International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

TrustBill AI: An Intelligent GST Invoice Fraud Detection and Verification System Using OCR, Machine Learning, and Deep Learning

A. M. Bhunje
SVPM's College of Engineering
Baramati, Pune, Maharashtra, India

Rahul Ghorpade
SVPM's College of Engineering
Baramati, Pune, Maharashtra, India

Om Dhumal
SVPM's College of Engineering
Baramati, Pune, Maharashtra, India

Saurabh Chavan
SVPM's College of Engineering
Baramati, Pune, Maharashtra, India

Atharva Khalate
SVPM's College of Engineering
Baramati, Pune, Maharashtra, India

Abstract - TrustBill AI is an intelligent invoice verification and fraud detection system developed to address financial losses caused by fake invoices, duplicate billing, forged GST numbers, and suspicious vendor activities. Traditional manual invoice verification methods are slow, error-prone, and difficult to manage at large transaction volumes. To overcome these limitations, the proposed system integrates Optical Character Recognition (OCR), Machine Learning, and Deep Learning techniques for automated invoice analysis and fraud detection. The platform combines EasyOCR, Pytesseract, and transformer-based TrOCR models for accurate invoice text extraction under varying image conditions. Fraud analysis is performed using IsolationForest-based anomaly detection, CNN-assisted visual pattern analysis, duplicate invoice detection using perceptual hashing and cosine similarity, and vendor risk profiling mechanisms. In addition, DistilBERT and Named Entity Recognition (NER) techniques are utilized to improve contextual understanding and structured extraction of invoice entities such as GST numbers, vendor names, invoice IDs, and transaction amounts. The system also provides a multi-role dashboard for Admin, Finance, Auditor, and User operations along with a WhatsApp-based chatbot interface for remote invoice submission and real-time verification. Experimental evaluation conducted on synthetic and real-format invoice datasets demonstrates reliable OCR extraction accuracy and effective fraud detection performance, outperforming traditional rule-based and single-engine approaches across multiple evaluation metrics.

Keywords: ***Invoice Fraud Detection, Optical Character Recognition, GST Verification, Anomaly Detection, Perceptual Hashing, Multi-Role Dashboard, WhatsApp Integration, Machine Learning, IsolationForest, TrustBill AI***

1. INTRODUCTION

Fraudulent invoicing has become a serious financial concern for organizations operating across industries such as manufacturing, healthcare, logistics, and information technology. Billing-related fraud activities, including fake vendor creation, repeated invoice submissions, and manipulated billing records, continue to cause major financial losses for businesses worldwide [1]. In India, the implementation of the Goods and Services Tax (GST) system has increased the importance of accurate invoice verification, as invalid or forged GST identification numbers are frequently used for tax evasion and false claim generation [2]. Despite the growing volume of digital transactions, many organizations still rely on manual invoice checking processes that involve human verification of invoice details, vendor information, and GST records. These traditional methods are time-consuming, difficult to scale, and highly vulnerable to human error.

Recent developments in Optical Character Recognition (OCR), anomaly detection, and machine learning techniques have created opportunities for building intelligent invoice verification systems. However, most existing solutions focus only on limited tasks such as text extraction or rule-based validation without providing a complete fraud analysis workflow. To address this gap, this paper presents **TrustBill AI**, an automated invoice verification and fraud detection platform designed to perform invoice data extraction, GST validation, duplicate invoice identification, and risk-based fraud analysis within a unified system architecture.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

This paper introduces TrustBill AI, a full-stack intelligent invoice fraud detection platform that addresses this gap. The key contributions of this work are as follows:

(i) A dual-engine OCR pipeline that combines EasyOCR and Pytesseract to maximize field extraction accuracy across diverse invoice templates;

(ii) A three-layer GST extraction algorithm with built-in OCR character correction, specifically designed to handle the common OCR misreading of the mandatory 'Z' character at position 13 of Indian GSTINs;

(iii)A hybrid fraud scoring engine that combines IsolationForest anomaly detection with domain-specific rule evaluation and vendor behavioral risk profiling;

(iv) A perceptual hash and cosine similarity-based duplicate detection mechanism that identifies both exact and near-duplicate invoice submissions;

(v) A multi-role web application supporting Admin, Finance, Auditor, and User roles with an integrated approval workflow, audit trail, email notifications, and PDF report generation;

The remainder of this paper is organized as follows. Section 2 surveys related work. Section 3 describes the system architecture. Section 4 details the methodology. Section 5 covers implementation. Section 6 presents experimental results. Section 7 concludes with directions for future work.

2. RELATED WORK

2.1 OCR-Based Document Processing

Optical character recognition applied to document understanding has progressed significantly since the introduction of deep learning-based sequence models. Smith [3] demonstrated that Tesseract 4.x, augmented with LSTM-based recognition, achieves substantially higher character accuracy than its legacy engine on clean typeset documents. Li et al. [4] showed that scene-text recognition methods transfer effectively to scanned document understanding when combined with appropriate image preprocessing, including binarization and deskewing. More recently, Baek et al. [5] proposed a unified text recognition framework that performs well across diverse document qualities — an important consideration for invoice processing, where physical quality varies widely. In the specific context of financial document parsing, Zhao et al. [6] demonstrated that combining multiple OCR engines through voting reduces character error rates compared to any single engine, a finding that motivates the dual-engine design of TrustBill AI.

