate test procedure that can be exercised against multiple test cases, as a matter of economy) but with one expected result or expected outcome. The optional fields are a test case ID, test step or order of execution number, related requirement(s), depth, test category, author, and check boxes for whether the test is automatable and has been automated. A test case should also contain a place for the actual result. These steps can be stored in a word processor document, spreadsheet, database or other common repository. In a database system, you may also be able to see past test results and who generated the results and the system configuration used to generate those results. These past results would usually be stored in a separate table.

The term test script is the combination of a test case, test procedure and test data. Initially the term was derived from the byproduct of work created by automated regression test tools. Today, test scripts can be manual, automated or a combination of both. The most common term for a collection of test cases is a test suite. The test suite often also contains more detailed instructions or goals for each collection of test cases. It definitely contains a section where the tester identifies the system configuration used during testing. A group of test cases may also contain prerequisite states or steps, and descriptions of the following tests.

**8.2 TEST CASES:**

**GUIDELINES FOR TEST CASES:**

**GUI Test Cases**

* Total no of features that need to be check Look and Feel
* Look for Default values if at all any (date & Time, if at all any require) Look for spell check.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Test case name** | **Testcase description** | **Expected value** | **Actual value** | **Result** |
| GUI test | Check for all the features in the screen. | The screen must contain features | The result obtained on checking | True/False |
| GUI test | Check for the alignment of the objects as per the validations. | The alignment should be in proper way. | The result obtained on checking | True/False |

**CHAPTER 9**

**SAMPLE CODE**

**9.1 Android (Java and xml)**

**MainActivity.java**

package com.geeks4ever.phish;

import android.accessibilityservice.AccessibilityService;

import android.accessibilityservice.AccessibilityServiceInfo;

import android.content.Context;

import android.content.Intent;

import android.content.pm.ServiceInfo;

import android.net.Uri;

import android.os.Build;

import android.os.Bundle;

import android.provider.Settings;

import android.util.Log;

import android.view.MenuItem;

import android.view.View;

import android.view.accessibility.AccessibilityManager;

import android.widget.Button;

import android.widget.TextView;

import androidx.annotation.NonNull;

import androidx.annotation.RequiresApi;

import androidx.appcompat.app.AppCompatActivity;

import com.google.android.material.bottomnavigation.BottomNavigationView;

import java.util.List;

public class MainActivity extends AppCompatActivity {

public final static int REQUEST\_CODE = 676;

private static final String TAG = "Phish\_MainActivity";

public boolean enabled;

public Button btn;

public TextView status;

public BottomNavigationView bottomnavigationbar;

public static boolean isAccessibilityServiceEnabled(Context context, Class<? extends AccessibilityService> service) {

AccessibilityManager am = (AccessibilityManager) context.getSystemService(Context.ACCESSIBILITY\_SERVICE);

List<AccessibilityServiceInfo> enabledServices = am.getEnabledAccessibilityServiceList(AccessibilityServiceInfo.FEEDBACK\_ALL\_MASK);

for (AccessibilityServiceInfo enabledService : enabledServices) {

ServiceInfo enabledServiceInfo = enabledService.getResolveInfo().serviceInfo;

if (enabledServiceInfo.packageName.equals(context.getPackageName()) && enabledServiceInfo.name.equals(service.getName()))

return true;

}

return false;

}

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

btn = findViewById(R.id.isEnabled);

status = findViewById(R.id.ProtectionStatus);

accessibilityEnabledStatusCheck();

bottomnavigationbar = (BottomNavigationView) findViewById(R.id.bottom\_navigation);

//findViewById(R.id.buttonCreateWidget).setOnClickListener(this);

bottomnavigationbar.setOnNavigationItemSelectedListener(new BottomNavigationView.OnNavigationItemSelectedListener() {

@Override

public boolean onNavigationItemSelected(@NonNull MenuItem item) {

switch (item.getItemId()) {

case R.id.logging:

startActivity(new Intent(MainActivity.this, LogActivity.class));

break;

case R.id.addlinks:

startActivity(new Intent(MainActivity.this, AddLinksActivity.class));

break;

case R.id.settings:

startActivity(new Intent(MainActivity.this, SettingsActivity.class));

break;

}

return true;

}

});

}

public void accessibilityEnabledStatusCheck() {

if (enabled = isAccessibilityServiceEnabled(getApplicationContext(), MyAccessibilityService.class)) {

//Enabled

btn.setText("Disable");

btn.setBackgroundColor(0xFF00FF00);

btn.setTextColor(0xFF000000);

status.setText("You're Protected!");

if (Build.VERSION.SDK\_INT >= Build.VERSION\_CODES.M && !Settings.canDrawOverlays(this)) {

askPermission();

}

} else {

//disabled

//show Enable btn

btn.setText("Enable");

btn.setTextColor(0xFFFFFFFF);

btn.setBackgroundColor(0xFFFF0000);

status.setText("You're NOT Protected!");

}

}

public void checkEnabled(View view) {

if (enabled) {

//permission given

Log.d(TAG, "accessibility permission given");

getAccessibilityPermissions();

} else {

//not given show UI to give permission

Log.d(TAG, "accessibility permission NOT given");

getAccessibilityPermissions();

}

}

public void getAccessibilityPermissions() {

Log.d(TAG, "getting accessibility permissions!");

Intent intent = new Intent(android.provider.Settings.ACTION\_ACCESSIBILITY\_SETTINGS);

startActivityForResult(intent, 0);

}

@RequiresApi(api = Build.VERSION\_CODES.M)

private void askPermission() {

Intent intent = new Intent(Settings.ACTION\_MANAGE\_OVERLAY\_PERMISSION,

Uri.parse("package:" + getPackageName()));

startActivityForResult(intent, REQUEST\_CODE);

