“Robust AI Driven Phishing Detection Tool.”

CHAPTER 1

INTRODUCTION

In an age where digital communication is paramount, the threat of cyberattacks continues to escalate, with phishing emerging as one of the most prevalent and damaging tactics. Phishing attacks leverage deceptive, websites, or messages to extract sensitive information from unsuspecting users. **[1]**Despite growing awareness, the sophistication of these attacks makes them increasingly difficult to detect and combat.

The motivation behind our project is to address this critical security challenge by providing a robust AI-driven solution specifically designed to detect phishing in URLs. As cybercriminals employ more advanced techniques to bypass traditional security measures, the need for intelligent, adaptable detection mechanisms has never been greater.

The existing methods for detecting phishing often rely on static blacklists or heuristic-based approaches, which can quickly become outdated or generate high false-positive rates. This leads to a significant gap in defense, exposing both individuals and organizations to potential data breaches, financial loss, and reputational damage.

1.1 Need for Project

Our project aims to bridge this gap by developing an advanced AI-based tool that can analyze and identify phishing URLs with high accuracy and efficiency. Utilizing javascript based algorithms, natural language processing, and data analytics, the tool is designed to detect subtle patterns and features indicative of phishing, even as attackers evolve their strategies.

By implementing our robust AI-driven phishing detection tool, we aim to significantly enhance the security posture of organizations and individuals, providing a reliable safeguard against the ever-changing landscape of cyber threats. **[2]**This project will not only help mitigate the risk of phishing attacks but also contribute to the broader goal of creating a safer digital environment for all users.

1.2 SCOPE OF THE PROJECT

- **Real-Time Phishing Detection:** The browser extension will provide real-time detection capabilities for both phishing and non-trusted URLs. It will analyse incoming as users read them within popular web browsers like Chrome and Firefox, assessing, URLs, and attachments for potential threats.

- **User-Friendly Interface:** The extension will offer a user-friendly interface, ensuring that users receive clear and actionable alerts when a potentially phishing or non-trusted URL is detected. The interface will guide users on how to respond effectively to both types of threats.

- **Compatibility:** The extension will be designed to work seamlessly with various web browsers ensuring broad compatibility and accessibility for users.

- **Alerting Mechanism:** Users will receive immediate alerts and warnings if a suspicious non-trusted URL is detected. The extension will provide information and recommendations for handling potential threats.

- **Data Privacy:** Privacy considerations will be a top priority in the extension's design, ensuring that user data is handled securely and remains confidential.

1.3 Problem Statement:

1. A chrome extension that detects the URL status(Phishing\No Phishing) in real time.

2. Integrate the extension with a Database(OracleDB).

3. Create Login and Registration page for convenient data storage.

4. Integrate the extension with the mobile app using Android Studio and Figma.

1.4 Our Solution:

Our solution is an advanced AI-driven phishing detection system designed to identify potentially harmful URLs in real-time. Through a Chrome extension, we continuously monitor the websites users visit, analyzing their URLs for signs of phishing. The extension provides immediate pop-up alerts when it detects suspicious activity, allowing users to quickly make informed decisions about their online safety. This real-time monitoring is powered by javascript based algorithms that learn from vast data sets, ensuring that our detection capabilities are constantly evolving to stay ahead of emerging threats.

Beyond immediate detection, our solution offers a centralized data management system where detection outcomes are stored securely. This database links each record with the user's ID, prediction result, and timestamp, enabling comprehensive trend analysis and security insights. To complement the browser extension, we offer an Android app that provides users with a clear visualization of their browsing activity, including graphical representations of phishing detection trends. Through this combination of real-time detection, secure data storage, and user-friendly mobile app integration, our solution provides robust protection against phishing threats.

CHAPTER 2

LITERATURE SURVEY AND RESEARCH BACKGROUND

Phishing detection tool represents a burgeoning field of study, marked by the publication of numerous surveys and review articles that examine the current state of intrusion detection. This section explores the importance of incorporating an Phishing detection tool on the client-side when designing an anti-phishing system. The subsequent chapter delves into a comprehensive discussion of relevant studies and research gaps, encompassing diverse research perspectives from various authors. In a subsequent section, we formulate the problem that drives the creation of the proposed anti-phishing system through javascript logic algorithm, underscoring the significance of this work.

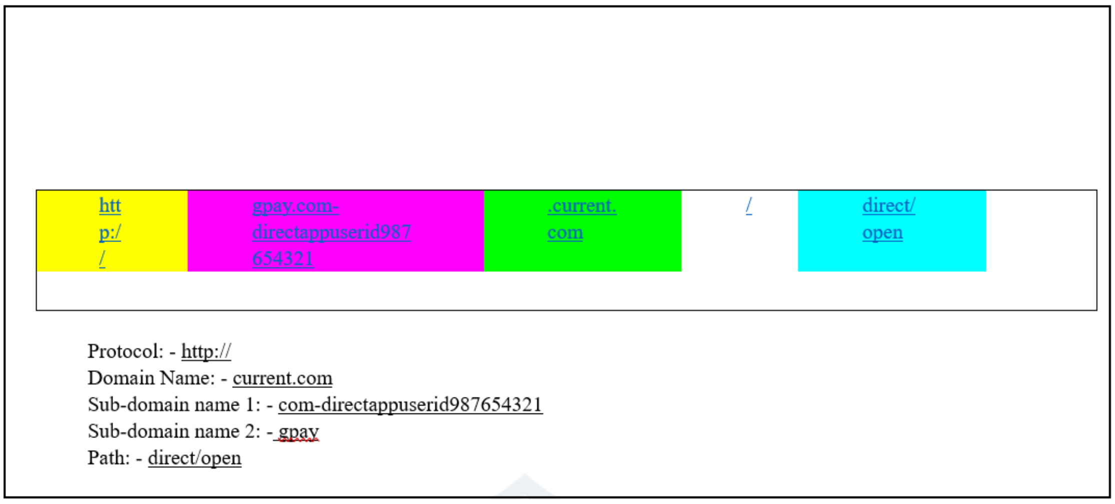
2.1 Cutting-edge: Dynamic and Static Analysis

In 2013,Birhanu explained that the malicious webpage detection industry employs two types of analyses. The first one is dynamic analysis, where the investigation starts after execution, and the malicious behaviour manifests itself in the form of actions. By closely monitoring actions such as internet data logs, the sequence of system calls, and suspicious system activities, attacks can be detected. This approach is known as anomaly detection, and the method used for this purpose is termed dynamic analysis. However, when it comes to identifying malicious URLs through dynamic analysis, it carries a high risk since the system has already been exposed to a malicious URL and might have been compromised. To prevent such infections, one can execute and analyse the malicious URL in a Controlled Cloud Sandbox.

The second category of analysis is referred to as Static analysis, where the investigation occurs before execution, and it relies on the information available within the URL itself. This information includes attributes like lexical parameters within the URL string, host details, and sometimes the content of Hypertext Markup Language (HTML) and JavaScript code. Benign and malicious URLs have distinct information distributions, and using these feature distributions, a predictive framework can be developed. Due to the safer environment for extracting relevant information to identify potential threats, and the need for creating a signature for detection, static analysis techniques have been extensively explored by applying logics.

2.2 Analysis of URLs:

Uniform Resource Locators (URLs), also commonly known as 'Hyperlinks,' serve as the primary means for individuals to access information on the internet. This chapter's objective is to detect malicious websites by analysing the lexical and host-based attributes within their URLs. As such, this section provides an overview of the problem at hand, offers insights into potential solutions involving URLs, and describes the features employed in the proposed application. Malicious actors can deliberately select domain names to deceive users, making it challenging to identify fraudulent sites. In Figure 2.1, although the actual domain name is 'current.com,' the attacker attempted to mimic 'gpay.com' by incorporating 'Free URL.' When users encounter 'gpay.com' at the initial stages of the URL, they tend to trust the webpage, interact with it, and may unknowingly share their sensitive information with the deceptive site. This scenario is a common tactic employed by attackers and is examined further below.



