International Journal of Advanced Trends in Multidisciplinary Research
ISSN: 3139-0986, Volume 2 Issue 7
July 2026, PP 1-8

Securing QR Codes Infrastructure Using AI to Detect Malicious Activity

[1]M. Sivaparvathi, [2]D. Sravya, [3]CH. Naveen
[1,2]Asst. Prof, Department of CSE, MAM Women's Engineering College, Narasaraopet, Palnadu, A.P., India.
[3]Assoc. Prof., Department of CSE, MAM Women's Engineering College, Narasaraopet, Palnadu, A.P., India

***Abstract* – Users may now share information quickly and simply with the use of laptops, cellphones, and QR codes. The security of this widely used technology is in jeopardy due to the increasing number of hacks that target QR codes. Users must have trust in the QR Code system for transactions to be secure, regardless of whether they are at a conference, restaurant, or payment station. Since 2020, there has been a 2400% rise in the deployment of QR vector malware assaults, and a 50% increase in phishing methods that use bogus codes. The need to restore integrity safeguards in order to regain the public's confidence is emphasised by these concerning outcomes. In response to these worries, this study employs AI and third-party services to identify QR Code security threats. Here we provide a Secure QR Codes system that employs four ML models, including ones for anomaly detection and risk warning, as well as ones trained on good and poor QR Codes. With the help of encryption engines, APIs, proxy services, AI evaluation modules, and predictions based on AI, the integrated system detected more than 96% of the simulated real-world threats, such as phishing, exfiltration, injection, and code execution. We have developed a method that can detect altered QR codes. As a viable, non-proprietary, web-deployable solution to anticipate and minimise these issues, it helps to rebuild confidence in QR Codes systems in many contexts. This index is associated with many subjects, including safe QR code infrastructure, fast response codes, cybersecurity, and anomaly detection.**

***Keywords* – QR Code Security, Artificial Intelligence (AI), Machine Learning, Deep Learning, Malicious QR Codes, QR Code Phishing (Quishing).**

I. INTRODUCTION

Modern digital communication would be incomplete without Quick Response (QR) Codes, which provide a simple and fast method to save and retrieve data via smartphones and other smart devices. Their widespread use has been very beneficial to several applications, including digital payments, healthcare, transportation, ticketing systems, product identification, and information sharing [1, 2]. The proliferation of QR Code-based assaults, such as phishing, malware distribution, credential theft, and malicious URL redirection, is a testament to the fact that cybercriminals see QR Codes as a target despite their numerous helpful uses [3]-[6].

Quick Response (QR) Codes lack the built-in security features of more conventional authentication techniques, such as encryption, authentication, and integrity checking. As a result, criminals may hack QR Codes and introduce dangerous payloads or malicious URLs, causing users to be sent to bogus websites or prompted to do unauthorised actions when scanned. Secure QR Codes are a major area of research concern [7]-[9]. Users risk losing money, having their privacy invaded, and getting malware because of these security holes.

In an effort to increase confidence in QR Code systems, some researchers have published security protocols. Security measures have been put in place to guarantee the confidentiality and authenticity of QR Codes. These measures include cryptographic protection, authentication processes, digital signatures, and secure QR Code production. Due to the necessity for specialised software, more processing capacity, or distinct verification processes, these solutions aren't always applicable in the real world, despite the fact that they better secure sensitive information [10]-[12].

New approaches for detecting cyber threats have been developed by recent advances in AI and ML, which automatically learn malicious patterns from large datasets. The ability of machine learning algorithms to distinguish between legitimate and counterfeit QR Codes is based on lexical URL features, QR Code characteristics, and behavioural patterns. These smart approaches beat traditional rule-based security systems in tests of scalability, adaptability, and detection accuracy [13]-[15]. In addition to machine learning, safe QR Code infrastructures employ multi-layer security frameworks that combine threat data with external security services, extract features, and identify anomalies. These techniques enhance the ability to identify malicious redirections, phishing, and malware dissemination using QR codes; they reduce the amount of false alarms and increase user confidence [16]-[18].

To combat these issues and protect consumers from cyber threats, this paper introduces an AI-powered Secure QR Codes Infrastructure. The proposed method incorporates lexical feature extraction, anomaly detection, and machine learning models to evaluate the QR Codes' security using VirusTotal and urlscan.io, two third-party security providers. By training and evaluating several machine learning algorithms including XGBoost, LightGBM, Random Forest, and Neural Networks, we may classify QR Codes as either innocuous or hazardous. Experimental results demonstrate that the proposed framework effectively identifies phishing, malware, payload injection, and malevolent redirection attacks [19], and it provides a practical and extensible answer for secure QR Code applications.