}

@Override

public void onRestart() {

super.onRestart();

// after resuming the activity

accessibilityEnabledStatusCheck();

}

}

**Accessibility service.java**

package com.geeks4ever.phish;

import android.accessibilityservice.AccessibilityService;

import android.content.Intent;

import android.net.Uri;

import android.provider.Settings;

import android.util.Log;

import android.view.accessibility.AccessibilityEvent;

import android.view.accessibility.AccessibilityNodeInfo;

import com.android.volley.AuthFailureError;

import com.android.volley.Request;

import com.android.volley.RequestQueue;

import com.android.volley.Response;

import com.android.volley.VolleyError;

import com.android.volley.toolbox.JsonObjectRequest;

import com.android.volley.toolbox.Volley;

import org.json.JSONArray;

import org.json.JSONException;

import org.json.JSONObject;

import java.io.UnsupportedEncodingException;

import java.net.URLDecoder;

import java.util.ArrayList;

import java.util.Arrays;

import java.util.HashMap;

import java.util.List;

import java.util.Map;

public class MyAccessibilityService extends AccessibilityService {

private static final String TAG = "Phish\_Defender";

private String currentURL = "";

public RequestQueue queue;

public boolean networkInit;

@Override

public void onAccessibilityEvent(AccessibilityEvent event) {

if(networkInit){

//already network request queue initialized

}else{

//init network request queue

queue = Volley.newRequestQueue(this);

}

AccessibilityNodeInfo source = event.getSource();

if (source == null)

return;

final String packageName = String.valueOf(source.getPackageName());

// Add browser package list here (comma seperated values)

String BROWSER\_LIST = "com.android.chrome, com.UCMobile.intl, org.mozilla.firefox, com.instagram.android, com.facebook.katana,com.whatsapp,com.oneplus.mms";

List<String> browserList

= Arrays.asList(BROWSER\_LIST.split(",\\s\*"));

if (event.getEventType()

== AccessibilityEvent.TYPE\_WINDOW\_CONTENT\_CHANGED) {

if (!browserList.contains(packageName)) {

return;

}

}

if (browserList.contains(packageName)) {

try {

// App opened is a browser.

// Parse urls in browser.

if (AccessibilityEvent

.eventTypeToString(event.getEventType())

.contains("TYPE\_WINDOW\_CONTENT\_CHANGED")) {

AccessibilityNodeInfo nodeInfo = event.getSource();

getUrlsFromViews(nodeInfo);

}

} catch(StackOverflowError ex){

ex.printStackTrace();

} catch (Exception ex) {

ex.printStackTrace();

}

}

}

/\*\*

\* Method to loop through all the views and try to find a URL.

\* @param info

\*/

public void getUrlsFromViews(AccessibilityNodeInfo info) {

try {

if (info == null)

return;

if (info.getText() != null && info.getText().length() > 0) {

String capturedText = info.getText().toString();

if (capturedText.contains("https://")

|| capturedText.contains("http://") || capturedText.contains("www.")) {

if (!currentURL.equals(capturedText)) {

// Do something with the url.

currentURL = capturedText;

Log.d(TAG, "Found URL: "+capturedText);

checkURL(currentURL);

}

}

}

for (int i = 0; i < info.getChildCount(); i++) {

AccessibilityNodeInfo child = info.getChild(i);

getUrlsFromViews(child);

if(child != null){

child.recycle();

}

}

} catch(StackOverflowError ex){

ex.printStackTrace();

} catch (Exception ex) {

ex.printStackTrace();

}

}

private void checkURL(String url){

if(url.contains("instagram")){

url = instagramURLDecoder(url);

}

//first check with Google

checkGoogleSafeBrowsing(url);

//then go for machine learning

//showFloatingWindow("https://www.google.com/sjhsfh/fsagfiushf");

}

private void showFloatingWindow(String urlToShowinView){

if (Settings.canDrawOverlays(this)) {

Intent i = new Intent(this, FloatingViewService.class);

i.putExtra("url",urlToShowinView);

startService(i);

//finish();

} else {

//askPermission();

Log.d(TAG, "You need System Alert Window Permission to do this");

//Toast.makeText(this, "You need System Alert Window Permission to do this", Toast.LENGTH\_SHORT).show();

}

}

private void machineLearningCheck(String urlToCheck){

try{

//url

final String urlToCheck2 = urlToCheck;

JSONObject urlObject = new JSONObject();

urlObject.put("url",urlToCheck);

String ML\_URL = "https://phish-defender.herokuapp.com/api";

String url = ML\_URL;

JsonObjectRequest jsonObjectRequest = new JsonObjectRequest(Request.Method.POST, url, urlObject,

new Response.Listener<JSONObject>()

{

@Override

public void onResponse(JSONObject response) {

// response

Log.d("Response", response.toString());

if(response.has("prediction")){

Log.d(TAG, "Got Response!!");

try{

int Result = response.getInt("prediction");

if(Result == 1){

//phishing site

//show floatUI

showFloatingWindow(urlToCheck2);

Log.d(TAG, "Phishing Site!!");

}

}catch(JSONException jsx){

Log.d(TAG, jsx.toString());

}

//showFloatingWindow();

}

}

},

new Response.ErrorListener()

{

@Override

public void onErrorResponse(VolleyError error) {

// error

Log.d("Error.Response", error.toString());

}

}

)

{

@Override

public Map<String, String> getHeaders() throws AuthFailureError {

Map<String, String> params = new HashMap<String, String>();

params.put("Content-Type", "application/json");

return params;