Protocol: - http://
Domain Name: - current.com
Sub-domain name 1: - com-directappuserid987654321
Sub-domain name 2: - gpay
Path: - direct/open

Figure 2.1 Fraudulent Web Link

In response to these vulnerabilities, various application-oriented strategies have been developed to counter phishing attacks. These approaches vary from basic, list-driven approaches to the utilization of javascript logics. Previous research has explored and discussed multiple research papers focused on detecting phishing attacks. Numerous anti-phishing solutions have been devised with the aim of addressing and mitigating this type of attack, ultimately safeguarding users from falling victim to phishing attempts. In response to these vulnerabilities, various application-oriented strategies have been developed to counter phishing attacks. These approaches vary from basic, list-driven approaches to the utilization of javascript logics. Previous research has explored and discussed multiple research papers focused on detecting phishing attacks. Numerous anti-phishing solutions have been devised with the aim of addressing and mitigating this type of attack, ultimately safeguarding users from falling victim to phishing attempts.

2.3 Phishing : A Malware Practice

Phishing is a deceptive tactic employed by cybercriminals to trick individuals into divulging sensitive data, including usernames, passwords, or credit card information. The methods for countering this menace have progressed over time. Here are a few noteworthy developments:

1. Javascript Logics: Javascript along with the algorithms have been employed to analyse and detect phishing websites. These algorithms can identify patterns and characteristics that distinguish legitimate websites from phishing ones. This technique has significantly improved detection accuracy.
2. URL Analysis: Advanced URL analysis techniques have been developed to identify deceptive URLs used in phishing attacks. These techniques examine the structure, components, and domain reputation to determine the likelihood of a URL being malicious.
3. Browser Warnings: Web browsers have implemented warning mechanisms to alert users when they visit potentially malicious websites. These warnings inform users about potential risks and discourage them from entering sensitive information..

2.4 Prior Studies and Knowledge Gaps:

Jain, A.K., Gupta, B.B. et al (2016) **[3]** had proposed a novel approach to protect against phishing attacks at the client side using an auto-updated white-list. It had highlighted the limitations of blacklist-based solutions, which offered fast access but struggled with low detection rates, as well as other methods like visual similarity , which had prioritized accuracy over speed. It had involved the use of an automatically updated whitelist of legitimate sites accessed by users. This approach had successfully combined fast access and high detection rates. When users had attempted to access a site not on the whitelist, their browser had issued a warning. The method had also verified webpage legitimacy using hyperlink features from the source code and had employed a phishing detection algorithm. Experimental results had validated the effectiveness of this approach, achieving an 86.02% true positive rate and less than a 1.48% false negative rate. Additionally, the system had efficiently detected various types of phishing attacks, including DNS poisoning, embedded objects, and zero-hour attacks.

P. Prakash, Manish Kumar et al (2010) **[4]** published a novel that said phishing remained an effective online deception tactic, despite existing solutions like URL.blacklisting. These solutions were vulnerable because they relied on exact matches with blacklisted entries, making it easy for attackers to evade detection through minor URL alterations. To address this, the Phish Net system utilized two key components. First, it employed five heuristics to identify simple variations of known phishing sites, uncovering new phishing URLs. Second, it employed an approximate matching algorithm that dissected URLs into components, individually checking them against the blacklist. In real-time blacklist feed evaluations, Phish Net detected approximately 18,000 new phishing URLs among 6,000 new blacklist entries. Furthermore, the approximate matching algorithm demonstrated a low false positive rate of 3% and a low false negative rate of 5%.

Diksha Goel, Ankit Kumar Jain et al (2018) **[5]** addressed the issue of mobile phishing, which involved attackers attempting to steal personal information from users, leading to financial losses for individuals and organizations. With the increasing use of smartphones, these devices had become a target for attackers. Detecting mobile phishing attacks had been challenging due to the differences in architecture compared to desktop computers. The paper aimed to provide a comprehensive analysis of mobile phishing, including attack techniques, defence mechanisms, and challenges. It discussed the history, motivations of attackers, and security concerns related to smartphones. Additionally, it presented a taxonomy of mobile phishing attacks and solutions proposed by researchers. The paper also highlighted the challenges faced in addressing mobile phishing and discussed evaluation methods used by researchers in this field.

Dou, Z., Khalil, I., Khreishah, et al (2017) **[6]** published a novel that said the number of unique phishing websites had been growing significantly, with an average annual growth rate of 36.29% over the past six years and 97.36% over the past two years, prompting increased interest in combating phishing attacks within the cybersecurity community. This had led to extensive research and development efforts focused on detecting phishing attempts using various content, network, and URL characteristics. These approaches varied in their underlying concepts, data analysis methods, and evaluation criteria. To address this diversity, this paper conducted a systematic study of phishing detection methods, particularly software-based ones. It covered aspects such as phishing detection taxonomy, evaluation datasets, detection features, techniques, and metrics. The goal was to provide valuable insights that could guide the development of more effective and efficient phishing detection strategies.

Sheng, S., Wardman, B., Warner et al (2009) **[7]** studied the effectiveness of phishing blacklists by conducting tests on eight anti-phishing toolbars using 191 recently created phishing websites. The study revealed that a significant portion of phishing campaigns (63%) lasted less than two hours. Initially, blacklists proved ineffective, catching less than 20% of phishing sites immediately. Blacklists also exhibited variations in update speed and coverage, with 47% to 83% of phishing sites appearing on blacklists within 12 hours. However, toolbars using heuristics in addition to blacklists performed better in the initial detection of phishing sites. The study also tested the toolbars for false positives on legitimate URLs, finding no mislabelling. The paper concluded with suggestions for enhancing anti-phishing tools based on these findings.

Rao, R.S., Pais, A.R et al (2019) **[8]** introduced a novel approach to detect phishing sites by using visual similarity-based techniques. Traditional methods relied on whitelists and had limitations when dealing with non-whitelisted legitimate domains or manipulated whitelisted content. The proposed approach utilized a lightweight visual similarity-based blacklist for initial filtering and incorporated heuristic mechanisms for non-blacklisted phishing sites. It employed various similarity measures and heuristic features to identify phishing sites. The results indicated that the blacklist filter could detect a significant portion of phishing sites, and an ensemble model achieved high accuracy and effectiveness, outperforming existing anti-phishing techniques. This approach was deployed as a Chrome extension called Black Phish, offering real-time client-side protection against phishing sites.

Jain, A.K., Gupta, B.B et al (2019) **[9]** had proposed a novel for a javascript logics based approach for phishing detection using hyperlink information. They had introduced a novel approach for detecting phishing attacks by analyzing hyperlinks within the HTML source code of websites. The approach had incorporated a wide range of specific hyperlink features, categorized into 12 groups, to use algorithms for detection. Performance evaluation had been conducted using phishing and non-phishing website datasets with various classification algorithms. Importantly, this approach had been entirely client-side, eliminating the need for third-party services, and had been language-independent, capable of detecting websites in any textual language. The proposed method had outperformed others with a high accuracy rate, achieving over 98.4% accuracy with the logistic regression classifier.

Taha Aahas and Asha Tariff et al (2023) **[10]** published an article on "End-Users' Perception Of cyber-crimes Towards E-Banking Adoption and Retention." It stated that the evolution of information and communication technology (ICT) had brought about significant transformations in various global business sectors. These changes had simultaneously introduced both convenience and challenges for users and stakeholders alike. One of the key concerns arising from these technological advances had been the rise of cybercrime. This research focused on assessing the impact of cybercrime on the adoption of electronic banking (e-banking) services in Pakistan. Specifically, it sought to explore how hacking, identity theft, and phishing influenced the acceptance of e-banking in Pakistan. Data for this study had been collected from 384 banking customers through structured questionnaires, encompassing demographic inquiries as well as questions related to the study's core constructs. The study's findings had indicated that hacking, phishing, and identity theft had a negative impact on the adoption of e-banking services in Pakistan. The results were discussed in the context of existing literature, and the study also addressed its limitations and potential areas for future research.