International Journal of Advanced Trends in Multidisciplinary Research
ISSN: 3139-0986, Volume 2 Issue 7
July 2026, PP 1-8

Background and Qr Codes Vulnerabilities

Quick Response (QR) Codes are two-dimensional matrix barcodes that customers may use to save and retrieve information fast via mobile device scanning. Quick Response (QR) Codes have become more important in many industries because to their high data storage capacity, fast reading speed, inherent error correction capabilities, and widespread application in digital payment systems, e-commerce, healthcare, transportation, product authentication, event tickets, and information sharing. Their ease of use and low installation cost have led to their widespread adoption in both residential and commercial settings.

Although QR Codes have many advantages, they do not yet have any built-in security features such as encryption, authentication, or integrity checks. This opens the door for criminals to covertly utilise QR Codes to distribute malware or direct users to malicious websites. Due to their striking similarity to legitimate QR Codes, users are naively led to believe that they carry valuable information.

In order to steal sensitive information like login passwords, bank credentials, or other financial data, attackers often utilise QR phishing, also known as Quishing, to insert malicious URLs into QR Codes. Clicking on QR Codes to download malicious programs or executable files is another major concern since it may infect devices and reveal sensitive information. Malware is another major worry.

The unlawful acquisition of sensitive user or device information via the scanning of a malicious QR Code is an example of a data exfiltration attack, another security issue associated with QR Codes. Furthermore, malicious actors may exploit software vulnerabilities or execute unauthorised code by adding scripts or instructions into QR Code content, a method called payload injection. When harmful URL redirections or drive-by download attacks push users to dangerous websites or begin downloading files automatically, it puts them at a higher risk of financial fraud and identity theft.

Another big security risk is that there aren't any verification techniques that show if a QR Code has been modified or not. In order to deceive consumers, criminals may utilise counterfeit QR Codes in public spaces such as restaurants, payment kiosks, advertisements, and transportation networks. Scanners of these phoney QR Codes run the risk of falling prey to fraud or disclosing sensitive information. To combat these concerns, an increasing number of cybersecurity solutions are using AI and ML techniques to identify suspicious QR Codes before the user interacts with them. By analysing the structure, lexical features, URL, and behavioural patterns of a QR Code, artificial intelligence detection systems can reliably ascertain its maliciousness. It is possible to construct a robust framework that protects users from evolving QR Code-based cyber threats while maintaining the technology's use and convenience by combining threat intelligence services, anomaly detection technologies, and secure web apps.

II. RELATED WORKS

Academics are very interested in QR Code security because of the widespread use of QR Codes in many different areas, including digital payments, healthcare, transportation, authentication, and information sharing. Several solutions have been proposed by researchers to strengthen QR Code systems and make them more resistant to threats such as phishing, malicious URLs, infection, and unauthorised access to data.

Some studies have looked at ways to make QR Codes more secure by using cryptographic techniques, digital signatures, and authentication processes. By using these techniques, we can ensure that QR Codes are secure and hard to forge, which in turn increases their anonymity. Solutions like this provide better security, but they aren't always applicable since they often need dedicated verification systems or specialist QR Code readers. Machine learning has recently emerged as a promising alternative for identifying fake QR Codes by uncovering hidden commonalities in legitimate and malicious information. Supervised learning algorithms are able to detect phishing attacks and classify suspicious URLs by analysing lexical features, URL structures, and information included in QR Codes. When it comes to evolving cyber threats, machine learning models outperform traditional rule-based detection methods in terms of accuracy and adaptability.

New research has shown that combining AI methods with external threat intelligence systems is essential for QR Code security. The use of third-party services for malware investigation, URLs, and domains is possible, and machine learning models may automate classification and risk assessment. When used in tandem, they improve the accuracy of real-time threat detection systems and shorten the time it takes to discover malicious QR Codes.

Despite these advancements, many of the existing solutions just deal with certain aspects of QR Code security, including cryptographic protection or hazardous URL detection. Not many systems exist that integrate many layers of protection into a unified framework capable of detecting a wide variety of attack vectors, including phishing, malware distribution, data exfiltration, payload injection, and harmful redirection. Furthermore, due to their reliance on specialist hardware or software, some existing approaches have limited practical application.

The study proposes an AI-powered Secure QR Codes Infrastructure that operates on the web to address these drawbacks. Some of the components of this system include lexical feature extraction, anomaly detection, machine learning models, and threat intelligence services provided by third parties. Realistically, scalable, and efficiently, the proposed system evaluates and classifies QR Codes as benign or dangerous in real-time, providing a means to enhance QR Code security.