}

};

//queue.add(postRequest);

//jsonObjectRequest.setTag(REQ\_TAG);

queue.add(jsonObjectRequest);

}catch(Exception ignored){

}

}

private void checkGoogleSafeBrowsing(String urlToCheck){

try{

final String urlToCheck2 = urlToCheck;

JSONObject json = getJsonObject(urlToCheck);

Log.d(TAG, json.toString());

String googleApiURL = "https://safebrowsing.googleapis.com/v4/threatMatches:find?key=AIzaSyDVhCTR3IWUfteUGVugMEepE235\_50TlLY";

String url = googleApiURL;

JsonObjectRequest jsonObjectRequest = new JsonObjectRequest(Request.Method.POST, url, json,

new Response.Listener<JSONObject>()

{

@Override

public void onResponse(JSONObject response) {

// response

Log.d("Response", response.toString());

if(response.has("matches")){

Log.d(TAG, "Phishing Site!!");

showFloatingWindow(urlToCheck2);

}else{

//check with machine learning API

machineLearningCheck(urlToCheck2);

}

}

},

new Response.ErrorListener()

{

@Override

public void onErrorResponse(VolleyError error) {

// error

Log.d("Error.Response", error.toString());

}

}

)

{

@Override

public Map<String, String> getHeaders() throws AuthFailureError {

Map<String, String> params = new HashMap<String, String>();

params.put("Content-Type", "application/json");

return params;

}

};

//queue.add(postRequest);

//jsonObjectRequest.setTag(REQ\_TAG);

queue.add(jsonObjectRequest);

}catch(JSONException jsx){

Log.d(TAG, jsx.toString());

}

}private String instagramURLDecoder(String url){

String mainLink = url;

try{

String afterDecode = URLDecoder.decode(url, "UTF-8");

Uri uri = Uri.parse(afterDecode);

mainLink = uri.getQueryParameter("u");

Log.d(TAG, "got intagram main link: "+mainLink);

}catch(UnsupportedEncodingException unsupp){

Log.d(TAG, unsupp.toString());

}

return mainLink;

}

private JSONObject getJsonObject(String url) throws JSONException {

//main object

JSONObject jsonObject = new JSONObject();

//url

JSONObject urlObject = new JSONObject();

urlObject.put("url",url);

List <JSONObject> urlList = new ArrayList<JSONObject>();

urlList.add(urlObject);

JSONArray urlArray = new JSONArray(urlList);

//client

JSONObject clientObject = new JSONObject();

JSONObject clientJson = new JSONObject();

clientJson.put("clientId", "RPADML");

clientJson.put("clientVersion", "1.5.2");

//clientObject.put("client", clientJson); //threat

JSONObject threatObject = new JSONObject();

JSONObject threatJson = new JSONObject();

threatJson.put("threatTypes","SOCIAL\_ENGINEERING");

threatJson.put("platformTypes","WINDOWS");

threatJson.put("threatEntryTypes","URL");

threatJson.put("threatEntries",urlArray);

//threatObject.put("threatInfo",threatJson);

//final Object

jsonObject.put("client",clientJson);

jsonObject.put("threatInfo",threatJson);

return jsonObject;

}

@Override

public void onInterrupt() {}

**}**

**Project.gradle**

// Top-level build file where you can add configuration options common to all sub-projects/modules.

buildscript {

ext.kotlin\_version = '1.4.31'

repositories {

google()

jcenter()

}

dependencies {

classpath 'com.android.tools.build:gradle:4.1.3'

classpath "org.jetbrains.kotlin:kotlin-gradle-plugin:$kotlin\_version"

// NOTE: Do not place your application dependencies here; they belong

// in the individual module build.gradle files

}

}

allprojects {

repositories {

google()

jcenter()

}

}

task clean(type: Delete) {

delete rootProject.buildDir

}

**9.2 Python (Backend code)**

**App.py**

# importing libraries

from flask import Flask, abort, jsonify, request, render\_template

from flask\_cors import CORS, cross\_origin

from flask\_ngrok import run\_with\_ngrok

import time

import json

import datetime

import os

from flask\_restful import Resource, Api

import joblib

import inputScript

import numpy as np

app = Flask(\_\_name\_\_)

# Start ngrok when app is run

# run\_with\_ngrok(app)

# root directory

@app.route("/")

def hello():

return render\_template('index.html')

@app.route("/api", methods=['POST'])

def make\_predict():

# error checking

data = request.get\_json(force=True)

# extract url from request

url\_to\_be\_predicted = data['url']

url = str(url\_to\_be\_predicted)

# load the pickle file

classifier = joblib.load('final\_models/rf\_final.pkl')

#checking and predicting

try:

checkprediction = inputScript.main(url)

prediction = int(classifier.predict(checkprediction))

print(prediction)

result = {"prediction": prediction}

except Exception as e:

print(e)

result = {"prediction": -9999}

return jsonify(result)

if \_\_name\_\_ == "\_\_main\_\_":

# when deploying in NGROK server(PC as server)

# app.run()

# When using Heroku or any other online platform for deployment

app.run(debug=True,host='0.0.0.0', port=int(os.environ.get("PORT", 5000)))

inputscript.py

# importing libraries

import regex

from tldextract import extract

import ssl

import socket

from bs4 import BeautifulSoup

import urllib.request

import datetime

import whois

def url\_having\_ip(url):

# using regular function

# symbol = regex.findall(r'(http((s)?)://)((((\d)+).)\*)((\w)+)(/((\w)+))?',url)

# if(len(symbol)!=0):

# having\_ip = 1 #phishing

# else:

# having\_ip = -1 #legitimate

# return(having\_ip)

return 0

def url\_length(url):

length = len(url)

if(length < 54):

return -1

elif(54 <= length <= 75):

return 0

else:

return 1

def url\_short(url):