Anchal Jain and Vineet Richaraya et al (2011) **[11]** proposed a novel on "Implementing a Web Browser with Phishing Detection Techniques." The novel noted that phishing represents a fusion of social manipulation and technical exploits, artfully designed to coax a target into revealing personal information, typically for the attacker's financial gain. It had become the most prevalent tactic employed by cybercriminals on the internet, with these phishing attacks increasing both in frequency and complexity. Falling victim to phishing could have severe repercussions, potentially leading to identity theft and financial losses. Online banking and e-commerce users, in particular, were increasingly susceptible to phishing scams. In their research, they introduced an innovative approach to detect phishing attacks, involving the development of a prototype web browser acting as an agent to systematically evaluate incoming for signs of phishing attacks. Analyzing URL data collected over a period, they demonstrated that their approach was more effective in detecting phishing attacks compared to existing methods.

Chuan Yue and Haining Wang *et al* (2008) **[12]** published novel based upon: -

The novel highlighted that while many current anti-phishing methods primarily focused on aiding users in verifying website authenticity, research on usability had shown that prevention-focused strategies alone were insufficient in effectively thwarting phishing attacks and protecting internet users from disclosing their credentials to phishing sites. In this study, rather than solely aiming to prevent users from falling into phishing traps, the authors proposed a novel approach to counter phishing attacks, termed 'deceptive bait.' They introduced Bogus Biter, a unique client-side anti-phishing tool that discreetly injected a substantial number of false credentials into suspected phishing websites. Bogus Biter concealed a victim's actual credentials among a pool of fake ones and allowed legitimate websites to promptly identify stolen credentials. By using client-side automatic phishing detection techniques, Bogus Biter complemented existing N preventative anti-phishing methods. The authors implemented Bogus Biter as an extension for the Firefox 2 web browser and assessed its effectiveness through real-world experiments on both phishing and legitimate websites. Their experimental findings suggested that Bogus Biter held promise as a transparent defense against phishing attacks.

NV Rajesh Kumar and S. Vinod Kumar et al (2015) **[13]** proposed an article titled "An effective method of phishing detection using IBC." The article emphasized that phishing, one of the growing forms of online theft, had surged due to the increasing number of internet users. In the past, various tactics such as DNS Spoofing and Chat rooms had been employed to deceive individuals and steal their personal information, posing a significant threat to the economy and necessitating the development of effective solutions. While alternative methods like Blacklist, Whitelist, and Hybrid approaches existed, the authors introduced a novel solution called "Email Detection using Classification with an Intelligent Approach" to combat this issue. Their method employed Intelligent-Based Classification (IBC) to differentiate between phishing and legitimate URL by analysing dataset and URL features. The results demonstrated that their proposed approach surpassed previous methods, and they suggested that further improvements could be achieved by strengthening the rules for IBC. In the future, they aimed to integrate machine learning algorithms to enhance the results further.

2.5 Summary:

In this study, a functional Google Chrome extension has been created to serve as an active monitor for identifying phishing websites and email scams in real-time. The detection of phishing attempts involves an examination of different characteristics, including attributes related to URLs, domains, webpage content, and page structure in URL. Additionally, this extension can be integrated with the application.

2.6.1 Limitation Table

Ref No.	*Author*	*Methodology*	*Limitations*
3	*Jain, A.K., Gupta, B.B., et al (2016)*	*Protecting against phishing attacks at the client side using an auto-updated whitelist*	*The high accuracy rate mentioned (over 98.4%) may indicate potential overfitting to the training data. The model might not generalize well to unseen data or new phishing techniques.*
4	*P. Prakash, Manish Kumar et al -2010*	*Aimed to address the shortcomings of existing URL blacklisting solutions for phishing detection.*	*The paper does not elaborate on how often the blacklist is updated and maintained. Regularly updating the blacklist with the latest phishing URLs is essential to keep the system effective*
5	*Diksha Goel, Ankit Kumar Jain et al (2018)*	*Aimed to comprehensively address the issue of mobile phishing, which involves attackers attempting to steal personal information from users through smartphones*	*Mobile phishing attacks are continually evolving, with attackers developing new and sophisticated techniques. The paper may not cover the most recent and emerging threats, making it important for readers to stay updated on the latest developments.*
6	*Dou, Z., Khalil, I., Khreishah, et al -2017*	*Phishing detection methods in response to the growing number of unique phishing websites and the increased interest in combating phishing attacks within the cybersecurity community.*	*The statistics and trends presented in the paper are based on historical data up to 2017. The phishing landscape is constantly evolving, and the growth rates and trends may have changed since then.*
7	*Sheng, S., Wardman, B., Warner et al (2009)*	*Assessing the effectiveness of phishing blacklists and antiphishing toolbars.*	*The study is based on data and tests conducted in 2009. The phishing landscape and anti-phishing technologies have evolved significantly since then. Therefore, the findings may not accurately reflect the current state of phishing detection.*

	Rao, R.S., Pais, A.R et al -2019	Introduced a novel approach to detect phising sites using visual similarity based techniques.	The computational resources required to perform visual similarity analysis can vary based on the complexity of the web page and the size of the blacklist. The paper may not thoroughly address potential resource limitations.
8	Jain, A.K., Gupta, B.B et al (2019)	Machine learning-based approach for phishing detection using hyperlink information. Their approach involved the analysis of hyperlinks within the HTML source code of websites, and it had several distinctive features.	The high accuracy rate mentioned (over 98.4%) may indicate potential overfitting to the training data. The model might not generalize well to unseen data or new phishing techniques that have emerged since the study, as the phishing landscape is continually evolving.
9	Taha Aahas and Asha Tariff et al -2023	Assessing the impact of cybercrime on the adoption of electronic banking (ebanking) services in Pakistan.	The study collected data from 384 banking customers in Pakistan. While this can provide useful insights, the size and demographic representation of the sample may not fully capture the diverse user base of e-banking in the country. Generalizing findings to a broader population might be limited.
10	Anchal Jain and Vineet Richaraya et al (2011)	Involved the development of a prototype web browser that systematically evaluated incoming emails for signs of phishing attacks.	The study's findings and the proposed method may be specific to the phishing threats and techniques prevalent at the time of the research. Phishing attacks continually evolve, and the effectiveness of the approach against newer and more sophisticated attacks might not have been assessed.
11	Chuan Yue and Haining Wang et al -2008	Focused on countering phishing attacks with a novel approach termed "deceptive bait" and the implementation of an anti-phishing tool called Bogus Biter.	The method involves interacting with potentially malicious websites, which raises ethical considerations. The paper may not thoroughly address the ethical implications and potential legal concerns of actively engaging with phishing sites.
12	NV Rajesh Kumar and S. Vinod Kumar et al (2015)	The introduction of a novel solution for detecting phishing emails using an approach called "Email Detection using Classification with an Intelligent Approach" based on Intelligent-Based	The article's findings and the proposed IBC method may be specific to the phishing techniques and features prevalent at the time of the research. Phishing attacks evolve, and the effectiveness of the approach against newer and more sophisticated attacks might not have been assessed.