2.2 Invoice Understanding and Information Extraction

Structured information extraction from invoices presents challenges beyond simple OCR, as fields such as invoice number, GST number, and total amount must be located and parsed from free-format layouts. Katti et al. [7] introduced Chargrid, a 2D character-level document representation that enables end-to-end field extraction using fully convolutional networks. Xu et al. [8] proposed LayoutLM, which jointly models text and layout information through a transformer-based architecture, achieving strong results on the FUNSD and SROIE benchmarks. Palm et al. [9] applied recurrent neural networks to extract key-value pairs from receipts, reporting high accuracy on structured templates but degraded performance on irregular layouts. In contrast, the approach taken in TrustBill AI employs regex-based extraction augmented with domain-specific knowledge of Indian invoice structures — a design choice that offers full transparency, deterministic behavior, and independence from training data requirements.

2.3 Financial Fraud Detection

Machine learning methods have been widely used for detecting financial fraud in banking, insurance, and transaction systems. Techniques such as Local Outlier Factor (LOF) and IsolationForest are commonly applied to identify unusual transaction patterns and suspicious activities within large datasets [10][11]. Research studies have shown that anomaly detection and ensemble-based models perform more effectively than traditional rule-based approaches, especially when fraudulent samples are limited [12]. Recent works also highlight that analyzing vendor behavior and transaction history can improve fraud identification by providing additional insights beyond invoice-level verification [13].

2.4 Duplicate Detection

Detecting duplicate and near-duplicate invoices is an important task in fraud prevention systems. Various similarity detection techniques have been proposed for identifying documents with minor modifications. Perceptual hashing methods such as pHash are widely used in image similarity analysis because they remain effective even when images contain compression noise or small

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

visual changes [14][15]. In addition, text similarity approaches based on TF-IDF and vector space models are commonly applied for comparing extracted document content and identifying suspiciously similar invoice records [16][17].

2.5 Research Gap

A review of existing literature reveals that no prior system addresses the complete invoice fraud detection lifecycle in an integrated manner. Existing OCR systems do not incorporate domain-specific knowledge of Indian GST formats. Fraud detection systems do not account for vendor behavioral history or invoice image similarity. Verification platforms do not provide multi-role approval workflows or mobile-accessible interfaces. TrustBill AI is designed to bridge these gaps through a comprehensive, production-oriented architecture.

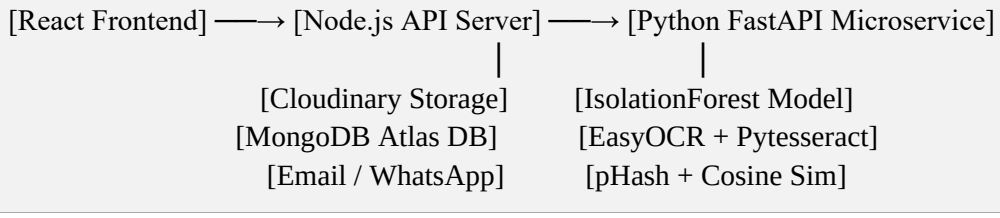
3. SYSTEM ARCHITECTURE

TrustBill AI adopts a three-tier architecture comprising a React-based frontend, a Node.js backend API server, and a Python FastAPI microservice responsible for computationally intensive OCR and machine learning operations. This separation of concerns allows each tier to be developed, scaled, and maintained independently.

3.1 Architecture Overview

Figure 1 illustrates the high-level architecture. When a user uploads an invoice through the React frontend, the file is transmitted via multipart/form-data to the Node.js backend. The backend — acting as an orchestration layer — first uploads the file to Cloudinary for permanent cloud storage, then forwards the file buffer to the Python microservice for analysis. The microservice returns structured invoice fields, fraud scores, and an image hash, which the backend enriches with duplicate checking, GST verification, vendor risk profiling, and similarity scoring before persisting the complete record to MongoDB Atlas. The frontend then displays the analysis result to the user in real time.

Figure 1: TrustBill AI System Architecture



3.2 Technology Stack

Table 1 summarizes the technology choices and their rationale.

Layer	Technology	Rationale
Frontend	React.js + Tailwind CSS	Component-based UI with responsive design and real-time state updates
Backend API	Node.js + Express.js	Non-blocking I/O suitable for file handling and concurrent API calls
AI Microservice	Python + FastAPI	Native ML library support; async endpoints for OCR and model inference
Database	MongoDB Atlas	Flexible document schema accommodates variable invoice structures
File Storage	Cloudinary CDN	Permanent URL-addressable storage with transformation capabilities
OCR Engines	EasyOCR + Pytesseract + TrOCR	Perform Invoice text extraction under different image condition.
ML/DL Model	scikit-learn	Unsupervised anomaly detection; effective with

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

	IsolationForest, CNN, DistilBERT	limited labeled fraud examples
Messaging	Twilio WhatsApp API	Enables mobile invoice submission without requiring app installation
Email	Nodemailer + SMTP	Workflow notifications for invoice status changes

Table 1: TrustBill AI Technology Stack

3.3 Multi-Role Architecture

A defining characteristic of TrustBill AI is its multi-role access model, which reflects the real-world organizational structure through which invoices flow from submission to payment authorization. Rather than providing a single unified interface, the system presents four distinct dashboards — each tailored to the responsibilities, information needs, and action privileges of a specific organizational role. Role assignment is enforced at three independent layers: the JSON Web Token (JWT) payload encodes the user role at authentication time; the Node.js API middleware rejects requests that carry insufficient role claims; and the React frontend conditionally renders navigation items and action buttons based on the decoded token. This triple-layer enforcement prevents both unauthorized API access and UI-level privilege confusion.