# ongoing

return 0

def having\_at\_symbol(url):

symbol = regex.findall(r'@', url)

if(len(symbol) == 0):

return -1

else:

return 1

def doubleSlash(url):

# ongoing

return 0

def prefix\_suffix(url):

subDomain, domain, suffix = extract(url)

if(domain.count('-')):

return 1

else:

return -1

def sub\_domain(url):

subDomain, domain, suffix = extract(url)

if(subDomain.count('.') == 0):

return -1

elif(subDomain.count('.') == 1):

return 0

else:

return 1

def SSLfinal\_State(url):

try:

# check wheather contains https

if(regex.search('^https', url)):

usehttps = 1

else:

usehttps = 0

# getting the certificate issuer to later compare with trusted issuer

# getting host name

subDomain, domain, suffix = extract(url)

host\_name = domain + "." + suffix

context = ssl.create\_default\_context()

sct = context.wrap\_socket(socket.socket(), server\_hostname=host\_name)

sct.connect((host\_name, 443))

certificate = sct.getpeercert()

issuer = dict(x[0] for x in certificate['issuer'])

certificate\_Auth = str(issuer['commonName'])

certificate\_Auth = certificate\_Auth.split()

if(certificate\_Auth[0] == "Network" or certificate\_Auth == "Deutsche"):

certificate\_Auth = certificate\_Auth[0] + " " + certificate\_Auth[1]

else:

certificate\_Auth = certificate\_Auth[0]

trusted\_Auth = ['Comodo', 'Symantec', 'GoDaddy', 'GlobalSign', 'DigiCert', 'StartCom', 'Entrust', 'Verizon', 'Trustwave', 'Unizeto', 'Buypass',

'QuoVadis', 'Deutsche Telekom', 'Network Solutions', 'SwissSign', 'IdenTrust', 'Secom', 'TWCA', 'GeoTrust', 'Thawte', 'Doster', 'VeriSign']

# getting age of certificate

startingDate = str(certificate['notBefore'])

endingDate = str(certificate['notAfter'])

startingYear = int(startingDate.split()[3])

endingYear = int(endingDate.split()[3])

Age\_of\_certificate = endingYear-startingYear

# checking final conditions

if((usehttps == 1) and (certificate\_Auth in trusted\_Auth) and (Age\_of\_certificate >= 1)):

return -1 # legitimate

elif((usehttps == 1) and (certificate\_Auth not in trusted\_Auth)):

return 0 # suspicious

else:

return 1 # phishing

except Exception as e:

print(e)

return 1

def domain\_registration(url):

try:

w = whois.whois(url)

updated = w.updated\_date

exp = w.expiration\_date

length = (exp[0]-updated[0]).days

if(length <= 365):

return 1

else:

return -1

except:

return 0

def favicon(url):

# ongoing

return 0

def port(url):

# ongoing

return 0

def https\_token(url):

subDomain, domain, suffix = extract(url)

host = subDomain + '.' + domain + '.' + suffix

if(host.count('https')): # attacker can trick by putting https in domain part

return 1

else:

return -1

def request\_url(url):

try:

subDomain, domain, suffix = extract(url)

websiteDomain = domain

opener = urllib.request.urlopen(url).read()

soup = BeautifulSoup(opener, 'lxml')

imgs = soup.findAll('img', src=True)

total = len(imgs)

linked\_to\_same = 0

avg = 0

for image in imgs:

subDomain, domain, suffix = extract(image['src'])

imageDomain = domain

if(websiteDomain == imageDomain or imageDomain == ''):

linked\_to\_same = linked\_to\_same + 1

vids = soup.findAll('video', src=True)

total = total + len(vids)

for video in vids:

subDomain, domain, suffix = extract(video['src'])

vidDomain = domain

if(websiteDomain == vidDomain or vidDomain == ''):

linked\_to\_same = linked\_to\_same + 1

linked\_outside = total-linked\_to\_same

if(total != 0):

avg = linked\_outside/total

if(avg < 0.22):

return -1

elif(0.22 <= avg <= 0.61):

return 0

else:

return 1

except:

return 0

def url\_of\_anchor(url):

try:

subDomain, domain, suffix = extract(url)

websiteDomain = domain

opener = urllib.request.urlopen(url).read()

soup = BeautifulSoup(opener, 'lxml')

anchors = soup.findAll('a', href=True)

total = len(anchors)

linked\_to\_same = 0

avg = 0

for anchor in anchors:

subDomain, domain, suffix = extract(anchor['href'])

anchorDomain = domain

if(websiteDomain == anchorDomain or anchorDomain == ''):

linked\_to\_same = linked\_to\_same + 1

linked\_outside = total-linked\_to\_same

if(total != 0):

avg = linked\_outside/total

if(avg < 0.31):

return -1

elif(0.31 <= avg <= 0.67):

return 0

else:

return 1

except:

return 0

def Links\_in\_tags(url):

try:

opener = urllib.request.urlopen(url).read()

soup = BeautifulSoup(opener, 'lxml')

no\_of\_meta = 0

no\_of\_link = 0

no\_of\_script = 0

anchors = 0

avg = 0

for meta in soup.find\_all('meta'):

no\_of\_meta = no\_of\_meta+1

for link in soup.find\_all('link'):

no\_of\_link = no\_of\_link + 1

for script in soup.find\_all('script'):

no\_of\_script = no\_of\_script+1

for anchor in soup.find\_all('a'):

anchors = anchors+1

total = no\_of\_meta + no\_of\_link + no\_of\_script+anchors

tags = no\_of\_meta + no\_of\_link + no\_of\_script

if(total != 0):

avg = tags/total

if(avg < 0.25):

return -1

elif(0.25 <= avg <= 0.81):

return 0

else:

return 1

except:

return 0

def sfh(url):

