International Journal of Computational Research in Science and Technology
E-ISSN: 3108-0871, Volume 2 Issue 5
Sep-Oct-2026, PP 01-08

UPI Qr-Code Tampering Detection System AI-Based Detection of Malicious and Tampered Qr Codes in UPI Payments

S. Kushala Shree[1] Tanishqa[2], Akanksha P. Bhat[3] , M. R. Prathima[4] , Suzzaine[5] , Jashwanth M. K[6]
Bachelor Of Computer Applications Padmashree Institute Of Management & Sciences,
Bengaluru

ABSTRACT- Unified Payments Interface (UPI) has made digital payments convenient by allowing users to transfer money through supported payment applications. QR codes simplify this process because merchant details can be encoded in a pattern that a phone camera can read. However, a QR code can also be replaced, covered, altered, or used to direct a user towards an unintended payment destination. A person may not be able to identify the destination by looking at the QR image alone. This creates a need for an additional warning layer that checks the decoded information before a user proceeds. This paper proposes an AI-assisted UPI QR-code tampering detection system. The proposed workflow captures or uploads a QR image, decodes its payload, checks payment-related fields, extracts suspicious indicators, and uses a machine-learning classifier together with clear security rules to produce a risk category. The output may be shown as "Lower Risk," "Needs Verification," or "High Risk," with a short explanation such as a mismatch between the expected merchant and decoded payee. The system is intended as a decision-support tool rather than a replacement for a bank or UPI application's security controls. The study discusses the problem, related work, system architecture, data requirements, feature extraction, model selection, evaluation measures, limitations, and future scope. Since no validated dataset or live payment integration is supplied for this paper, the proposed system is not presented as a tested product. A prototype should be evaluated on representative, legally obtained data before any real-world deployment.

Keywords: UPI, QR code, tampering detection, artificial intelligence, machine learning, digital payment security, anomaly detection, fraud prevention.

I. Introduction

1. Background

Digital payment systems are now used in shops, transport, online commerce, and person-to-person transfers. UPI supports payment flows through participating banks and applications. QR-based payment is particularly useful for small merchants because a printed code can represent payment information without requiring the customer to type every detail manually. NPCI explains that QR-based UPI integration can store merchant information such as a UPI ID and amount, which is displayed to the customer in the payment interface [1]. The same convenience creates a security concern when the displayed code is not the code the customer expects. A malicious person may place a sticker over a genuine merchant QR code, share a misleading code through a message, or direct a scan to a fraudulent web page. These situations are often described broadly as QR-code fraud or QR phishing. RBI awareness material warns users to be careful with unknown QR codes and states that a PIN or OTP is not required simply to receive money [2].

2. Problem Statement

A conventional QR scanner primarily reads encoded information; it does not necessarily know whether the destination belongs to the intended merchant. The proposed problem is to design an additional screening system that examines decoded QR content and available context, identifies suspicious patterns, and explains why a code should be checked before a payment is approved.

3. Objectives

- Study common risks associated with malicious, replaced, or tampered QR codes in UPI payment scenarios.
- Design a clear pipeline for QR image processing, decoding, feature extraction, and risk classification.
- Explore how machine learning can support rule-based security checks.
- Provide understandable warnings without collecting unnecessary personal or banking information.
- Define evaluation measures and ethical limits for a future prototype.

4. Scope

The proposed work focuses on the QR image and its decoded content. It does not access bank accounts, initiate transactions, or guarantee that a payment is safe. A practical implementation would need consent, careful data handling, testing with legitimate and suspicious samples, and review by qualified security professionals.

5. Research Questions

- **RQ1:** Does combining visual features with decoded-payload features improve malicious-QR classification compared with visual-only and payload-only baselines?
- **RQ2:** Does matching the decoded payee against a trusted merchant reference identify recipient substitution that cannot be inferred reliably from appearance alone?
- **RQ3:** How do blur, rotation, perspective distortion, glare, low light, compression, and partial occlusion affect detection quality, false alarms, and inference latency?
- **RQ4:** Can the system provide concise, evidence-linked explanations that help a user understand why a code is flagged without implying a guarantee of safety?