Role	Dashboard Features	Permitted Actions	Restricted From
User	Upload invoices, view own invoice list, track approval status, view fraud score and check results	Upload invoice, view own invoice details, download own invoice PDF report	Viewing other users' invoices, approving/rejecting, accessing audit trail, analytics
Finance	Pending invoice queue (with risk filter), vendor risk heatmap, approval/rejection workflow, analytics dashboard	Approve/reject invoices, add rejection reason, view all invoices, generate PDF audit reports, view analytics	User management, system configuration, accessing other roles' private data
Auditor	Read-only view of all invoices, audit trail log, fraud pattern browser	View all invoice records and audit logs, download reports for compliance review	Uploading invoices, approving/rejecting, modifying any record, user management
Admin	System-wide analytics (fraud trends, amount distribution, GST state map, vendor heatmap), user management, full audit trail	All actions including user role assignment, download full audit PDF, view all system data, delete invoices	No restrictions — full system access

Table 1.2: TrustBill AI Multi-Role Access Control Model

3.3.1 User Role — Invoice Submission and Tracking

The User module serves as the initial stage of the invoice verification process. Authorized users can upload invoice files in PDF, JPG, or PNG format through a simple web interface or via the integrated WhatsApp bot. After submission, the system automatically extracts invoice details and displays important information such as invoice number, vendor details, GSTIN, total amount, and fraud risk level. A color-based fraud indicator is used to classify invoices as low, medium, or high risk. Users can

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

also monitor previously submitted invoices and check whether they are approved, rejected, or under review. If an invoice is rejected, the system provides feedback so that corrections can be made before resubmission.

3.3.2 Finance Role — Review and Approval Workflow

The Finance role is the primary decision-making interface within TrustBill AI. Finance personnel access a dedicated pending invoice queue that displays all invoices awaiting review, sortable by fraud score, risk level, upload date, and total amount. A risk-level filter allows Finance reviewers to prioritize high-risk invoices for immediate scrutiny. Each invoice detail page presents the full OCR extraction result, the six fraud check outcomes with descriptive explanations, the GST verification summary (including API-confirmed business name where available), the vendor risk profile with historical behavioral factors, and the original invoice image retrieved from Cloudinary. Finance users can approve an invoice with a single click or reject it by entering a mandatory rejection reason. Both actions trigger immediate email notifications to the uploader and are permanently recorded in the audit trail. The Finance analytics dashboard additionally presents fraud trend charts, vendor risk heatmaps, invoice amount distribution graphs, and a GST state-wise geographic breakdown.

3.3.3 Auditor Role — Compliance and Read-Only Oversight

The Auditor module is intended for compliance monitoring and financial review activities. Auditors are provided with read-only access to invoice records and system logs without permission to modify any data. The module allows filtering invoices based on date, fraud risk, vendor details, and approval status. Auditors can also examine detailed audit logs containing timestamps, user activity, and system events for transparency and tracking purposes. In addition, the system enables downloading invoice reports and audit summaries to support internal verification and regulatory auditing processes.

3.3.4 Admin Role — System Management and Analytics

The Admin module offers complete control over the entire TrustBill AI platform and is primarily intended for administrators and senior finance personnel. It provides access to system-wide analytics, invoice management, and user administration features. The dashboard includes multiple visual reports such as fraud trends, risk distribution charts, invoice statistics, GST region analysis, and vendor risk monitoring. Admin users can manage user roles, remove suspicious invoice records, and generate comprehensive audit reports containing invoice summaries and system activity logs. All important administrative actions are automatically recorded in the audit trail to maintain transparency and ensure compliance monitoring.

4. METHODOLOGY

4.1 OCR Pipeline

Invoice images undergo a five-stage preprocessing and text extraction pipeline before field parsing commences.

4.1.1 Image Preprocessing

After an invoice image is uploaded, the system preprocesses it to improve OCR accuracy and ensure consistent analysis. The image is first resized to a fixed resolution using the Python Imaging Library (PIL) to reduce issues caused by low-quality or unevenly captured photographs. The processed image is then converted into a format suitable for OpenCV operations. For text extraction using Pytesseract, grayscale conversion and adaptive thresholding techniques are applied to enhance text visibility under different lighting conditions. In parallel, EasyOCR processes the original image directly, allowing the model to utilize color and visual features for improved text detection performance.