# ongoing

return 0

def email\_submit(url):

try:

opener = urllib.request.urlopen(url).read()

soup = BeautifulSoup(opener, 'lxml')

if(soup.find('mailto:')):

return 1

else:

return -1

except:

return 0

def abnormal\_url(url):

# ongoing

return 0

def redirect(url):

# ongoing

return 0

def on\_mouseover(url):

# ongoing

return 0

def rightClick(url):

# ongoing

return 0

def popup(url):

# ongoing

return 0

def iframe(url):

# ongoing

return 0

def age\_of\_domain(url):

try:

w = whois.whois(url)

start\_date = w.creation\_date

current\_date = datetime.datetime.now()

age = (current\_date-start\_date[0]).days

if(age >= 180):

return -1

else:

return 1

except Exception as e:

#print("Here we go")

print(e)

return 0

def dns(url):

# ongoing

return 0

def web\_traffic(url):

# ongoing

return 0

def page\_rank(url):

# ongoing

return 0

def google\_index(url):

# ongoing

return 0

def links\_pointing(url):

# ongoing

return 0

def statistical(url):

# ongoing

return 0

def main(url):

check = [[url\_having\_ip(url), url\_length(url), url\_short(url), having\_at\_symbol(url),

doubleSlash(url), prefix\_suffix(

url), sub\_domain(url), SSLfinal\_State(url),

domain\_registration(url), favicon(url), port(

url), https\_token(url), request\_url(url),

url\_of\_anchor(url), Links\_in\_tags(url), sfh(

url), email\_submit(url), abnormal\_url(url),

redirect(url), on\_mouseover(url), rightClick(

url), popup(url), iframe(url),

age\_of\_domain(url), dns(url), web\_traffic(

url), page\_rank(url), google\_index(url),

links\_pointing(url), statistical(url)]]

print(check)

return check

**Procfile**

web: gunicorn --bind 0.0.0.0:$PORT app:app

**Randomforest.py**

# importing libraries

from sklearn.metrics import confusion\_matrix

from sklearn.model\_selection import GridSearchCV

from sklearn.model\_selection import train\_test\_split

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

from sklearn.ensemble import RandomForestClassifier

import joblib

import os

# get current directory

current\_directory = os.getcwd()

# importing the dataset

dataset = pd.read\_csv(current\_directory+"/train\_model/datasets/dataset.csv")

dataset = dataset.drop('id', 1) # removing unwanted column

x = dataset.iloc[:, :-1].values

y = dataset.iloc[:, -1:].values

# spliting the dataset into training set and test set

x\_train, x\_test, y\_train, y\_test = train\_test\_split(

x, y, test\_size=0.25, random\_state=0)

# applying grid search to find best performing parameters

parameters = [{'n\_estimators': [100, 700],

'max\_features': ['sqrt', 'log2'],

'criterion':['gini', 'entropy']}]

grid\_search = GridSearchCV(RandomForestClassifier(),

parameters, cv=5, n\_jobs=-1)

grid\_search.fit(x\_train, y\_train.ravel())

# printing best parameters

print("Best Accuracy =" + str(grid\_search.best\_score\_))

print("best parameters =" + str(grid\_search.best\_params\_))

# fitting RandomForest regression with best params

classifier = RandomForestClassifier(

n\_estimators=100, criterion="gini", max\_features='log2', random\_state=0)

classifier.fit(x\_train, y\_train.ravel())

# predicting the tests set result

y\_pred = classifier.predict(x\_test)

# confusion matrix

cm = confusion\_matrix(y\_test, y\_pred)

print(cm)

# pickle file joblib

joblib.dump(classifier, current\_directory+"/final\_models/rf\_final.pkl")

# features importance random forest

names = dataset.iloc[:, :-1].columns

importances = classifier.feature\_importances\_

sorted\_importances = sorted(importances, reverse=True)

indices = np.argsort(-importances)

var\_imp = pd.DataFrame(

sorted\_importances, names[indices], columns=['importance']

# plotting variable importance

plt.title("Variable Importances")

plt.barh(np.arange(len(names)), sorted\_importances, height=0.7)