6. Planned Contributions

The planned contributions are a dual-channel architecture, explicit trusted-reference matching, a three-outcome decision policy, human-readable explanations, and a reproducible evaluation protocol. These are design contributions, not evidence of measured performance. Their value must be established through implementation and testing.

II. Literature Review

Existing guidance and research show that QR-code security needs both user awareness and technical checks. The following sources inform the proposed design.

- **Official payment-safety guidance**

  NPCI's fraud-awareness guidance describes QR codes among the methods that can be misused in scams. It warns that scanning a code and entering a UPI PIN is for making a payment, not for receiving money [2]. RBI's public awareness portal similarly advises people not to scan unknown QR codes or click unknown links and clarifies that a PIN or OTP is not required to receive funds [3]. These recommendations motivate a system that highlights unexpected destinations and asks users to verify the payee.

- **Machine-learning research**

  Research on recognising valid and malicious QR codes has explored image-based deep-learning models. A 2024 IEEE conference paper describes using QR images and convolutional neural network architectures to distinguish benign and malicious codes [4]. Such approaches can potentially identify visual patterns, but their usefulness depends on dataset quality, realistic examples, and testing on unseen samples. Other work explores classification of QR codes and associated URLs using combinations of machine-learning methods and extracted features [5]. URL-based checks can help when a QR code contains a web link. However, a UPI payment QR may contain structured payment information rather than a normal web URL, so a system focused on UPI must inspect payment-related fields as well.

1. Research gap

A general malicious-QR detector may not fully address the specific question: does the decoded payee match the merchant the user intended to pay? A UPI-focused system should therefore combine payload analysis with context-aware checks, where trusted reference information is available. It should also explain the reason for a warning instead of displaying only a probability score.

2. Summary

The reviewed guidance supports safe scanning practices, while technical studies support exploring image analysis and machine learning. The proposed framework combines these ideas but requires independent evaluation before any claims about accuracy or fraud reduction can be made.

3. Proposed System And Architecture

Overview

The proposed system acts as a pre-payment screening layer. A user uploads or captures a QR image. The image-processing module improves readability if needed and passes the image to a decoder. The decoded payload is checked for structure, unexpected fields, suspicious links, and differences from trusted merchant details supplied by the user or a verified source.

Main modules

- QR image input: receives a photograph or image file with user permission.
- Image preprocessing: corrects rotation, adjusts contrast, and reduces noise when necessary.
- QR decoding: extracts the encoded text or payment payload using a recognised decoding library.
- Payload parser: identifies the payload type and safely parses available fields.
- Feature extraction: calculates indicators such as payload type, field consistency, link characteristics, and whether the expected payee is available.
- Risk engine: combines deterministic rules with a machine-learning model.
- Warning interface: shows a risk category and the reason for the warning.
- Audit log: optionally records minimal technical outcomes without storing sensitive payment details.

Text-based architecture

User captures QR image → Image preprocessing → QR decoder → Payload parser → Feature extraction → Rule checks + ML classifier → Risk category and explanation
→ User verifies payee in the official payment app.

4. Risk categories

Lower Risk means no warning indicator was found in the available checks; it does not mean the code is guaranteed safe. Needs Verification means some details are missing or unusual. High Risk means one or more strong warning indicators were found, such as a known malicious destination or a mismatch against trusted merchant information. The user should not proceed until the concern is resolved.

5. Design principle

The system should fail safely. If the image cannot be decoded, the payload is malformed, or required reference information is unavailable, it should say that it cannot verify the code rather than labelling it safe.

6. Database Design and ER Diagram

Proposed database structure for the AI-based detection of malicious and tampered UPI QR codes.
Purpose of the Database
The proposed database stores the information needed by the QR-code screening system. It keeps merchant reference details, QR-code records, transaction information, AI-based detection outcomes, and security alerts in an organised form. The database supports comparison between the payee information decoded from a QR code and the trusted merchant details available to the system. It is a proposed design for a student research prototype, not a live banking database.