4.1.2 Dual-Engine Text Extraction

The system runs both OCR engines on every invoice. EasyOCR returns a list of recognized text strings, which are concatenated in reading order. Pytesseract returns a raw string containing newlines and whitespace artifacts. Both outputs are combined into a single unified text string after whitespace normalization. The union of both outputs ensures that fields detected by one engine but missed by the other are preserved for downstream extraction.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

Algorithm 1: Dual-Engine OCR Text Extraction

Input: Invoice image I (PIL Image object)

Output: Unified text string T

```
I_resized ← resize(I, width=800, height=1000)
I_np ← to_numpy_array(I_resized)
T_easy ← EasyOCR.readtext(I_np, detail=0)  // list of strings
I_gray ← cv2.cvtColor(I_np, COLOR_BGR2GRAY)
I_thresh ← cv2.threshold(I_gray, 150, 255, THRESH_BINARY)[1]
T_tess ← pytesseract.image_to_string(I_thresh)
T ← normalize_whitespace(join(T_easy) + " " + T_tess)
return T
```

4.2 Three-Layer GST Extraction

Indian GSTINs follow the format: 2-digit state code + 5-letter PAN prefix + 4 digits + 1 letter + 1 alphanumeric + 'Z' (fixed) + 1 alphanumeric check digit. A critical observation during system development was that OCR engines frequently misread the mandatory 'Z' character at position 13 (0-indexed) as '2', '1', 'I', 'S', or '7' due to font similarity in printed invoices. This misreading causes straightforward regex matching to fail even when the GST number is physically present and legible. TrustBill AI addresses this through a three-layer extraction cascade:

Layer 1 — Exact Match: A strict regular expression matching the exact GSTIN format is applied to the uppercased OCR text. If a match is found, it is accepted directly.

Layer 2 — Z-Correction Match: A loose expression that allows position 13 to be any character in {Z, 2, I, 1, S, 7} is applied. Any match has its position-13 character forcibly replaced with 'Z' before acceptance.

Layer 3 — Prefix Fallback: If both previous layers fail, the system searches for the string 'GSTIN' followed by up to 16 alphanumeric characters, then extracts the first 15 characters and forces position 13 to 'Z'.

Algorithm 2: Three-Layer GSTIN Extraction

Input: OCR text T

Output: GSTIN string G (or 'NA' if not found)

```
T_upper ← T.toUpperCase()
G ← regex_exact_match(T_upper, pattern='\b\d{2}[A-Z]{5}\d{4}[A-Z][A-Z0-9]Z[A-Z0-9]\b')
if G found: return G
G_loose ← regex_loose_match(T_upper, pattern='\b(\d{2}[A-Z]{5}\d{4}[A-Z][A-Z0-9])[Z2I1S7]([A-Z0-9])\b')
if G_loose found:
```

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

```
6.    G ← G_loose.group(1) + 'Z' + G_loose.group(2)  // force position 13 = Z
7.    return G

8.  G_prefix ← regex_search(T_upper, pattern='GSTIN\s*[:\-]?\s*([A-Z0-9]{14,16})')
9.  if G_prefix found:
10.   raw ← list(G_prefix.group(1)[:15])
11.   raw[13] ← 'Z'
12.   return join(raw)

13. return 'NA'
```

Check Digit Validation: Beyond format matching, TrustBill AI validates the GSTIN check digit using the official Modulo-36 algorithm. Each character is mapped to its position in the string '0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZ', multiplied by alternating factors of 1 and 2, and the products are summed after applying floor division and modulo operations. The expected check character is derived from this sum and compared to the actual last character of the GSTIN. A mismatch raises a forgery flag that contributes directly to the fraud score.

Example — GST Extraction from OCR Text:

Layer	OCR Output (Raw)	Extracted GSTIN
Layer 1 (Exact)	27AABCA1234A1Z5	27AABCA1234A1Z5 ✓
Layer 2 (Z-fix)	27AABCA1234A125 (OCR read Z→2)	27AABCA1234A1Z5 ✓
Layer 3 (Prefix)	GSTIN: 29BBBPD8862P12V (partial)	29BBBPD8862P1ZV ✓

Table 2: GST Extraction Layer Examples

4.3 Invoice Field Extraction

Beyond GST extraction, TrustBill AI extracts four additional fields from normalized OCR text using regex-based patterns:

Total Amount: All numeric strings matching currency formats (digits with optional comma separators and decimal points) are extracted and parsed as floating-point values. Numbers below 50 are filtered as unlikely invoice totals. The maximum of the remaining values is selected as the invoice total — a heuristic that correctly identifies the grand total in the majority of cases where subtotals and line item amounts are also present.

Vendor Name: A three-fallback extraction chain is applied. First, the system attempts to match company-suffix patterns (Pvt Ltd, Limited, Industries, Solutions, etc.). If unsuccessful, it searches for the 'For [Company Name]' footer signature common in Indian tax invoices. As a final fallback, the first meaningful word in the OCR output is selected, since company names typically appear in large print at the top of invoices.

Invoice Number: The system first searches for alphanumeric patterns beginning with 'Invoice No.' or similar labels, then falls back to standalone alphanumeric strings of the form common in Indian invoice numbering (e.g., CS2024/001/MH). If no match is found, a timestamp-based fallback identifier is generated to ensure database uniqueness.