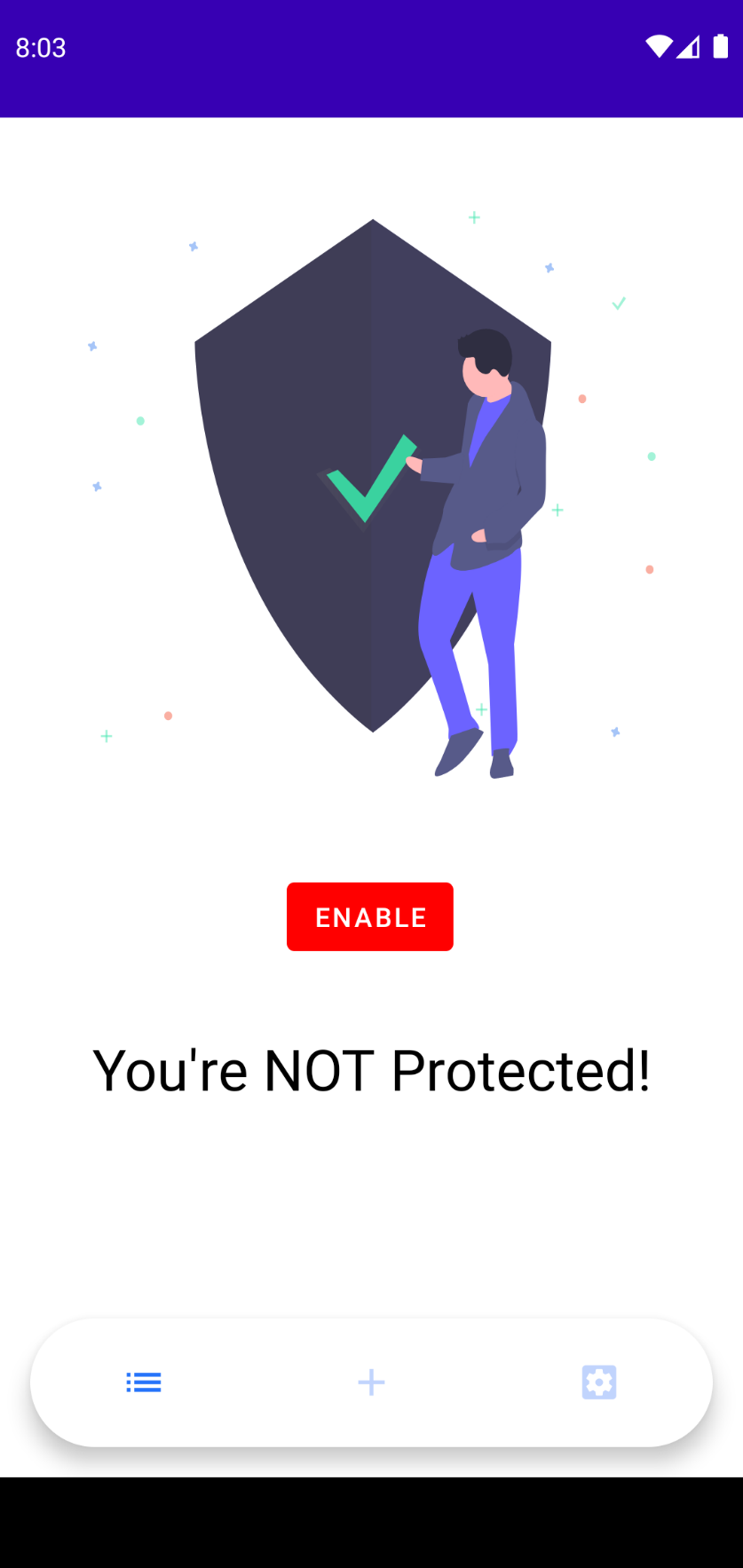
plt.yticks(np.arange(len(names)), names[indices], fontsize=7)

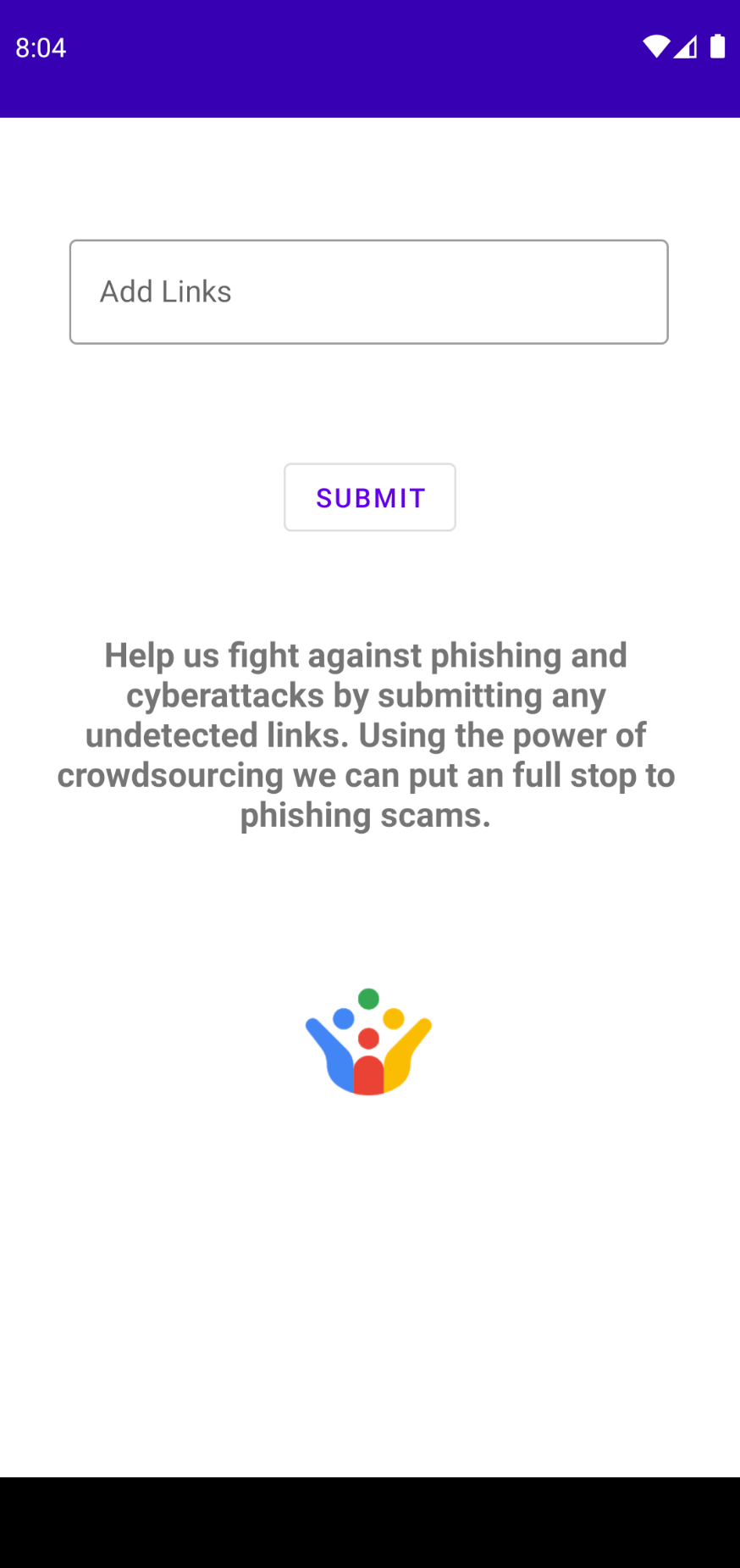
plt.xlabel('Relative Importance')