7. Main Database Entities

- Merchant: Stores the merchant's basic details, including merchant_id (Primary Key), merchant_name, and contact_number. The merchant_id uniquely identifies each merchant.
- QR_Code: Stores QR-code reference information such as qr_id (Primary Key), merchant_id (Foreign Key), qr_hash, payee_id, qr_status, and created_at. The hash can help identify whether a stored QR image or reference has changed. The payee_id represents the payment destination identifier available in the decoded payload.
- Transaction: Stores a transaction reference, qr_id (Foreign Key), amount, payment_status, and transaction_time. In a classroom prototype, these should be test records or consented sample data rather than real banking credentials or private histories.

8. Detection and Alert Records

- Detection_Result: Stores detection_id (Primary Key), qr_id and transaction_id (Foreign Keys), risk_score, detection_status, and reason. The result records the system's assessment, for example, whether the decoded payee differs from a trusted merchant reference. A risk score is an indicator for review and must not be treated as proof of fraud.
- Alert: Stores alert_id (Primary Key), detection_id (Foreign Key), alert_message, and alert_time. An alert can

explain why the user should verify the payee before proceeding.

9. Relationships Between Tables

One merchant can be associated with multiple QR-code records. One QR code can be linked to multiple transaction records and can have multiple detection records over time. A detection result may generate one or more alerts. Foreign Keys connect related records and help maintain consistency between the tables.

10. Database Workflow

When a QR image is submitted, the system decodes its payment payload and checks the available payee details. The QR-code reference and relevant test transaction information are retrieved from the database. The rule checks and machine-learning model produce a detection result. If suspicious indicators are found, the system stores an alert explaining the concern. The user must still verify the recipient name and amount in the official UPI app.

11. Privacy and Security

The database must never store UPI PINs, OTPs, passwords, or unnecessary account details. Only the minimum information required for the research prototype should be retained. Access should be restricted, sample data should be anonymised where possible, and records should be protected from unauthorised changes. A 'lower risk' result does not guarantee that a QR code is safe.

III. Methodology And Feature Extraction

1. Research method

The work follows a design-oriented methodology: define the threat, review public safety guidance and related research, design a processing pipeline, identify measurable features, select candidate models, and define an evaluation plan. A student prototype can be developed in Python using image-processing and machine-learning libraries, but it must be tested using authorised data.

2. Data collection and labelling

A useful dataset should contain legitimate QR samples and suspicious or manipulated samples. Legitimate examples may be generated for test merchants using synthetic details. Suspicious examples should be sourced only from lawful research datasets or created in a controlled lab without using real people's payment information. Each sample should have a label, source, collection date, and a note describing the reason for the label. Duplicate images should be removed, and training and test sets should be separated by underlying QR payload to reduce data leakage.

3. Candidate features

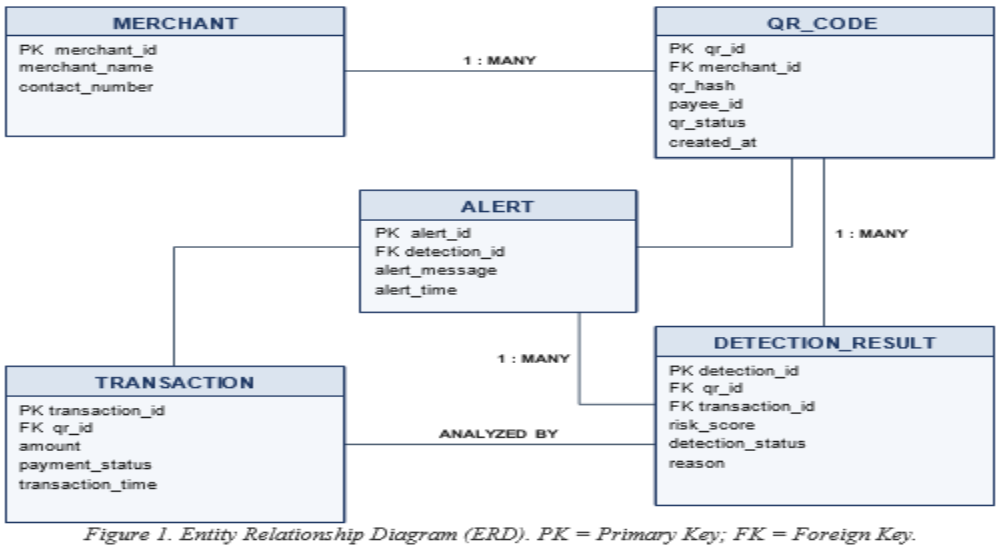


Figure 1. Entity Relationship Diagram (ERD). PK = Primary Key; FK = Foreign Key.