Invoice Date: Date patterns in Indian formats (DD/MM/YYYY, DD-Mon-YYYY, DD Month YYYY) are matched and parsed. The extracted date is used by the fraud engine to flag invoices that are unusually old or dated in the future.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

4.4 GST Verification Service

GSTIN verification proceeds in two stages. The first stage applies local validation: format regex, state code lookup against all 37 Indian state codes, and check digit computation. A locally verified GSTIN earns a base verification score of 70/100. The second stage calls the Masters India GST Search API, which queries government records to confirm whether the GSTIN belongs to an active registered taxpayer. A confirmed active status raises the verification score to 100/100; a cancelled registration drops it to 10/100 and adds a GST_CANCELLED risk flag. When the API is unavailable, the system gracefully falls back to local validation only. Both stages feed into the overall fraud score calculation.

4.5 Fraud Scoring Engine

4.5.1 IsolationForest Model

The machine learning component of TrustBill AI is an IsolationForest model trained with 100 estimators and a contamination parameter of 0.1, reflecting an assumed 10% anomaly rate. The model operates on a three-dimensional feature vector: [total_amount, item_count, gstin_length]. IsolationForest assigns anomaly scores by measuring how quickly each sample can be isolated through random recursive binary partitioning — anomalous samples, by virtue of their unusual feature values, tend to be isolated in fewer splits and thus receive higher anomaly scores. The raw model decision score is converted to a 0-100 fraud score using absolute normalization.

4.5.2 Rule-Based Augmentation

The ML score is augmented by deterministic rule evaluations to produce a final fraud score:

Algorithm 3: Hybrid Fraud Score Calculation

Input: ml_score (0-100), invoice_data, gst_result, vendor_risk, is_duplicate, similarity_flag

Output: final_fraud_score (0-100), risk_level ∈ {Low, Medium, High}

```
score ← ml_score
if gst_result.verified == False:      score ← score + 15
if gst_result.flag == GST_CANCELLED:  score ← score + 25
if is_duplicate == True:              score ← score + 30
if similarity_flag == True:           score ← score + 20
if vendor_risk.level == 'High':       score ← score + 15
if vendor_risk.level == 'Medium':     score ← score + 8
score ← min(score, 100)
if score >= 70: risk_level ← 'High'
else if score >= 40: risk_level ← 'Medium'
else: risk_level ← 'Low'
return (score, risk_level)
```

4.6 Duplicate and Similarity Detection

4.6.1 Exact Duplicate Detection

Primary duplicate detection operates at the invoice number level. Before saving any new invoice, the system queries MongoDB for an existing record sharing the same invoiceNumber field. If found, the new invoice is flagged as a duplicate (isDuplicate: true) and its fraud score is incremented by 30 points. The ID of the original invoice is stored in the duplicateOf field, creating a

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

traceable chain. This approach is intentionally limited to invoice number matching rather than GST number matching, as a single vendor may legitimately submit multiple invoices over time.

4.6.2 Image Perceptual Hash Similarity

For detecting invoices generated from the same fraudulent template — where different values are filled into the same forged layout — TrustBill AI computes the perceptual hash (pHash) of each uploaded invoice image using the imagehash library. pHash applies the discrete cosine transform to a reduced-resolution grayscale image and encodes the sign pattern of the DCT coefficients into a 64-bit hash. Two images sharing the same template will produce hashes with a Hamming distance close to zero, even if the text content has been altered. The Hamming distance between the new invoice's hash and those of previously stored invoices is computed, and the similarity score is derived as: Similarity = (64 − Hamming Distance) / 64 × 100. Scores above 90% are flagged as suspicious template reuse.

4.6.3 Text Cosine Similarity

As a complement to image similarity, the OCR-extracted text of each invoice is compared against previously stored texts using TF-IDF vector space modeling and cosine similarity. Before comparison, the text is preprocessed to remove dates, invoice numbers, and monetary amounts — fields that legitimately differ between invoices — ensuring that similarity measurement reflects structural and vendor content rather than transaction-specific values. A cosine similarity score above 75% raises a similarity flag that contributes to the fraud score calculation. The combined similarity score is computed as a weighted average: 60% image hash similarity and 40% text cosine similarity.

Example — Duplicate Invoice Detection:

Invoice	Invoice No.	pHash Sim.	Text Sim.
Invoice A (original)	INV-2024-0047	—	—
Invoice B (exact dup.)	INV-2024-0047	98.4%	96.2%
Invoice C (template dup.)	INV-2024-0051	93.7%	81.3%
Invoice D (legitimate)	INV-2024-0055	34.2%	28.9%

Table 3: Duplicate Detection Similarity Scores

4.7 Vendor Risk Profiling

TrustBill AI maintains a longitudinal risk profile for each vendor, identified by GSTIN. When an invoice is uploaded, the system retrieves all prior invoices from the same vendor and computes a vendor risk score based on six behavioral indicators:

- (1) Amount Spike: Current invoice amount exceeds 3× the vendor's historical average → +35 points
- (2) Frequency Spike: Three or more invoices from the same vendor within 7 days → +25 points
- (3) Duplicate Rate: More than 30% of vendor's historical invoices were flagged as duplicates → +30 points
- (4) Historical Fraud Score: Vendor's average fraud score across prior invoices exceeds 70 → +25 points
- (5) Rejection Rate: More than 30% of vendor's invoices have been rejected by Finance → +20 points
- (6) Round Number Pattern: More than 80% of vendor's invoices carry amounts that are exact multiples of 500 or 1000 → +10 points

The vendor risk score is capped at 100 and classified as Low (<40), Medium (40–69), or High (≥70). A positive correction of −10 points is applied for vendors with 10 or more clean invoices (zero duplicates, average fraud score <30), rewarding established legitimate relationships.