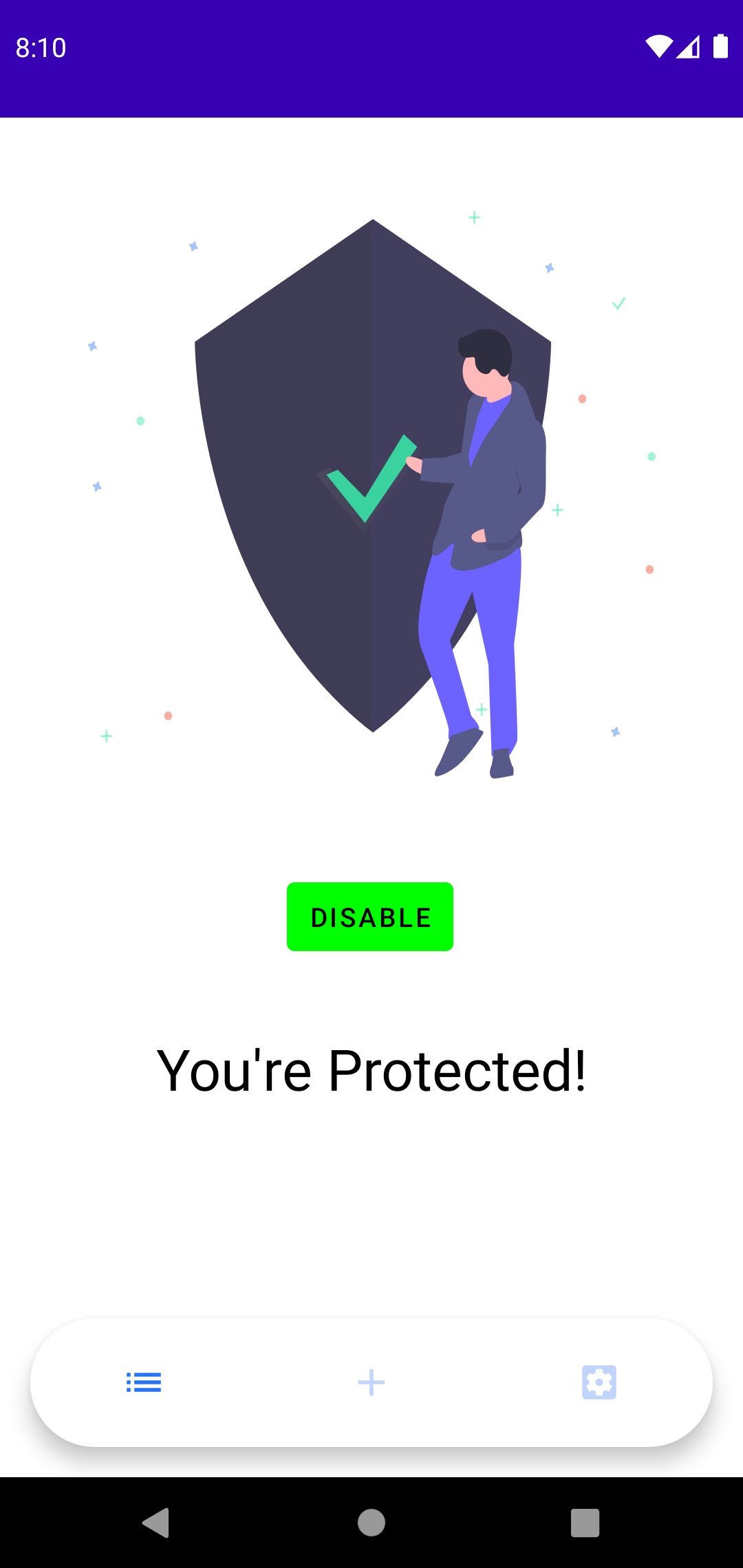
plt.show()