CHAPTER 3

METHODOLOGY

3.1 Project Overview

This advanced AI-driven system is designed to provide robust protection against phishing threats by identifying potentially harmful URLs in real-time. A Chrome extension continuously monitors websites as users browse, analyzing URL patterns for signs of phishing. If suspicious activity is detected, a pop-up alert appears, enabling users to make informed decisions about their online safety. The system's javascript algorithms are trained on extensive data sets, allowing it to adapt to new threats and maintain high detection accuracy. In addition to real-time detection, there's a centralized data management system to store detection outcomes securely. Each record contains user ID, prediction result, and timestamp, facilitating detailed trend analysis and security insights. To enhance the experience, an Android app provides a user-friendly interface for visualizing browsing activity, with clear graphical representations of phishing detection trends. This integrated approach—combining real-time monitoring, secure data storage, and mobile accessibility—offers comprehensive protection against phishing attacks

3.1.1 Significance

Phishing has emerged as one of the most pervasive cybersecurity threats, targeting individuals and organizations alike. This form of cyberattack uses deceptive tactics to trick users into divulging sensitive information, such as passwords, credit card numbers, or social security numbers, often through emails, messages, or fake websites designed to look legitimate. Phishing attacks are highly effective due to their ability to mimic trusted entities, making it challenging for even tech-savvy individuals to identify them.

The consequences of falling victim to phishing can be severe, leading to identity theft, financial loss, unauthorized access to sensitive data, and compromised business operations. As cybercriminals continue to develop more sophisticated techniques, the number of phishing attacks is expected to grow, posing an ever-greater risk to online users.

Developing a tool that can accurately detect phishing attempts in real-time is crucial in combating this threat. Real-time detection allows users to receive immediate alerts about suspicious URLs, providing them with the information needed to avoid potentially harmful interactions. By catching phishing attempts early, this tool can prevent users from inadvertently sharing sensitive information, thereby reducing the likelihood of identity theft and other forms of cybercrime.

Moreover, real-time phishing detection contributes to the overall security of online communications. When users are alerted to phishing risks, they become more cautious and better educated about safe online practices. This heightened awareness can lead to a more security-conscious digital environment, where individuals and organizations are less likely to fall prey to cybercriminals.

In summary, by providing a robust and accurate real-time phishing detection tool, we can significantly reduce the risk of phishing attacks, safeguard sensitive information, and foster a more secure online ecosystem. This not only protects individuals and organizations from the immediate impacts of phishing but also contributes to a broader culture of cybersecurity awareness.

3.1.2 Project Purpose

The purpose of this project is to develop a robust, AI-driven tool designed to detect phishing attempts in real-time. This tool aims to reduce the risk of phishing attacks by providing users with immediate alerts about potentially malicious URLs while browsing the internet. By enabling quick identification and response to phishing threats, this tool will help individuals and organizations protect sensitive information and reduce the likelihood of identity theft and financial loss.

3.2 Methodology

3.2.1 Phishing Detection in Chrome Extension

The core functionality of the Chrome extension is to detect phishing attempts based on various features extracted from the URL and the webpage content. This process involves several steps:

- **Feature Extraction:**
  Below are the features used in the prediction logic, along with their interpretation:
  - isIPInURL(): Checks if the URL contains an IP address. Phishing sites often use IP addresses to hide their true domain.
  - isLongURL(): Determines if the URL is unusually long. Long URLs can indicate attempts to obfuscate phishing links.
  - isTinyURL(): Checks if the URL is extremely short, indicating the use of URL shorteners which are common in phishing.
  - isAlphaNumericURL(): Checks for the presence of the "@" symbol, which can indicate attempts to obfuscate URLs.
  - isRedirectingURL(): Detects if the URL uses multiple slashes to redirect to other sites.
  - isHypenURL(): Checks if the URL contains hyphens, which can be used to mimic legitimate domain names.
  - isMultiDomainURL(): Checks if the URL has multiple subdomains, which can be a sign of phishing.
  - isFaviconDomainUnidentical(): Compares the favicon domain with the main domain. Mismatches can indicate phishing.
  - isIllegalHttpsURL(): Checks if the URL does not use HTTPS properly.
  - isImgFromDifferentDomain(): Determines if images are sourced from different domains, which may indicate external phishing content.
  - isAnchorFromDifferentDomain(): Checks if anchor tags (<a>) lead to different domains, suggesting external links to phishing sites.
  - isScLnkFromDifferentDomain(): Evaluates if script or link tags source content from different domains.

  - isFormActionInvalid(): Examines if form actions are invalid, which can suggest phishing data collection.
  - isMailToAvailable(): Checks if there are "mailto:" links, which can be used in phishing scams.
  - isStatusBarTampered(): Checks for manipulations of the status bar, a common phishing technique.
  - isIframePresent(): Determines if iframes are present, which can be used to hide malicious content

- **Prediction Algorithm:**
  - It takes two arguments: data and weight.
  - Data is likely an array of numerical features representing various attributes or characteristics of a website or a context.
  - Weight is an array of predefined values, representing the importance or coefficients for each feature.
  - Formula Used for Prediction
  - In the predict function, a basic linear combination of the data and weight arrays is used to calculate a score f:
  - For each index j, f is incremented by the product of the corresponding elements from data and weight.
  - Essentially, it performs: f += data[j] * weight[j].
  - Once f is calculated, the function returns 1 or -1 based on whether f is greater than 0 or not. This represents a binary classification:
  - If f > 0, it returns 1, possibly indicating a "positive" result (like identifying a phishing site).
  - If f <= 0, it returns -1, suggesting a "negative" result (like indicating a non-phishing site)
- **User Notification and Data Storage:** Based on the prediction result, the Chrome extension sends a pop-up alert to inform the user of the potential risk. Additionally, the result, along with the URL and user ID, is sent to a backend server for data logging and further analysis. The server endpoint '/api/query' is used for this data insertion, allowing us to track phishing patterns and provide visualizations of the collected data.

3.2.2 Server-Side Communication and Database Interaction

The server-side application built with Node.js serves multiple purposes, including storing user data, executing Python scripts, and communicating with the Chrome extension. Key functionalities include:

- **Node.js Server:** The server is set up with Node.js, using middleware such as CORS and' body-parser ' to handle cross-origin requests and JSON payloads. The server listens for incoming HTTP requests on a specified port and includes error handling mechanisms to ensure robustness.
- **Oracle Database Configuration:** The server connects to an Oracle Database using the 'oracledb' package. It establishes a connection to the database, allowing data insertion and transaction commitment. This connection setup ensures proper error handling and recovery in case of failures.
- **API Endpoints:** Several API endpoints are implemented to handle different operations:
  - POST /api/query: Handles data insertion into the Oracle Database based on POST requests from the Chrome extension. This endpoint is used to store phishing detection results and other related information.
  - POST /api/extension-data: Processes requests from the Chrome extension, including actions such as running a Python script, handling login scenarios, and managing phishing detection results.
  - GET /api/get-data: Sends real-time data back to the Chrome extension, allowing it to fetch necessary information like user session status.
- **Python Script Execution:** The server provides an endpoint for executing Python scripts. When the action 'execute_python' is received via the ' /api/extension-data ' endpoint, the server runs the specified Python script using the 'exec' function from the 'child_process' module. The server captures the output and returns it to the client. This functionality is used to automate tasks like user authentication and backend operations.

3.2.3 Android App Integration

The Android app serves as a mobile interface, allowing users to manage their accounts, view phishing detection results, and interact with the backend server. The app is developed using Android Studio, and its GUI is designed with Figma. The integration process involves:

- **Development Environment:** Android Studio is used to build the app. It provides tools for developing, debugging, and testing Android applications. The app communicates with the backend server to fetch user data and display phishing-related information.
- **User Interface Design:** Figma is used to design the GUI of the app. The design includes user-friendly elements like buttons, input fields, and interactive components to ensure an intuitive user experience.
- **API Integration:** The app communicates with the Node.js backend through HTTP requests. It retrieves user data, phishing detection results, and other relevant information. The server responses are used to populate the app's user interface and provide real-time feedback to users.
- **User Management:** The app allows users to manage their accounts, including login, logout, and session management. It can send requests to the backend server to execute specific actions, such as fetching user data or triggering Python script execution.