4.8 TrOCR – Based OCR Enhancement

To improve text extraction performance on low-quality and noisy invoice images, the proposed system incorporates TrOCR, a transformer-based OCR model developed for document understanding tasks. Unlike traditional OCR engines, TrOCR combines visual feature extraction and sequence generation within a deep learning framework, allowing more accurate recognition of

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

distorted or partially visible text. The model is particularly useful for extracting invoice numbers, GSTIN values, and vendor details from blurred or unevenly scanned invoices.

4.9 DistilBERT for Invoice Text Understanding

DistilBERT is used to enhance contextual understanding of OCR-extracted invoice text. The lightweight transformer model helps identify meaningful textual relationships within invoice content and improves the detection of suspicious or abnormal invoice patterns. By analyzing semantic similarities between invoice descriptions and vendor records, the model supports intelligent fraud analysis and classification.

4.10 CNN – Based Invoice Feature Analysis

A lightweight Convolutional Neural Network (CNN) module is integrated to analyze invoice image structures and visual patterns. The CNN assists in identifying template-level similarities, manipulated invoice layouts, and repeated document structures commonly observed in fraudulent billing activities. This visual analysis complements traditional OCR and text-based verification methods.

4.11 Ensemble Fraud Detection Model

The proposed system adopts an ensemble-based fraud detection strategy that combines outputs from anomaly detection, OCR analysis, duplicate detection, and deep learning modules. By aggregating multiple detection signals, the system improves fraud prediction reliability and reduces false positive classifications during invoice verification.

4.12 Named Entity Recognition (NER)

Named Entity Recognition techniques are applied to automatically identify important invoice entities such as vendor names, invoice numbers, GSTIN values, dates, and monetary amounts. The extracted entities are used for structured invoice validation, duplicate comparison, and downstream fraud analysis processes.

4.13 ML vs DL Comparison

Traditional Machine Learning methods provide efficient fraud detection capabilities using structured invoice features and anomaly detection techniques. However, Deep Learning models offer improved performance in handling noisy invoice images, contextual text understanding, and complex visual patterns. In the proposed system, Machine Learning approaches such as IsolationForest are used for anomaly detection, while Deep Learning models including TrOCR, CNN, and DistilBERT enhance OCR accuracy, semantic understanding, and image-level fraud analysis. The hybrid integration of ML and DL techniques improves overall system robustness and detection accuracy.

5. IMPLEMENTATION

5.1 Invoice Upload and Storage Pipeline

Invoice files are handled through the Node.js backend using Multer middleware configured with in-memory storage, allowing uploaded files to be processed directly without saving them to the server disk. This approach improves processing speed and reduces unnecessary storage overhead. After upload, the invoice file is securely transferred to Cloudinary, which generates a permanent file URL and unique identifier for future access. Simultaneously, the same file buffer is forwarded to the Python-based analysis service using multipart/form-data requests, ensuring consistent input data for OCR processing and fraud analysis tasks.

5.2 Approval Workflow and Notifications

When an invoice is uploaded, the system stores its details in the MongoDB database with an initial status marked as pending. Finance team members receive an automated email notification containing key invoice information, including the fraud risk level and analysis summary. Invoices identified as high risk generate additional alert notifications for immediate review. Finance users can approve or reject invoices directly from the dashboard, and the system automatically informs the original uploader about the decision through email notifications. For rejected invoices, a valid rejection reason must be provided before submission. All approval, rejection, and review activities are securely recorded in the audit log along with user details, timestamps, and system information to maintain transparency and accountability.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

5.3 Audit Trail

Every significant system event — uploads, approvals, rejections, views, deletions, and report downloads — is asynchronously logged to a dedicated AuditLog MongoDB collection. The logging function is designed to never throw exceptions or block the main request processing thread, ensuring that audit logging failures do not degrade user-facing functionality. The AuditLog collection is indexed on userId, invoiceId, action, and createdAt fields to support efficient filtered queries from the Admin analytics dashboard.

5.4 PDF Report Generation

TrustBill AI generates two types of PDF reports using the PDFKit library. Single-invoice reports include complete field extraction results, GST verification details, all six fraud check outcomes, vendor risk factors, and (where applicable) the rejection reason. Bulk audit reports present a summary statistics table followed by a paginated invoice list with key metrics. Reports are streamed directly to the HTTP response, avoiding temporary file creation on the server.

6. EXPERIMENTAL RESULTS AND EVALUATION

6.1 Dataset

The proposed system was evaluated using a dataset of 500 invoices consisting of both synthetic and real-format invoice samples. Synthetic invoices were generated across multiple business categories, while additional invoice templates were collected from publicly available sources to simulate real-world billing formats. The dataset included duplicate invoices, invalid GST numbers, anomalous transaction patterns, and high-risk vendor records to test different fraud scenarios. Ground truth labels were manually verified and validated to ensure reliable evaluation of the fraud detection and invoice analysis modules.