- **Required-field checks:** whether expected fields are present and correctly formatted.

- **Payee consistency:** whether the decoded payee matches a trusted merchant reference, if one is available.
- **Amount consistency:** whether a pre-filled amount is unexpected or differs from the amount the user intended.
- **URL indicators**: suspicious or misleading host names, unusual redirects, or known threat-intelligence matches when the QR contains a URL.
- **Image-quality indicators**: blur, partial obstruction, low contrast, or signs that the code may be difficult to read.
- **Context indicators:** source of the QR image and whether it was received unexpectedly, when the user chooses to provide that information.

4. Model selection

A baseline can use logistic regression or a decision tree because these models are relatively easy to explain. Random Forest may be evaluated as a stronger tabular-data baseline. A convolutional neural network may be considered if the project has a sufficiently large and diverse image dataset. Model selection should be based on independent testing, not on a single accuracy score.

5. Privacy and security

The prototype should avoid collecting UPI PINs, OTPs, passwords, account balances, or full transaction histories. Sensitive fields should be masked or discarded where possible. Any logs should have a defined purpose and retention period. The system should not send a user's payment details to an external service without clear consent.

6. Implementation Plan

1. Suggested tools

Python can coordinate the prototype. OpenCV can be used for image preprocessing, a QR decoding library can extract payload text, and pandas can organise labelled examples. Scikit-learn can train and evaluate baseline models. A simple Streamlit interface may allow a user to upload a QR image and read the warning. These are proposed options, not a claim that the system has already been implemented.

2. Processing steps

- Step 1: Load a QR image from an authorised sample folder or user upload.
- Step 2: Check image dimensions and attempt QR decoding.
- Step 3: If decoding fails, display "Unable to verify" and stop.
- Step 4: Parse the payload using a safe parser; do not automatically open embedded links.
- Step 5: Extract applicable features and compare payee details with trusted reference data when available.
- Step 6: Apply explicit safety rules and pass eligible features to the trained classifier.
- Step 7: Combine the results into a risk category and provide a plain-language explanation.
- Step 8: Ask the user to verify the displayed payee and amount in their official UPI app before authorising any transaction.

3. Example of a warning

Example only: "Needs verification: the decoded payee does not match the merchant name you entered. Do not continue until you confirm the recipient." This example illustrates the intended user experience; it is not a real detection result.

4. Testing plan

Test cases should include clear legitimate codes, blurred images, rotated images, partially covered codes, malformed payloads, unexpected URLs, duplicate images, and samples whose payee differs from a trusted reference. Testing should include cases that the model has not seen during training.

5. Implementation boundary

A classroom prototype should not connect directly to a bank or attempt live payments. It should only analyse test images and display guidance. Any real deployment would require security review, consent, regulatory assessment, and integration approved by the relevant payment providers.

7. Evaluation And Expected Outcomes

1. Evaluation measures

The system should be evaluated with a held-out test set. Accuracy alone is insufficient because a dataset with many legitimate codes can make a weak detector appear successful. Precision measures how many codes flagged as malicious are actually malicious in the labelled test set. Recall measures how many malicious samples are detected. F1-score balances precision and recall. A confusion matrix shows true positives, false positives, true negatives, and false negatives.

$$\text{Precision} = TP / (TP + FP) \quad \text{Recall} = TP / (TP + FN)$$

$$\text{F1-score} = 2 \times (\text{Precision} \times \text{Recall}) / (\text{Precision} + \text{Recall})$$

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP + FN)$$

Here, TP means malicious codes correctly flagged, TN means legitimate codes correctly accepted by the test classifier, FP means legitimate codes incorrectly flagged, and FN means malicious codes missed by the detector.