International Journal of Advanced Trends in Multidisciplinary Research
ISSN: 3139-0986, Volume 2 Issue 7
July 2026, PP 1-8

Qr Codes: Crafting Attack Vectors
We created and executed many realistic QR Code-based cyberattack scenarios to test how well the proposed Secure QR Codes Infrastructure worked. The proposed system's capacity to identify and neutralise malicious QR Codes is evaluated using these attack vectors, which imitate frequent dangers found in real-world scenarios. Data exfiltration, drive-by-download, malicious payload injection, and JavaScript-based cross-site scripting (XSS) assaults are some of the attack scenarios that have been put into place.

Data Exfiltration Attack
An attacker may stealthily capture sensitive information from users' scanning devices by creating malicious QR Codes that lead them to a site controlled by the attacker. Details about the device, the browser, the network, and even an approximate location might be included in this kind of data. After collection, this information might be used to create profiles of users, conduct targeted phishing attacks, or bolster other harmful endeavours. Consequently, the suggested security architecture relies heavily on the ability to detect suspicious network activity and unauthorised redirections.

Section B: Injecting a Dangerous Payload
Criminals may take advantage of QR Codes' storage capacity by inserting destructive scripts, infected documents, or executable files as payloads. Despite their seemingly innocuous appearance, these payloads pose a threat to the scanning device's security when activated. The suggested system identifies questionable QR Codes before users access the embedded material by performing feature extraction, anomaly detection, and content validation. This is done to counter this danger.

Attack against Drive-by-Download (C)
An attack known as "drive-by-download" uses a QR Code to trick visitors into visiting a malicious website, where they may download malicious files or misleading software. Such assaults may lead to the installation of malware or the theft of credentials since consumers often trust QR Codes without checking the destination URL. To reduce the possibility of illegal downloads, the suggested framework checks the security of destination URLs before letting users continue.

Section D. XSS Attacks Based on JavaScript
Malicious QR Codes may include URLs that are based on JavaScript and, when processed by browsers or apps that aren't secure, can run scripts. Some of the consequences of these types of attacks include data theft, session hijacking, and unauthorised script execution. The suggested system checks QR Codes for legitimate content, checks URLs for unsafe schemes, looks for suspicious JavaScript patterns, and blocks requests that might be harmful before they are executed, therefore reducing these hazards.

Common security risks linked to QR Code technology are represented by the suggested attack vectors. The Secure QR Codes Infrastructure proves its mettle in detecting malicious QR Codes, lowering cybersecurity risks, and enhancing user safety via secure QR Code validation and intelligent threat detection by analysing various attack scenarios.

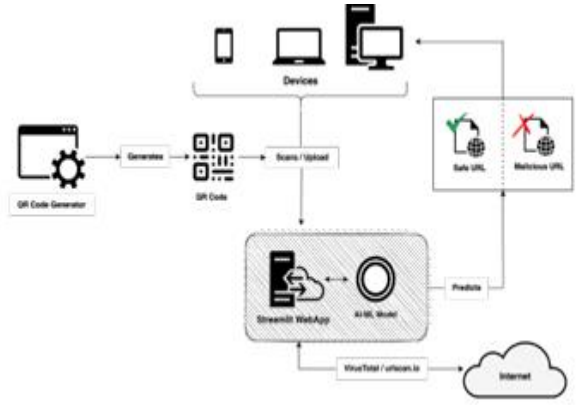

Fig. 1. Secure QR Codes Infrastructure Design

Ap3x Secure Qr Codes Infrastructure
Spotting Attacks on QR Codes with the Help of AI Despite QR Codes' extensive usage in payment processing, authentication, and data transfer, they are susceptible to attacks like SQL injection, phishing, and malware distribution due to their absence of integrity safeguards. Common entrance points for potential dangers are: 1) Disclosure of information. Second, insert all of the harmful code there. including sending a harmful payload over encrypted messages. We built a dataset with both good and bad QR codes and trained machine learning models using visual criteria like pixel distribution and shape to see how well AI could identify dangerous codes. We checked the models' generalisability by subjecting them to various attacks. A number of measures were used to evaluate the robustness of the models, such as F1-score, recall, accuracy, and precision. Our experimental method made it easier to find practical techniques for detecting multi-vector QR Code malware by incorporating dataset creation, attack simulation, and model assessment. We found a security flaw in the QR Code method that most scanners use, which is that they just give you the URLs without checking whether the content is secure. Figure 1 shows the secure infrastructure we built to solve this problem; it allows users to assess the safety of a QR Code according to a number of criteria. In Figure 2 you can see our system's web app. Using machine learning methods, it can identify QR Codes that are dangerous in user-submitted photos or videos. As an added measure, we include VirusTotal and urlscan.io API research into our thorough security evaluation. B. The application's functionalities are easily accessible via the user interface's tabbed navigation. Various interfaces for submitting QR Codes (Short Bar Codes) are provided to users. Picture Forecasting: Users may easily upload PNG, JPG, or JPEG files by dragging and dropping or using the "Browse file" option (Figure 2). Clicking the "Predict" button initiates processing after the user has selected a file.