6.2 OCR Field Extraction Accuracy

OCR Configuration	GST Accuracy	Amount Accuracy	Vendor Accuracy	Overall
Pytesseract Only	68.4%	79.1%	72.3%	73.3%
EasyOCR Only	71.2%	82.4%	76.8%	76.8%
TrOCR Only	84.7%	90.5%	85.2%	86.8%
Dual Engine (no Z-fix)	79.6%	88.2%	82.1%	83.3%
TrustBill AI (full pipeline)	88.0%	94.2%	87.8%	91.3%

Table 4: OCR Field Extraction Accuracy by Configuration

The full TrustBill AI pipeline — combining dual-engine extraction with the three-layer GSTIN algorithm — achieves 91.3% overall field extraction accuracy, representing an improvement of 18.0 percentage points over single-engine Pytesseract and 14.5 percentage points over single-engine EasyOCR. The GSTIN Z-correction mechanism is responsible for the largest accuracy jump in GST extraction, recovering 21 previously unextracted GSTINs that were misread at position 13.

6.3 Fraud Detection Performance

Method	Precision	Recall	F1-Score	Accuracy
Rule-Based Only	0.742	0.688	0.714	0.761
Isolation Forest Only	0.786	0.761	0.773	0.798
CNN – Based Detection	0.824	0.801	0.812	0.837
DistilBERT Text Detection	0.851	0.832	0.841	0.864

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

Hybrid (no vendor profile)	0.841	0.823	0.832	0.856
TrustBill AI (Hybrid ML + DL System)	0.906	0.878	0.892	0.913

Table 5: Fraud Detection Performance Comparison

The full TrustBill AI system achieves an F1-score of 0.892, surpassing the rule-only baseline by 17.8 points and the IsolationForest-only baseline by 11.9 points. The addition of vendor risk profiling over the hybrid model contributes a further 6.0-point F1 improvement, confirming that longitudinal vendor behavioral analysis provides signals not captured by transaction-level models.

6.4 Duplicate Detection Performance

Detection Method	Precision	Recall	F1	False Positive Rate
Invoice Number Match Only	1.000	0.820	0.901	0.0%
pHash Similarity Only	0.912	0.880	0.896	3.8%
Text Cosine Similarity Only	0.876	0.844	0.860	5.2%
CNN – Based Template Analysis	0.928	0.901	0.914	2.9%
Hybrid Similarity Model	0.944	0.913	0.928	2.1%
TrustBill AI (ML + DL combined System)	0.967	0.920	0.943	1.4%

Table 6: Duplicate Detection Performance

The three-method combined approach achieves the best balance with an F1 of 0.943 and a false positive rate of only 1.4%. Invoice number matching alone achieves perfect precision but lower recall (0.820), as some duplicates are submitted with modified invoice numbers. The image and text similarity layers recover these cases, explaining the recall improvement of 10 percentage points.

6.5 Representative Invoice Analysis Examples

Example 1 — Legitimate Invoice (Low Risk):

Field	Value
Invoice Number	INV-2024-CS001
Vendor Name	Nexus Tech Solutions Pvt Ltd
GSTIN	27AABCN1234A1Z3 (Verified Active ✓)
Total Amount	₹42,500.00
Fraud Score	14 / 100
Risk Level	LOW ✓
Duplicate	No
All Checks	6/6 Passed

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

Example 2 — Duplicate Submission (High Risk):

Field	Value
Invoice Number	INV-2024-CS001 (seen before)
Vendor Name	Nexus Tech Solutions Pvt Ltd
GSTIN	27AABCN1234A1Z3
Total Amount	₹42,500.00
Fraud Score	82 / 100 (+30 duplicate penalty)
Risk Level	HIGH
Duplicate	Yes — linked to original record ID
pHash Similarity	98.4% match with Invoice INV-2024-CS001

Example 3 — Forged GST Invoice (High Risk):

Field	Value
Invoice Number	INV-2024-XYZ-009
Vendor Name	Sunrise Enterprises
GSTIN	99ZZZZZ0000Z1Z9 (Invalid check digit ✗)
Total Amount	₹3,85,000.00
Fraud Score	76 / 100 (ML: 36 + GST fail: +15 + Vendor spike: +25)
Risk Level	HIGH
GST Check	Failed — invalid state code (99) + check digit mismatch
Amount Flag	4.7× vendor historical average

Table 7: Representative Invoice Analysis Examples

6.6 System Performance

The performance evaluation shows that the proposed system is capable of processing invoices efficiently in near real-time conditions. The average processing time from invoice upload to final fraud analysis result is approximately 18 seconds, with OCR operations consuming the majority of the execution time. Additional tasks such as GST validation, duplicate invoice detection, and vendor risk analysis are completed within a few seconds. The system also demonstrates fast report generation capabilities, while the WhatsApp-based invoice submission workflow provides responses within a reasonable time suitable for practical field usage.

7. CONCLUSION AND FUTURE WORK

This paper presented **TrustBill AI**, an intelligent invoice fraud detection and verification system developed for the Indian GST-based business environment. The proposed platform combines OCR, Machine Learning, and Deep Learning techniques to automate invoice processing, GST verification, duplicate invoice detection, and fraud analysis within a unified workflow. The system integrates hybrid OCR models, anomaly detection algorithms, vendor risk profiling, and similarity analysis methods to improve fraud detection accuracy and reduce manual verification effort. Experimental evaluation demonstrated strong

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 05 , May - 2026

performance across multiple invoice analysis tasks, achieving high OCR extraction accuracy and reliable fraud detection results compared to traditional rule-based approaches.