2. Expected outcomes

The intended outcome is a working prototype that decodes QR images, applies consistent checks, and provides understandable warnings. It is reasonable to expect that clear rule-based checks will catch some obvious mismatches, but the level of performance must be measured. No numerical accuracy, processing speed, or reduction in financial loss is claimed in this paper.

3. Error analysis

False positives can inconvenience users and reduce trust if safe codes are repeatedly flagged. False negatives are more serious because a malicious code may pass through the system. Error analysis should examine whether failures arise from poor images, incomplete labels, new attack patterns, or missing merchant context. The model should be updated only through a documented and reviewed process.

4. Limitations of evaluation

Results from synthetic QR images may not reflect real-world conditions such as glare, sticker overlays, changing merchant information, or new fraud techniques. Public datasets may also contain duplicate or outdated examples. Therefore, testing should report the dataset composition, split method, class balance, and limitations.

8. Benefits, Challenges And Ethical Considerations

Potential benefits

- Adds a screening step before a user proceeds to a payment.
- Can highlight payee or amount mismatches when reliable reference details are available.
- May help users understand why an unfamiliar code deserves caution.
- Supports cybersecurity awareness through practical examples.
- Provides a modular project that can be extended with better datasets and explainable models.

9. Challenges

- Reliable labels for malicious QR codes may be difficult to obtain.
- Attackers can change their techniques, so static rules may become outdated.
- Some QR payloads do not contain enough information to establish the real-world identity of a merchant.
- Image damage may prevent decoding even when the code is legitimate.
- Machine-learning models may overfit to the source or appearance of the training images.
- Excessive warnings may cause users to ignore important alerts.

10. Ethical considerations

The project should use legally obtained data and avoid exposing personal payment details. Test samples must not be used to trick real users or initiate transactions. Results should be communicated honestly, including the possibility of mistakes. A

"lower risk" label must never be presented as a guarantee. The system should support, not replace, user verification and official payment-app safeguards.

11. User safety recommendations

Users should check the recipient name and amount shown by their UPI application before approving a payment. They should not scan unknown codes just because someone claims that scanning is necessary to receive money, and should never disclose a UPI PIN or OTP. These practices align with public guidance from NPCI and RBI [2][3].

12. Case Study: Reported Retail Upi Qr-Code Tampering Incident

- **Incident Overview**
  A reported incident involving a 19-year-old youth named Manish Verma illustrates how QR-code tampering may affect retail payments. According to the incident details supplied for this research paper, Delhi Police arrested the suspect in Jaipur, Rajasthan, over the alleged use of AI software to alter shop QR codes and divert digital payments to a bank account under his control. The incident involved a garment shop in Chandni Chowk, Delhi. A customer making a purchase worth ₹2.5 lakh scanned the QR code displayed at the store and made two UPI payments of ₹90,000 and ₹50,000, totalling ₹1.4 lakh. The shop management reportedly did not receive the funds, although the customer showed payment screenshots.

- **Reported Method of QR-Code Tampering**
  The suspect allegedly contacted vendors online and requested copies of their official QR codes while posing as a prospective customer. The QR images were reportedly modified using AI tools so that the encoded payment destination changed while the shopkeeper's displayed name remained intact. In some instances, altered QR images were reportedly saved in vendors' phone galleries for later use. This illustrates why a familiar-looking merchant name alone cannot establish that a QR code directs funds to the intended recipient.

- **Investigation and Recovery**
  The payment trail reportedly led investigators to Rajasthan, where the suspect was arrested in Jaipur. More than 100 edited and original QR codes, along with financial records, were reportedly recovered. These details are presented as reported case information and are not experimental findings produced by the proposed system.

- **Relevance to the Proposed Detection System**
  This case relates directly to the threat model discussed in this paper. The unchanged merchant name alongside a modified payment destination is an example of payee-identifier substitution. A visual-only classifier may fail when the altered QR image remains clean and readable. Therefore, the proposed framework should decode the payment payload and compare the payee identifier with an independently trusted merchant reference. A mismatch should generate a high-priority warning even when the QR image appears ordinary. If no trusted reference is available, the system should report insufficient evidence rather than declaring the QR code safe.