International Journal of Advanced Trends in Multidisciplinary Research
ISSN: 3139-0986, Volume 2 Issue 7
July 2026, PP 1-8

Fig 2: AP3X Secure QR Codes for Prediction with URLs.

With WebScan Prediction, users can effortlessly take pictures with the tap of a button while watching a live video stream on their smartphone. To examine the collected picture, click the "Predict" button. Last but not least, Figure 2 displays URL Prediction. When users click the "Predict" button, it examines the URL they entered. The processed data is presented in two parts, "Trust metrics" and "QR Codes data & reports." The first section displays all the data gathered via APIs and QR codes. To assist customers make educated decisions, Section 2 informs them of any potentially dangerous substance and provides essential points from APIs and ML models in a side drawer. Section C. Methods Processing makes use of Python utilities to parse the supplied file or picture for data and generate a list of URLs. If VirusTotal does not detect any URLs while inspecting the file, a caution message will be shown. As an alternative, ML models such as XGBoost, LightGBM, Random Forest, and Neural Network are used for prediction, while VirusTotal and urlscan are utilised for scanning.put another way.

In the end, the conclusion helps customers make educated choices. D. Virus Management Through the Use of Third-Party ServicesTo sum up, Total is an antivirus application and web-based scan engine aggregator that lets users check files and URLs for contamination and verify false positives. We analyse user-generated content from QR Codes using this service.

Evaluate the usability of URLs.Io: Website scanning and evaluation made easy with this free service. Using information including URLs, IPs, screenshots, DOM content, JavaScript variables, and cookies, this program automates online surfing and records user activity. Part E: Processing Data The website's front end and back end are both managed via the Streamlit Python package. According to the documentation, "the data is copied to the Streamlit backend via the browser and contained in a BytesIO buffer in Python memory (i.e., RAM, not disc)". When the data is no longer required, it is deleted instantly as it just resides in memory. After a session ends—whether the user closes a tab or hits the refresh button—our program removes all user data from memory.

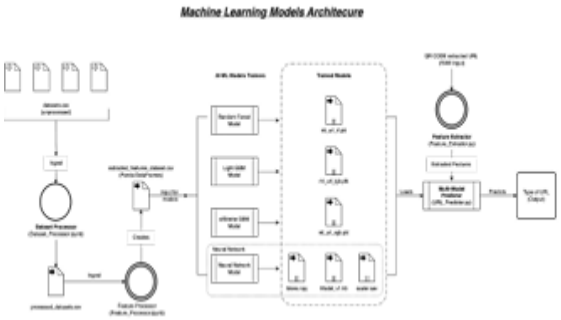

Fig. 3. Machine Learning Models Architecture

Table 1: Extracted Features For Ml Models

Feature Description
having_ip_address: Checks if the URL contains an IP address.
abnormal_url: Determines if the URL lacks a hostname.
google_index: Checks if the URL is indexed by Google.
count_dot: Occurrences of '.' in the URL.
count_www: Occurrences of 'www' in the URL.
count_atrate: Number of '@' characters in the URL.
count_dir: Number of directory paths in the URL.
count_embed_domain: Occurrences of '//' in the path .
shortening_service: Use of known URL Shortening Service.
count_https: Number of times 'https' appears in the URL.
count_http: Number of times 'http' appears in the URL.
count_percent: Number of '%' characters in the URL.
count_question_mark: Number of '?' characters in the URL.
count_hyphen: Number of '-' characters in the URL.
count_equal: Number of '=' characters in the URL.
url_length: Measures the total length of the URL.
hostname_length: Measures the length of the hostname in the URL.
suspicious_words: Presence of suspicious words in the URL.
digit_count: Number of digits in the URL.
letter_count: Number of alphabetic characters in the URL.
fd_length: Measures the length of the first directory in the URL Path.
tld_length: Measures the length of the top-level domain in the URL.
entropy: Calculates the Shannon entropy of the URL.
has_port: Checks if the URL has a port number.