**CHAPTER 10**

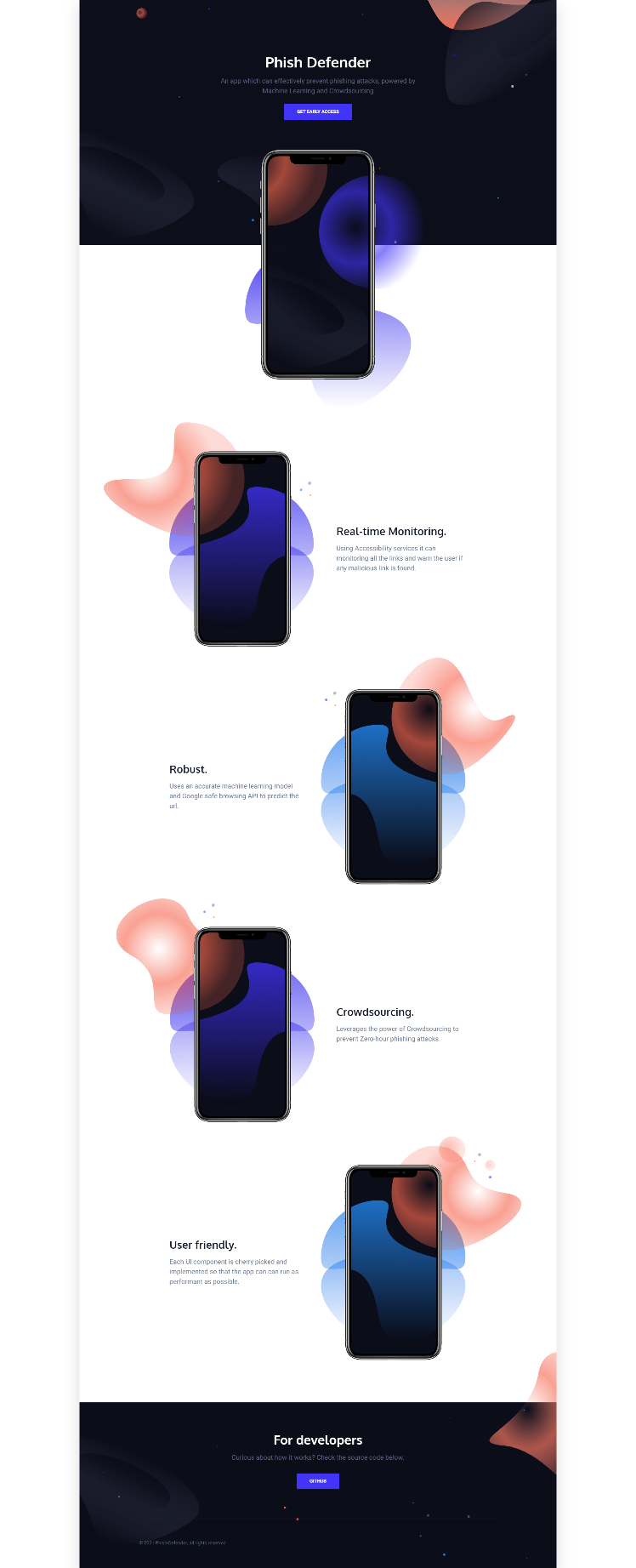
**SCREENSHOTS**

****

****

****

**Flask API**

****

**CHAPTER 11**

**CONCLUSION AND FUTURE ENCHANCEMENT**

Phishing attacks through URLs is one of the major issues faced by the Internet Community because of the online transactions performed on a day-to-day basis. The phishing attacks cause severe loss to companies, customers and web users. Social networking sites such as Facebook, Twitter and LinkedIn have been the victims of phishing. The Machine Learning based Phishing Detection System detects the malicious URLs and specify the reason for classifying a URL as phishing which will help the users to be aware of such malicious and suspicious URLs. After the user enters a URL, the proposed model applies an input URL to determine whether a URL is malicious or legitimate. The malicious URLs are blocked accordingly. It stores all the input URLs in the Crowdsourcing database which can be retrieved for the later purpose. The complete Intelligent Phishing website detection model is developed as a user- friendly Android based mobile application.

In future, the links can be detected without the need of the server and the accuracy can be improved by reducing useless features.

**CHAPTER 12**

**REFERENCES**

[1] MahaLakshmi, N. Swapna Goud, Dr. G. Vishnu Murthy,” Survey on Phishing And It’s Detection Techniques Based on Support Vector Method (SVM) and Software Defined Networking(SDN)”, nternational Journal of Engineering and Advanced Technology (IJEAT) December 2018.

[2] Gururaj Harinahalli Lokesh & Goutham BoreGowda,”Phishing website detection based on effective machine learning approach” , Journal of Cyber Security Technology, Published online 31 Aug 2020.

[3] Mohith Gowda , Adithya , Gunesh Prasad and Vinay “ Development of anti-phishing browser based on random forest and rule of extraction framework” ,Received: 20 February 2020 Accepted: 2 September 2020,Published Online:october 14,2020

[4] Raghav Kaul, Shahriar Badsha, Shamik Sengupta “ An Automated Framework for Real-time Phishing URL Detection”Dept. of Computer Science and Engineering University of Nevada, Reno, NV, USA fsadique@nevada.unr.edu,, sbadsha@unr.edu, [ssengupta@unr.edu](mailto:ssengupta@unr.edu).

[5] Routhu Srinivasa Rao, Tatti Vaishnavi, Alwyn Roshan Pais “CatchPhish: detection of phishing websites by inspecting URLs” published Online May 10 2019,

[6] Rishikesh Mahajan, Irfan Siddavatam”Phishing Website Detection using Machine Learning Algorithms” International Journal of Computer Applications October 2018

[7] Subasi, A. Molah, E.Almkallawi, F.Chaudhery, T. J.; “Intelligent phishing website detection using random forest classifer”. International Conference Electrical Computer Technology Approved in IEEE, (2017)

[8] Singh, C, Smt. Meenu,” Phishing website detection based on machine learning”,published on 6th International Conference on Advanced Computing & Communication Systems (ICACCS), 2020.

[9] Yazan Ahmad Alsariera, Adeyemo Victor Elijah, Abdullateef O. Balogun “Phishing Website Detection” Forest by Penalizing Attributes Algorithm and Its Enhanced Variations” Journal of Cyber Security Technology, Published online 31 Aug 2020.

[10] Yasin Sönmez, Turkey yasin, Hüseyin Gökal, Türker Tuncer,” Phishing Web Sites Features Classification Based on Extreme Learning Machine”, International Journal of Computer Applications October 2019