- **Preventive Measures for Merchants and Customers**
  Merchants should confirm receipt through their bank application, payment soundbox, or point-of-sale (POS) terminal instead of relying only on a customer's payment screenshot. Physical QR standees and stickers should be inspected regularly for replacement or unauthorized overlays. Customers should check the recipient details and amount displayed by their UPI application before approving a payment. A QR code is used to make a payment, not to receive one. A person does not need to scan a QR code or enter a UPI PIN merely to receive money. Anyone who suspects digital-payment fraud should contact the National Cyber Crime Helpline at 1930 promptly or submit a complaint through the National Cyber Crime Reporting Portal.

- **Case-Study Implications**
  The incident reinforces the central motivation of this research: QR security should be assessed using both image integrity and the meaning of the decoded payment payload. Recipient consistency against a trusted source is particularly important because an altered destination may be concealed behind an apparently legitimate merchant name. The proposed prototype can evaluate this risk offline using synthetic or consented samples; it cannot independently validate the police account or establish the outcome of any real transaction.

- **Source note:** The case details in this section were supplied for inclusion in the research paper. Before academic submission, add a verifiable news report or official police source with publication date and URL. The incident should be described as alleged/reported unless independently verified.

V. Conclusion And Future Scope

1. Conclusion

QR codes make UPI payments quick, but the information they encode can be misleading when a code is replaced, altered, or shared by an untrusted source. This paper proposed an AI-assisted screening framework that combines image preprocessing, QR decoding, payload validation, feature extraction, rule-based checks, and machine-learning classification. It is designed to return an understandable risk category and explain the reason for a warning. The key contribution is the proposed workflow and evaluation plan, not a claim of proven fraud detection. A reliable system requires representative data, careful handling of false negatives, privacy protection, and independent testing. Even when a scanner gives a lower-risk result, users must verify the recipient and amount in the official payment application before confirming a transaction.

2. Future Scope

- Build a prototype with Python, OpenCV, a QR decoder, and Scikit-learn.
- Create a documented dataset of synthetic and authorised samples with clear labels.
- Compare rule-based checks, Logistic Regression, Decision Tree, and Random Forest.
- Add explainable warnings that identify the specific reason for concern.
- Explore secure integration with merchant verification services, subject to provider approval.
- Evaluate performance on new QR samples, varied camera conditions, and different devices.
- Conduct usability testing to ensure warnings are understandable and accessible.

Final note: The proposed system should be treated as an educational research design until a prototype is implemented and validated. It should never be used as the sole basis for deciding whether a financial transaction is safe.

References

1. National Payments Corporation of India (NPCI), “UPI – Frequently Asked Questions,” QR-based integration mode. Available: https://www.npci.org.in/what-we-do/upi/faqs
2. National Payments Corporation of India (NPCI), “Fraud Awareness.” Available: https://www.npci.org.in/fraud-awareness
3. Reserve Bank of India, “Be Aware: Fraud through UPI QR / QR-code safety guidance.” Available: https://rbikehtahai.rbi.org.in/qr
4. “Recognition of Valid QR Codes with Machine Learning,” IEEE 13th International Conference on Communication Systems and Network Technologies (CSNT), 2024. DOI: 10.1109/CSNT60213.2024.10546171.
5. “HybridStack-MLP: Advanced Ensemble Learning for Malicious QR Code and URL Detection,” IEEE 2nd International Conference for Women in Computing (InCoWoCo), 2025; published/added to IEEE Xplore in 2026. DOI: 10.1109/InCoWoCo68239.2025.11407094.
6. Reserve Bank of India, “Master Direction on Digital Payment Security Controls,” February 18, 2021. Available through RBI’s official website. Academic integrity note This is an original, student-oriented draft written from the cited sources and general technical concepts. Similarity-checking tools may still identify standard technical terms, titles, or properly cited source details. Review the paper, adapt it to your own project and college format, verify each reference, and cite any additional material you use.