Table I outlines the key features engineered from the URLs in the dataset used to train all the AI models for detecting malicious QR Codes. In total, 24 distinct features that quantify different attributes of the URLs were extracted.

page—preventing attacks that try to save data and lightening the processing burden for data. Section F. Intelligent Machines Data Science Fundamentals The first stages include preparing the dataset and creating the model. Within a Docker container, the web app communicates with other parts of the project using Streamlit APIs. In Figure 3, we can see how efficiently Python modules such as Pandas and Itertools were used to analyse 600,000 URLs from various datasets. Machine learning models were trained and tested on the following datasets: 1. 2016-ISCX-URL Database: Primary database of URLs sorted into safe, malicious, phishing, and defacement categories. (2) Pointers to Anti-Malware Libraries Domain Database for Malware The dataset also contained other URLs that were associated with malware and phishing. 3. The Bad URLs Database, which is a collection of benign URLs extracted from the Faizan Git repository for balance reasons.Datasets from PhishStorm and PhishTank include more phishing URLs (4). Extra URLs that aren't harmful or benign are included in the dataset of safe and dangerous web addresses (5). To guarantee accurate harmful URL categorisation, datasets were joined into a single DataFrame using pd.concat() and were assigned consistent labels for

International Journal of Advanced Trends in Multidisciplinary Research
ISSN: 3139-0986, Volume 2 Issue 7
July 2026, PP 1-8

categories ('benign,' 'defacement,' 'phishing,' and'malicious'). The Second Step: Feature Extraction To aid in the accurate categorisation of hidden QR Codes using embedded URLs, we retrieved attributes such as URL structure, lexicon, entropy, and keywords (Table I). These features are essential for detecting QR Codes that contain malicious URLs. Threat actors often use peculiar patterns to avoid detection and trick consumers. Data gathered from threat intelligence indicates that malicious URLs are notoriously unpredictable and need constant alteration to stay hidden. Their unique IP addresses, strange port numbers, lengthy or complicated pathways, suspicious language, and plenty of special characters set them different from the norm.

Directory depth, obfuscation, and URL length are features that make it harder to comprehend or see potentially harmful material; tests for Google indexing and shortening services show URLs that aren't trustworthy or transparent. When taken as a whole, these traits provide a firm basis for identifying these possible dangers. 3. Machine Learning Models: Our team trained four different machine learning models: XGBoost, NN, RF, and Light Gradient Boosting Machine. In order to conduct a thorough assessment of performance, the data was divided into two sets: training and test. An effective ensemble approach that utilises decision trees, Random Forest (RF) is resistant to overfitting and produces accurate results. Table IV shows that the RF model obtained an astounding 96.7% accuracy, in addition to a good true positive rate across the board and especially for phishing and defacement URL detection. Differentiating between safe and dangerous URLs is an area that needs development.

The LightGBM algorithm is a scalable and effective gradient boosting method. More effort is needed to differentiate between safe and malicious URLs, even though the LightGBM model was able to identify 94.4% of phishing and defacement URLs. The efficiency and speed improvement is in XGBoost, a library for gradient boosting. Regarding the identification of defacement and phishing URLs in particular, the XGBoost model attained an outstanding 94.7% accuracy rate and a high true positive percentage. Our capacity to differentiate between legitimate and malicious URLs may be enhanced. "Neural networks" (NNs) are a kind of pattern recognition system that mimics the way the brain functions. When tested against defacement and phishing efforts, the NN model showed high URL recognition skills with an accuracy rate of 95.0%. With a few adjustments, it may be even more effective at distinguishing between safe and dangerous URLs.

Experimental Evaluation

Two experiments were also developed utilising the AP3X Secure QR Codes Infrastructure as part of the study. Section A delves into the framework for creating and executing

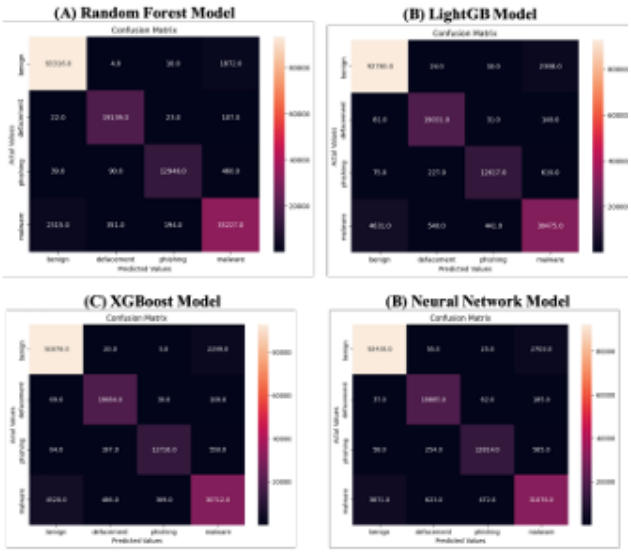

Fig. 4. Confusion Matrices of the Machine Learning Models

trained four models using data from Random Forest, LightGBM, XGBoost, and Neural Network to classify URLs as malicious, benign, defacement, or phishing, and then evaluated their performance. In terms of resistance and misclassification rates, ensemble models outperform the neural network when it comes to detecting phishing and malware assaults, as can be shown from the diagonals of the matrices.