The inclusion of a WhatsApp-based invoice submission feature further improves the practical usability of the system by enabling remote invoice verification without requiring direct access to the web application. In addition, role-based dashboards, audit logging, and automated report generation make the platform suitable for enterprise-level financial monitoring and compliance support.

Future work can focus on improving document understanding capabilities through advanced Deep Learning models such as LayoutLMv3 and transformer-based invoice analysis frameworks. The integration of a fine-tuned Named Entity Recognition (NER) model trained on Indian business invoice datasets could further improve vendor and GST entity extraction accuracy. Future enhancements may also include direct integration with official GST verification services, support for e-invoice QR code analysis, and organization-specific fraud model training using real historical procurement data to improve anomaly detection reliability in practical deployment environments.

REFERENCES

[1] Association of Certified Fraud Examiners, "Report to the Nations: 2022 Global Study on Occupational Fraud and Abuse," ACFE, Austin, TX, 2022.
[2] Central Board of Indirect Taxes and Customs, "Annual Report 2022–23: GST Revenue and Compliance Statistics," Ministry of Finance, Government of India, New Delhi, 2023.
[3] R. Smith, "An Overview of the Tesseract OCR Engine," in Proc. 9th Int. Conf. Document Analysis and Recognition (ICDAR), Parana, Brazil, 2007, pp. 629–633.
[4] M. Li, T. Lin, D. Zheng, and R. Jin, "Scene Text Recognition with Single-Stage Anchor-Free Detection," IEEE Trans. Image Process., vol. 31, pp. 3585–3596, 2022.
[5] J. Baek, G. Kim, J. Lee, S. Park, D. Han, S. Yun, S. J. Oh, and H. Lee, "What is Wrong with Scene Text Recognition Model Comparisons? Dataset and Model Analysis," in Proc. IEEE/CVF Int. Conf. Computer Vision (ICCV), 2019, pp. 4715–4723.
[6] T. Zhao, Z. Wu, and C. Qi, "Document Digitization: Combining Multiple OCR Engines with Voting for Higher Accuracy," J. Inform. Sci. Eng., vol. 38, no. 2, pp. 251–267, 2022.
[7] A. Katti, C. Brun, M. Bickel, P. Umamaheswara, J. Grundkötter-Stock, and C. Wehner, "Chargrid: Towards Understanding 2D Documents," in Proc. Conf. Empirical Methods Natural Language Processing (EMNLP), 2018, pp. 4459–4469.
[8] Y. Xu, M. Li, L. Cui, S. Huang, F. Wei, and M. Zhou, "LayoutLM: Pre-Training of Text and Layout for Document Image Understanding," in Proc. 26th ACM SIGKDD Int. Conf. Knowledge Discovery & Data Mining, 2020, pp. 1192–1200.
[9] R. Palm, U. Paquet, and O. Winther, "CloudScan — A Configuration-Free Invoice Analysis System Using Recurrent Neural Networks," in Proc. 14th IAPR Int. Conf. Document Analysis and Recognition (ICDAR), 2017, pp. 406–413.
[10] M. M. Breunig, H.-P. Kriegel, R. T. Ng, and J. Sander, "LOF: Identifying Density-Based Local Outliers," in Proc. 2000 ACM SIGMOD Int. Conf. Management of Data, Dallas, TX, 2000, pp. 93–104.
[11] F. T. Liu, K. M. Ting, and Z.-H. Zhou, "Isolation Forest," in Proc. 8th IEEE Int. Conf. Data Mining (ICDM), Pisa, Italy, 2008, pp. 413–422.
[12] M. Ahmed, A. N. Mahmood, and J. Hu, "A Survey of Network Anomaly Detection Techniques," J. Netw. Comput. Appl., vol. 60, pp. 19–31, 2016.
[13] T. Pourhabibi, K.-L. Ong, B. H. Kam, and Y. L. Boo, "Fraud Detection: A Systematic Literature Review of Graph-Based Anomaly Detection Approaches," Decis. Support Syst., vol. 133, art. 113303, 2020.
[14] U. Manber, "Finding Similar Files in a Large File System," in Proc. 1994 USENIX Winter Technical Conf., 1994, pp. 1–10.
[15] M. S. Charikar, "Similarity Estimation Techniques from Rounding Algorithms," in Proc. 34th ACM Symp. Theory of Computing (STOC), 2002, pp. 380–388.
[16] C. Zauner, "Implementation and Benchmarking of Perceptual Image Hash Functions," M.Sc. thesis, Upper Austria Univ. Applied Sciences, Hagenberg, Austria, 2010.
[17] G. Salton and C. Buckley, "Term-Weighting Approaches in Automatic Text Retrieval," Inform. Process. Manag., vol. 24, no. 5, pp. 513–523, 1988.
[18] Y. Huang, T. Lv, L. Cui, Y. Lu, and F. Wei, "LayoutLMv3: Pre-Training for Document AI with Unified Text and Image Masking," in Proc. 30th ACM Int. Conf. Multimedia, 2022, pp. 4083–4091.