3.2.4 Communication and Data Handling

Communication between the Chrome extension, Node.js server, Oracle Database, and Android app is a critical aspect of the project's methodology. Here's how data flows among these components:

- **Chrome Extension to Node.js Server:** The Chrome extension sends phishing detection results to the server for storage and analysis. It also sends requests to execute Python scripts and manage user sessions.
- **Node.js Server to Oracle Database:** The server interacts with the Oracle Database to insert and retrieve data. This data includes phishing detection results, user information, and session details.
- **Node.js Server to Android App:** The server provides the Android app with user data and phishing detection results. The app can send requests to the server to trigger specific actions, such as logging in, fetching data, or executing Python scripts.

- **Oracle Database Interaction:** The server connects to the Oracle Database to manage data insertion and retrieval. It ensures that all data operations are reliable and include proper error handling.

3.2.5 Error Handling and Response

To maintain system robustness, error handling is implemented at various stages of the project:

- **Chrome Extension:** The extension includes error handling to manage potential issues with phishing detection and communication with the backend server.
- **Node.js Server:** Error handling is implemented for database interactions, Python script execution, and communication with the Chrome extension and Android app. If errors occur, appropriate responses are returned to the client for debugging and problem resolution.
- **Oracle Database:** The server ensures reliable database operations with proper error handling during data insertion and transaction commitment.

3.2.6 Summary for Methodology

This comprehensive phishing detection system integrates a Chrome extension for real-time URL monitoring, an Node.js server for backend processing and OracleDB interaction, and an Android app for user management and mobile interface. The methodology incorporates phishing detection, backend communication, data handling, error handling, and Android app integration. By combining these elements, the project provides a robust and scalable solution for detecting phishing risks and ensuring user safety while collecting valuable data for ongoing analysis and improvement.

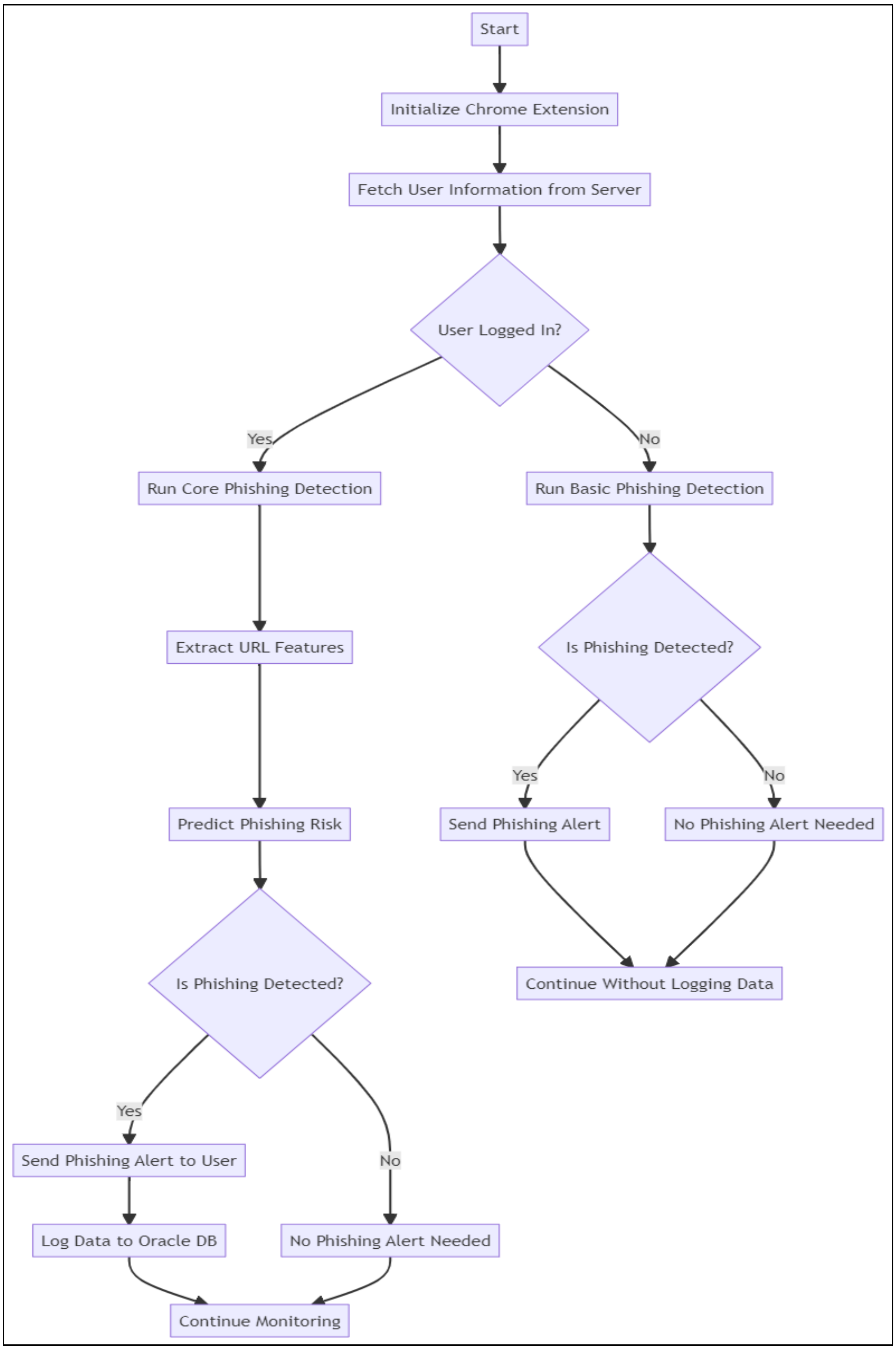


Figure 3.1 Data Flow

CHAPTER 4

IMPLEMENTATION

4.1 Software Development Methodology

The choice of the right software development methodology will be vital for the successful execution of this project. We will opt for the Agile methodology, known for its iterative and collaborative approach. Agile will provide the flexibility and adaptability required to navigate evolving requirements and challenges effectively. It will promote frequent communication, short development cycles, and continuous testing.

4.2 Technology Stack

The technology stack will be the foundation of our project. Each component will be carefully selected to ensure efficiency, security, and scalability.

a. Programming Languages:

- Python

- JavaScript: For developing the Chrome extension.

b. Libraries and Frameworks:

- Chrome Extension API: For integration with the browser.

- Flask: For building the backend API.

c. Database:

- OracleDB: database chosen for its flexibility and scalability in handling large datasets.

4.3 Project Milestones

To manage the complexity of this project, we will establish well-defined milestones that will guide our future implementation process:

a. Data Collection and Preparation:

- We will collect a diverse dataset of phishing and legitimate URLs.

- Data will be preprocessed and cleansed to remove inconsistencies.

- The dataset will be prepared

c. Chrome Extension Development:

- Our Chrome extension will be at the forefront of user interaction with the tool.

- We will design the user interface for an intuitive user experience. It will offer a seamless integration into the browser while being unobtrusive.

- The extension will process URLs in real-time, analyzing their safety and providing instant feedback to users.