Table 2: Performance Metrics of Different Models

Model	Accuracy	Precision	Recall	F1 Score
RF	0.967	0.97	0.96	0.96
LGB	0.944	0.95	0.93	0.96
XGB	0.947	0.95	0.94	0.94
NN	0.950	0.94	0.94	0.94

Table II compares the four separate ML models using a battery of comprehensive performance metrics, such as F1 score, accuracy, recall, and precision. the web app with ML models built in. While Section B describes the experimental evaluation utilising QR-crafted attacks, Section C covers the same assessment using randomly generated QR Codes. First, the Lab Setup: A Docker container running Ubuntu 22.04 housed the web app, models, and hardware components, including a 12th-generation Intel Core i7 CPU, 8 GB of RAM, and 4 application cores. This project was interpreted using Python 3.10+ and depended extensively on third-party libraries such as Streamlit, urlextract, requests, pyzbar, and pillow. In contrast, ML models rely on a plethora of libraries, including scikit-learn, lightgbm, xgboost, tensorflow, googlesearch-python, and many more. A Oneplus 7T HD1901 running Android 12. Oxygen OS 12.1, and an iPhone 14 Pro MQ0N3LL/A running iOS 17.1 were used to model and carry out the QR code-based attacks. One example of this kind of attack is a QR Codes Modelled Exfiltration server that, after capturing user information, redirects it. Malware is another potential threat that QR codes might conceal.

International Journal of Advanced Trends in Multidisciplinary Research
ISSN: 3139-0986, Volume 2 Issue 7
July 2026, PP 1-8

B. that exploit browser vulnerabilities to spread malware, scan QR codes, execute JavaScript payloads (including XSS), and trick users into visiting dangerous websites and downloading dangerous files. Since data exfiltration occurs as a result of a custom implementation of a QR Codes generator, we generated QR Codes using a PHP-Laravel project that could connect to any destination. Upon completion of the implementation, the user will be sent to the exfiltration URL that is included in the QR Codes. Injecting malicious payloads into QR codes was possible with the use of the command line tool qren code, which could generate QR codes using a Windows executable. Metasploit manufactured the executable by using its msfvenom framework. At its most fundamental, all it did was open a new window with some text shown within. You may get the executable using zbarimg, a command line program that is part of zbar-tools. QR code-based mechanism for downloading and running: We were able to access a Google Drive file using a compromised version of the Brave browser (v1.50.114 based on Chromium 112.0.5615.49) and QR codes that we generated during our assault. Using the QR Codes Generator, we created the QR Codes for the attack. An exploitable vulnerability in Mozilla Firefox for Android version 93.2.0, known as Universal Cross-site Scripting (UXSS), might be achieved by generating URLs using a QR Codes generator that begin with 'javascript:' rather than the usual 'http:' or 'https:'. Upon receiving the infected QR Codes, we wasted no time scanning and uploading them to our system. After reading the QR Codes, the AP3X Secure QR Codes Infrastructure analyses the data and delivers the result. In making this choice, we consulted a number of machine learning models and APIs, such as VirusTotal and urlscan.io. The hyper-looped utility uses C and randomly selected QR codes. Green Threads We used over a thousand randomly chosen QR Codes for this experiment. We constructed an asynchronous framework as part of our research to test our suggested approach on the QR Codes dataset that was previously described, using green threads. In order to ensure that testing went off without a hitch, we used Python's greenlets to speed up and improve the interface. Under this configuration, the code iterated over the dataset, retrieving and analysing the prediction outcomes for every QR Code. We were able to gauge the system's ability to detect and resist QR Code security threats in this way. Part 7.

III. RESULTS AND DISCUSSIONS

Data is categorised as URL or non-URL, features are collected from QR Codes, and urlscan.io and VirusTotal are used for prediction as part of the entire security plan.

Experimental Results of or Codes-Based Attacks Using Mobile Devices

Results may be seen in Table III, which includes the OnePlus 7T and the iPhone 14 Pro. They found that AP3X Secure QR was able to successfully counter four different types of QR code-based attacks.

QR Attack	Without Security	With AP3X Secure QR
Exfiltration Attack	Successful	Redirection
Payload Injection	Successful	Executable
Drive-by-download	Successful	Malicious
UXSS JS Execution	Successful	Unknown

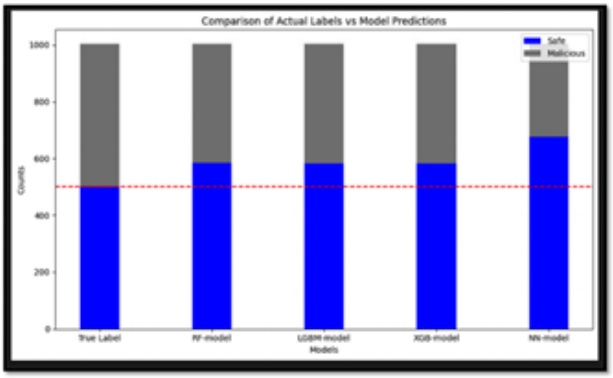

Fig. 5. Graphical Comparison of the Actual Labels vs ML Model Predictions for safe and malicious URLs for URL data.

Using VirusTotal, we compress data that is free of URLs. As we saw in Section 4, this method successfully lessens the dangers of QR code-based assaults; the results are shown below. A Comparative Analysis of AP3X Secure QR Codes and Attacks on QR Codes Based on Their Structure In this experiment, the AP3X Secure QR Codes Infrastructure was tested for its robustness by inserting a payload, driving users to download the code, and executing UXSS JS. The inherent AI modules or logic in the system identified every single assault. The outcomes and total efficacy of every strike are shown in Table III. Rationale for Comparing AP3X Secure QR Codes to Regular QR Codes Using information from reliable sources like VirusTotal and urlscan.io, we ran 497 "safe" QR Codes through the AP3X Secure QR Codes Infrastructure and 503 "malicious" ones. We built this dataset to mimic real-world circumstances and threats so you can test with confidence. We were able to examine each QR Code using a state-of-the-art hyper-looped tool and the AP3X Secure QR solution. The results are shown in Figure 5. In order to evaluate the efficacy of our detection systems in simulated operating settings, we developed a strategy to identify the inherent hazards. Despite the potential for incorrectly identifying harmless URLs, the Random Forest (RF) model achieves respectable results in recall (0.83300) and F1 score (0.90889), as seen in Table IV, making it an appropriate choice for situations necessitating strong true positive detection. Despite concerns that LightGBM could miss certain threats (recall = 0.83101), it strikes a good mix between scalability and accuracy (r=0.99), making it a great choice when avoiding false positives is priority number one. Even though XGBoost has a modest recall of 0.83499, it is an excellent choice for applications requiring low false positive rates and good overall dependability because to its peak accuracy of 0.91724 and precision of 1.00. Many metrics show that the Neural Network needs more optimisation; for example, recall is 0.54870 and accuracy is 0.72183. While XGBoost

and LightGBM are great gradient-boosted models to use when precision is critical, RF is great for balanced detection in situations when risk is high. Their resilience could be enhanced by integrating these components into a hybrid approach.

Limitations

Using our ML models in the real world yielded encouraging findings, however there are still obstacles. The possibility that bad actors may take advantage of AI systems by manipulating inputs to bypass safeguards is a major cause for worry. The models' resistance to data poisoning and adversarial assaults was not tested. Evaluation of QR Code Security Thoroughly assessing infrastructure in relation to these dangers need to be a top research priority. Furthermore, models may be unable to acquire real-time URL attributes owing to query constraints and related costs if they depend on third-party APIs such as VirusTotal and urlscan.io. To prevent scanning system impairments caused by availability issues like denial-of-service attacks, our security patches primarily targeted user privacy and integrity. Research in the future must prioritise investigating a broader spectrum of possible risks. We need to test the system in real-world circumstances, integrate it better, confirm encrypted security explicitly, fix usability difficulties, and evaluate robustness against adversarial ML, among other things. Because cybersecurity is a dynamic field, training datasets could be out of date. It is possible that the model's predictions may be enhanced by including more datasets and attributes. Attacks that target specific content, the ability to extract data in real time, and details about domain registration are all instances of URL characteristics. In terms of computational costs, system latency, and dependence on external APIs, scaling the study for practical usage is challenging. These factors include, but are not limited to, cost, query limits, and possible failure locations. The proliferation of QR Codes necessitates industry-specific detection algorithms and modular architectures to accommodate their adaptability, which further complicates matters. Some decentralised approaches that could aid with scalability, reducing external dependencies, and maintaining systems resilient with real-time threat awareness include federated learning and edge-based detection. Further investigation is necessary in this domain.