- User interaction will be facilitated through components such as pop-ups alerts

- The extension will be designed to be lightweight, minimizing resource consumption and avoiding any negative impact on browsing performance.

d. Integration with Kavach - The Shield:

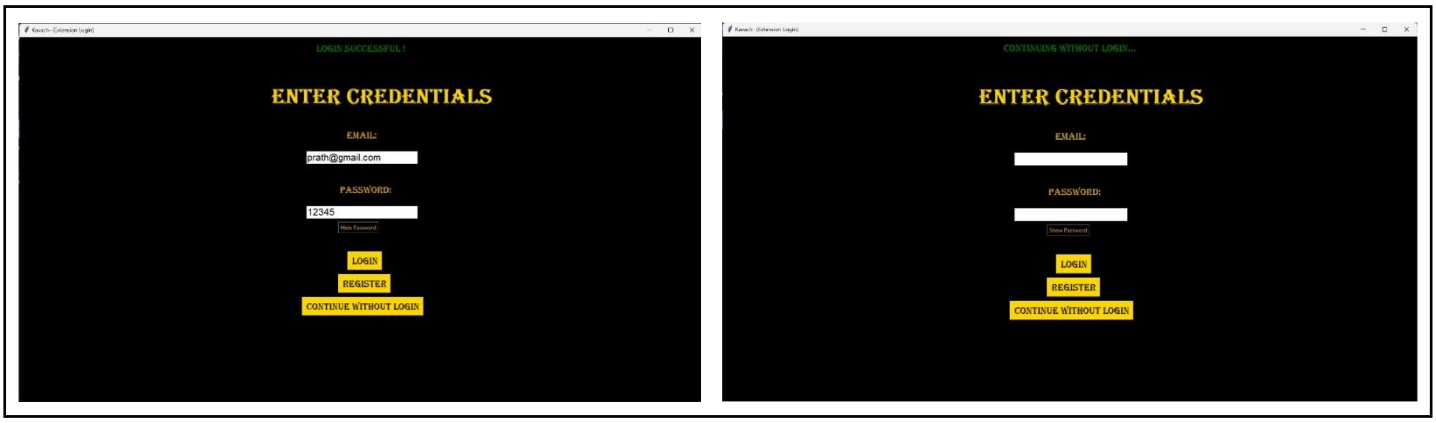
- Seamless integration with Kavach will be a critical component of the project, and every detail will be meticulously addressed.

- We will design a robust and secure API for communication between the extension and Kavach, following industry best practices.

- Real-time data synchronization will ensure that Kavach receives continuous updates on detected threats.

4.4 Real Time Images And GUI

1.As the extension is loaded the first thing it will ask you for login.you can either login or continue without login. Also for new users it will ask them to register.



Kavach- (Extension Login)

LOGIN SUCCESSFUL !

ENTER CREDENTIALS

EMAIL: prath@gmail.com

PASSWORD: 12345

Kavach- (Extension Login)

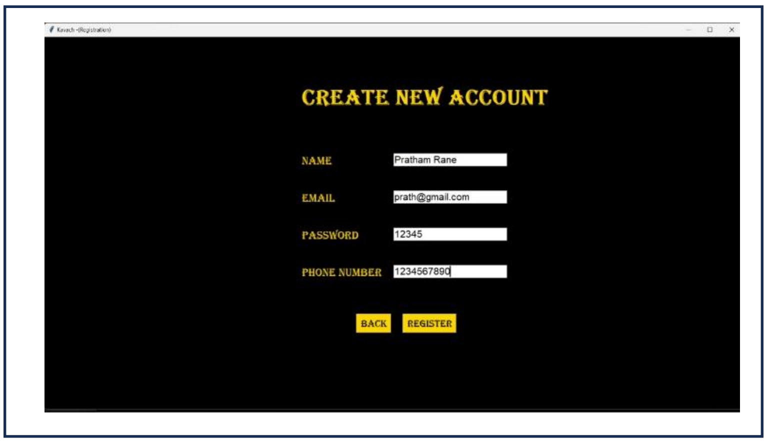
CONTINUING WITHOUT LOGIN...

ENTER CREDENTIALS

EMAIL:

PASSWORD:

Figure 4.1 Login Window



Kavach -(Registration)

CREATE NEW ACCOUNT

NAME: Pratham Rane

EMAIL: prath@gmail.com

PASSWORD: 12345

PHONE NUMBER: 1234567890

Figure 4.2 Registration Window

2.After logging, you can surf through browser the extension will do the work properly. If the user has logged in the Data will be stored in the Oracle server else it will not be saved.

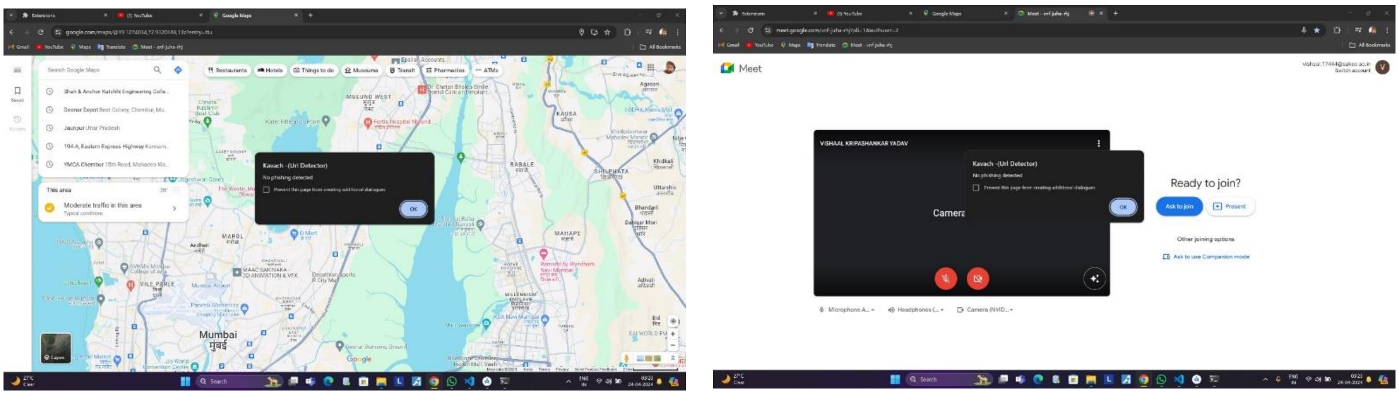


Figure 4.3 No Phishing detected Prediction

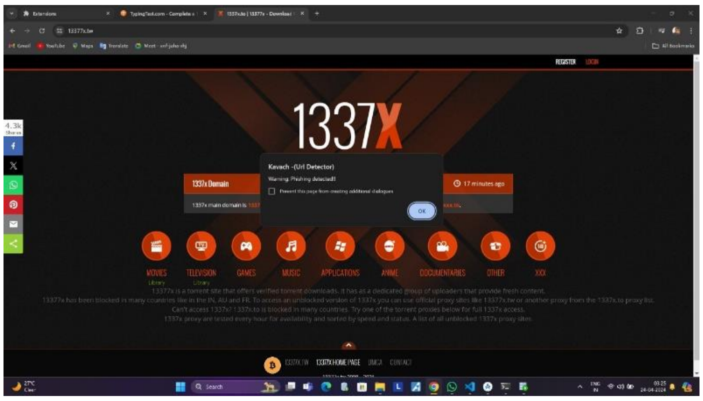
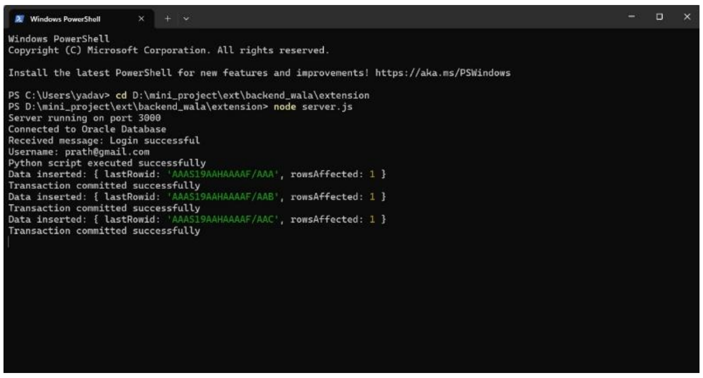

1337X
Kavach -(Url Detector)
Warning: Phishing detected!
Prevent this page from creating additional dialogues
OK

Figure 4.4 Phishing detected Prediction

3.The Node.js Server helps to append the data into the database.