IV. CONCLUSION AND FUTURE WORK

Given the increasing sophistication of cyberattacks, this study's finding that QR Code systems benefit from combining conventional security measures with AI and ML is quite pertinent. An impressive 93% or higher was achieved by all four models (Random Forest, LightGBM, XGBoost, and Neural Network) when it came to identifying simulated assaults. Phishing redirection, QR code exfiltration, payload injection, and JavaScript exe cution were among the several assaults. Advancements could result from studies examining the potential of Secure QR Codes in the future. The separation of the web server architecture and the machine learning models is one potential architectural choice. Another option is to use an orchestrator such as Kubernetes. This might potentially enhance performance and enable modifications to models without impacting the functioning of the servers. The hunt for assaults using QR codes can benefit from trying out various AI algorithms. Finally, our research shows that ML, in conjunction with other tools like integrity checks, APIs, anomaly detection engines, proxy filtering, and AI-based predictions, may modify QR Code security measures to counter new attack approaches. Synthesising human oversight with AI that studies dispersed data vectors may help safeguard systems over the long run from cyber threats that are always changing. Protecting against threat actors who are developing countermeasures requires further study on adversarial ML.

REFERENCES

1. H. Huang, F. Chang and W. Fang, “Reversible data hiding with histogram-based difference expansion for QR Codes applications,” IEEE Transactions on Consumer Electronics, vol. 57, (2), pp. 779-787, 2011.
2. N. P. A. Karniawati et al, “Community Perception of Using QR Codes Payment in Era New Normal,” PalArch’s Journal of Archaeology of Egypt/Egyptology, vol. 18, (1), pp. 3986-3999, 2021. By
3. (November 15). “Be careful what you scan: QR scams increase 51%”. Available: qr-code-phishing-scams-quishing/. https://cybernews.com/news/
4. (October 27,). “The Rise in QR Codes Attacks.” Available: https://www. digit.fyi/comment-the-rise-in-qr-code-attacks/.
5. (November 21,). “The alarming rise of quishing is a red flag for CISOs.” Available: https://www.csoonline.com/article/1248084/ the-alarming-rise-of-quishing-is-a-red-flag-for-cisos.html
6. (October 29,). “Surge in QR Codes Quishing: Check Point Records 587% Attack Spike.” Available: https://www.hackread.com/ qr-code-quishing-check-point-attack-spike/.
7. K. Krombholz et al, “QR Codes security: A survey of attacks and challenges for usable security,” in Human Aspects of Information Security, Privacy, and Trust: Second International Conference, HAS 2014, Held as Part of HCI International 2014, Heraklion, Crete, Greece, June 22 27, 2014. Proceedings 2, 2014.
8. A. Pawar et al, “Secure QR Codes scanner to detect malicious URL using machine learning,” in 2022 2nd Asian Conference on Innovation in Technology (ASIANCON), 2022.
9. P. Kieseberg et al, “QR Codes security,” in Proceedings of the 8th Inter national Conference on Advances in Mobile Computing and Multimedia, 2010.
10. R. Focardi, F. L. Luccio and H. A. Wahsheh, “Usable security for QR Codes,” Journal of Information Security and Applications, vol. 48, pp. 102369, 2019.

11. X. Han et al, “Medusa attack: Exploring security hazards of {in app}{QR} code scanning,” in 32nd USENIX Security Symposium (USENIX Security 23), 2023.
12. M. S. Ahamed and H. A. Mustafa, “A secure QR Codes system for sharing personal confidential information,” in 2019 International Conference on Computer, Communication, Chemical, Materials and Electronic Engineering (IC4ME2), 2019.
13. L. Finˇzgar and M. Trebar, “Use of NFC and QR Codes identification in an electronic ticket system for public transport,” in SoftCOM 2011, 19th International Conference on Software, Telecommunications and Computer Networks, 2011.
14. T. Vidas et al, “QRishing: The susceptibility of smartphone users to QR Codes phishing attacks,” in Financial Cryptography and Data Security: FC 2013 Workshops, USEC and WAHC 2013, Okinawa, Japan, April 1, 2013, Revised Selected Papers 17, 2013.
15. A. Kharraz et al, “Optical delusions: A study of malicious QR Codes in the wild,” in 2014 44th Annual IEEE/IFIP International Conference on Dependable Systems and Networks, 2014.
16. V. Mavroeidis and M. Nicho, “Quick response code secure: A crypto graphically secure anti-phishing tool for QR Codes attacks,” in Com puter Network Security: 7th International Conference on Mathematical Methods, Models, and Architectures for Computer Network Security, MMM-ACNS 2017, Warsaw, Poland, August 28-30, 2017, Proceedings 7, 2017.
17. M. O. Yadav et al,“Online reservation system using QR Codes based android application system,” International Journal of Scientific and Research Publications, vol. 4, (12), pp. 1-6, 2014.
18. (August). “Fitting snake into a QR Codes.” Available: https://hackaday. com/2020/08/17/fitting-snake-into-a-qr-code/.
19. MattKC, “Can you fit a whole game into a QR Codes?,” vol. Web, July 31, 2020. [20] “Open redirect due to scanning QR Codes via brave browser.” Available: https://hackerone.com/reports/1946534.