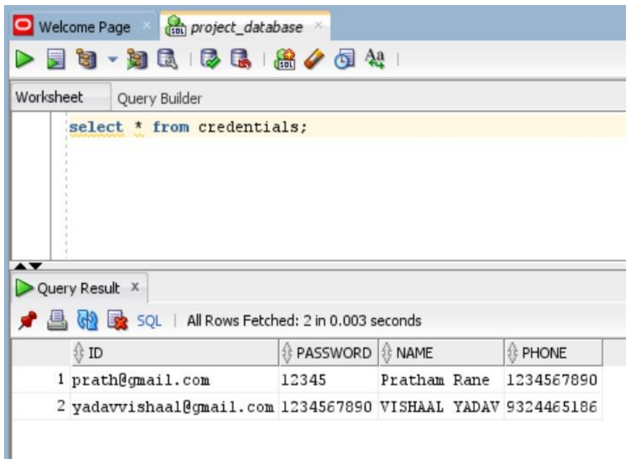
```
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\yadav> cd D:\mini_project\ext\backend_wala\extension
PS D:\mini_project\ext\backend_wala\extension> node server.js
Server running on port 3000
Connected to Oracle Database
Received message: Login successful
Username: prath@gmail.com
Python script executed successfully
Data inserted: { lastRowid: 'AAAS19AAHAAAAF/AAA', rowsAffected: 1 }
Transaction committed successfully
Data inserted: { lastRowid: 'AAAS19AAHAAAAF/AAB', rowsAffected: 1 }
Transaction committed successfully
Data inserted: { lastRowid: 'AAAS19AAHAAAAF/AAC', rowsAffected: 1 }
Transaction committed successfully
```

Figure 4.5 The Server Connection

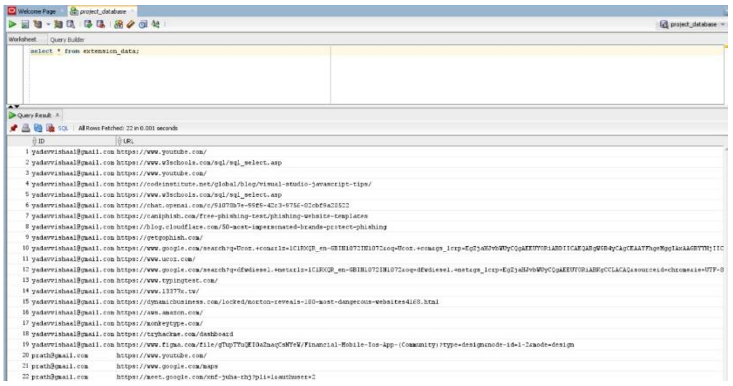
4.In the database the login Credentials and the phishing prediction is stored along with the username and visited URL.



```
select * from credentials;
```

All Rows Fetched: 2 in 0.003 seconds

	ID	PASSWORD	NAME	PHONE
1	prath@gmail.com	12345	Pratham Rane	1234567890
2	yadavvishaal@gmail.com	1234567890	VISHAAL YADAV	9324465186



```
select * from extension_data;
```

All Rows Fetched: 22 in 0.001 seconds

	ID	URL
1	yadavvishaal@gmail.com	https://www.youtube.com/
2	yadavvishaal@gmail.com	https://www.w3schools.com/sql/sql_select.asp
3	yadavvishaal@gmail.com	https://www.youtube.com/
4	yadavvishaal@gmail.com	https://codeinstitute.net/global/blog/visual-studio-javascript-tips/
5	yadavvishaal@gmail.com	https://www.w3schools.com/sql/sql_select.asp
6	yadavvishaal@gmail.com	https://chat.openai.com/c/51078b7e-59f5-42c3-9756-02cbf9a20522
7	yadavvishaal@gmail.com	https://caniphish.com/free-phishing-test/phishing-website-templates
8	yadavvishaal@gmail.com	https://blog.cloudflare.com/50-most-impersonated-brands-protect-phishing
9	yadavvishaal@gmail.com	https://getgophish.com/
10	yadavvishaal@gmail.com	https://www.google.com/search?q=Ucoz.+com&rlz=1C1RXQR_en-GBIN1072IN1072&oq=Ucoz.+com&gs_lcrp=EgZjaHJvbWUyCQgAEEUYORiABDIICAEQABgWGB4yCAgCEAAYFhgeMggIAxAAGBYYHjIIC
11	yadavvishaal@gmail.com	https://www.ucoz.com/
12	yadavvishaal@gmail.com	https://www.google.com/search?q=dfwdiesel.+net&rlz=1C1RXQR_en-GBIN1072IN1072&oq=dfwdiesel.+net&gs_lcrp=EgZjaHJvbWUyCQgAEEUYORiABKgCCLACAQ&sourceid=chrome&ie=UTF-8
13	yadavvishaal@gmail.com	https://www.typingtest.com/
14	yadavvishaal@gmail.com	https://www.13377x.tw/
15	yadavvishaal@gmail.com	https://dynamicbusiness.com/locked/norton-reveals-100-most-dangerous-websites4168.html
16	yadavvishaal@gmail.com	https://aws.amazon.com/
17	yadavvishaal@gmail.com	https://monkeytype.com/
18	yadavvishaal@gmail.com	https://tryhackme.com/dashboard
19	yadavvishaal@gmail.com	https://www.figma.com/file/gTupTYuQEIOa2maqCsNYeW/Financial-Mobile-Ios-App-(Community)?type=design&node-id=1-2&mode=design
20	prath@gmail.com	https://www.youtube.com/
21	prath@gmail.com	https://www.google.com/maps
22	prath@gmail.com	https://meet.google.com/xnf-juha-rhj?pli=1&authuser=2

Figure 4.6 Data Base for Credentials And Prediction Entries

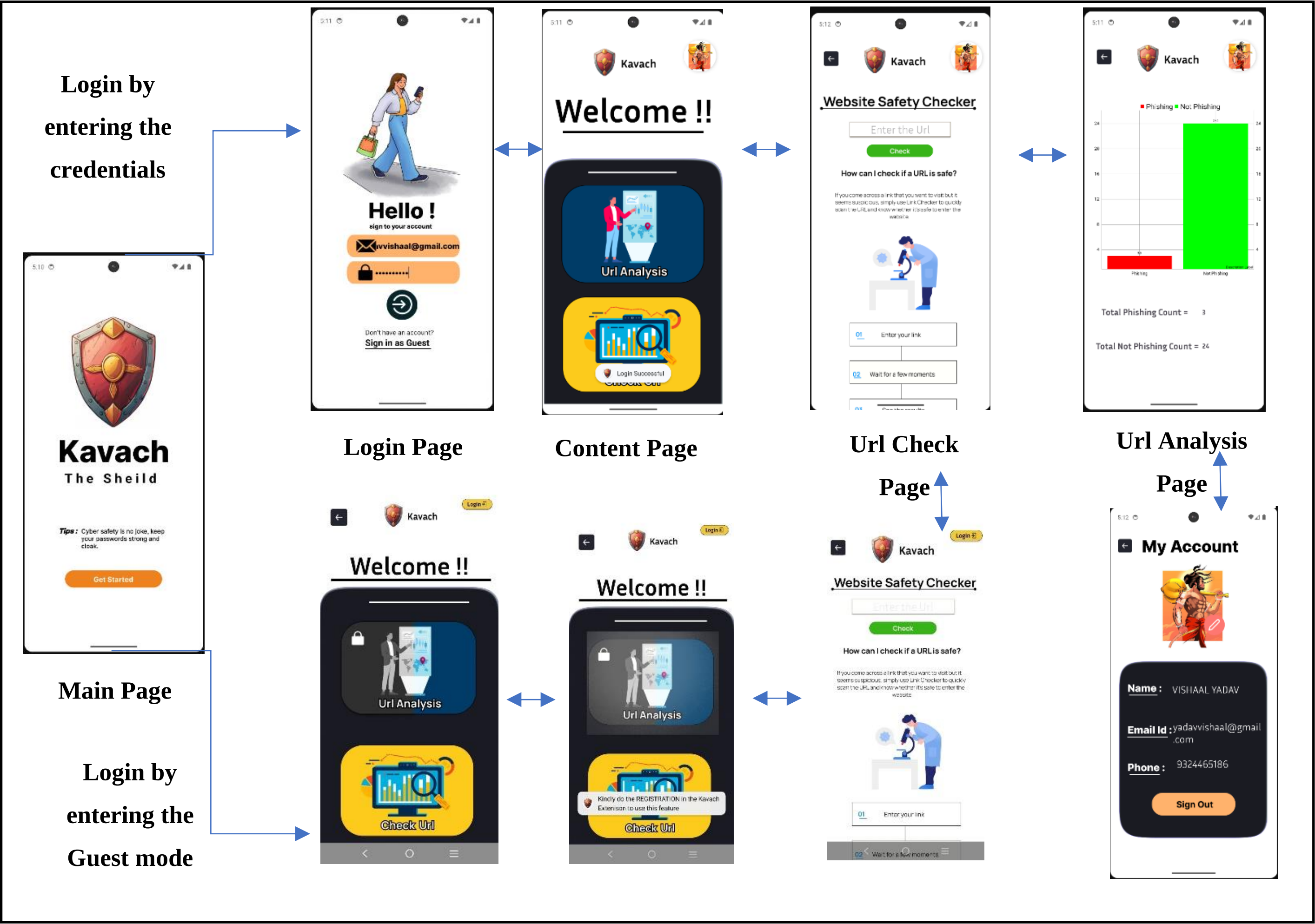


Figure 4.7 UI of Application

4.5 Monitoring and Maintenance

Ongoing monitoring and maintenance will be essential to the tool's continued effectiveness:

- Threat database updates will be regularly performed to update the database with new phishing patterns and malicious URLs.

- Performance monitoring will ensure that the tool's performance is continuously monitored, and any deviations are addressed promptly.

4.6 Scalability and Future Enhancements

Scalability will be an integral part of the project's design:

- Cloud-based infrastructure will allow seamless scalability as the user base grows.

- The project will be designed with the flexibility to accommodate future enhancements, such as additional threat detection capabilities or support for more platforms.

CHAPTER 5

CONCLUSION

The AI-driven phishing detection system in this project combines a Chrome extension for real-time monitoring, an Node.js backend for data management, an Oracle Database for persistent storage, and an Android app for user interaction, creating a robust and comprehensive framework to combat phishing threats. The Chrome extension continuously analyzes URLs using feature extraction algorithm to detect phishing attempts, alerting users to potential risks. The Node.js backend serves as a central hub, processing and storing data in an Oracle Database, and also supports additional functionalities like Python script execution. The Android app, designed in Figma and developed with Android Studio, extends the system's reach to mobile users, allowing them to manage accounts and access phishing-related data on the go. Together, these components form an integrated approach that not only provides real-time protection but also facilitates secure data storage and multi-platform user engagement, offering a comprehensive solution to the growing problem of online phishing threats.

5.1 Key Achievements:

- **Real-Time Phishing Detection:** The Chrome extension offers immediate feedback on potential phishing threats, alerting users to malicious URLs while they browse the internet.
- **Backend Integration:** The Node.js server, connected to the Oracle Database, allows seamless data storage and retrieval, supporting further analysis of phishing patterns.
- **User Management and Interaction:** The Android app, developed with Android Studio and designed using Figma, provides a user-friendly interface for managing accounts and viewing phishing-related data. This app extends the reach of the system to mobile devices, enhancing user engagement.
- **Scalability and Flexibility:** The architecture's modular design allows for easy extension and scalability, accommodating future enhancements and additional features.

5.2 Project Summary:

The project aimed to create a system that could detect phishing attempts in real-time, store relevant data for analysis, and offer a user-friendly mobile interface. The Chrome extension performs the initial detection by analyzing URLs and applying a predictive model. The results are sent to the Node.js backend, where they are stored in an Oracle Database. This backend also facilitates additional operations, such as running Python scripts and managing user sessions.

The integration with an Android app allows users to interact with the system from their mobile devices, offering greater flexibility and accessibility. This cross-platform approach ensures that the phishing detection system is effective and accessible across different devices and platforms.

Overall, the project demonstrates a practical solution to a growing cybersecurity threat. By combining real-time detection, backend data management, and mobile integration, the system provides a comprehensive tool for identifying and mitigating phishing risks, contributing to a safer online environment for users.

REFERENCES

[1]Alkhalil Z, Hewage C, Nawaf L and Khan I Phishing Attacks: A Recent Comprehensive Study and a New Anatomy. Front. Comput. Sci. 3:563060. doi: 10.3389/fcomp.2021.563060, 2021.

[2]Tosin Ige et al,"Deep Learning-Based Speech and Vision Synthesis to Improve Phishing Attack Detection through a Multi-layer Adaptive Framework,"link.springer.com/article/10.1007/s11235-020-00733-2

[3]Jain, A.K., Gupta, B.B. et al, " Fighting against phishing attacks: state of the art and future challenges", Springer Access DOI: https://doi.org/10.1007/s00521-016-2275-y, 2016.

[4]Prakash, Manish Kumar et al," PhishNet: Predictive Blacklisting to Detect Phishing Attacks"IEEE Access,DOI: https://doi.org/10.1109/INFCOM.2010.5462216, 2010.

[5]Diksha Goel, Ankit Kumar Jain et al," Mobile phishing attacks and defence mechanisms: State of art and open research challenges",ScienceDirect Access,DOI: https://doi.org/10.1016/j.cose.2017.12.006, 2018.

[6]Dou, Z., Khalil, I., Khreishah, et al," Systematization of Knowledge (SoK): A Systematic Review of Software-Based Web Phishing Detection",IEEE Access,DOI:
[7]https://doi.org/10.1109/COMST.2017.2752087, 2017.
Sheng, S., Wardman, B., Warner et al," An Empirical Analysis of Phishing
[8]Blacklists",FigShare Access,DOI: https://doi.org/10.1184/R1%2F6469805.V1, 2009.
Rao, 4.S., Pais, A.R et al," Two level filtering mechanism to detect phishing sites using lightweight visual similarity approach",Springer
[9]Access,DOI:https://link.springer.com/article/10.1007/s12652-019-01637-z, 2019.
Jain,5A.K., Gupta, B.B et al," A machine learning based approach for phishing detection using hyperlinks information",ResearchGate Access , DOI: https://link.springer.com/article/10.1007/s12652-018-0798-z, 2015
[10]Anchal Jain and Vineet Richaraya et al," Implementing a Web Browser with Phishing Detection Techniques",Cornell University Access,DOI: https://doi.org/10.48550/arXiv.1110.0360, 2011.

[11]Chuan Yue and Haining Wang et al," Anti-Phishing in Offense and Defense",ResearchGate Access,DOI: https://doi.org/10.1109/ACSAC.2008.32, 2010.
[12]NV Rajesh Kumar and S. Vinod Kumar et al," An effective method of phishing detectionusingIBC",ResearchGateAccess,DOI:https://www.researchgate.net/publication/283101545_An_effective_method_of_phishing_detection_using_IBC, 2015.
[13]Sudhanshu Gautam, Kritika Rani & Bansidhar Joshi et al, ,"Detecting Phishing Websites Using Rule-Based Classification Algorithm: A Comparison",ResearchGate Access, DOI: http://dx.doi.org/10.1007/978-981-10-3932-4_3, 2018.
[14]Ferhat Ozgur Catak,Volkan Dortkardes,Kevser Şahinbaş "Malicious URL Detection and Classification Analysis using Machine Learning",ResearchGate Access,DOI: https://doi.org/10.4018/978-1-7998-5101-1.ch008,2020.