SCARA: An Intelligent Agent Framework for Real-Time Fraud Detection in Indian Public Sector Supply Chains Using Synthetic Transaction Simulation and Large Language Model-Powered Risk Scoring

Pushpendra Neniwal[1], Dr. Vikas Kapoor[2]
[1,2]*Department of Production & Industrial Engineering, MBM University, Jodhpur, Rajasthan, India*

Abstract—**Fraudulent procurement practices within public sector supply chains represent a deeply entrenched and economically damaging challenge in large emerging economies, where the volume of transactions is enormous and oversight is unevenly distributed across administrative layers. This manuscript introduces SCARA (Supply Chain Anomaly and Risk Assessment Agent), a fully integrated intelligent framework that brings together synthetic data generation, multi-category fraud simulation, geospatial logistics modelling, and a large language model (LLM)-powered analytical layer to identify and explain five canonical categories of procurement fraud: Invoice Inflation, Ghost Shipments, Product Mismatches, Product Diversion, and Shipment Delay.**

The synthetic dataset was built using a statistically grounded simulation pipeline that models 50 suppliers, 20 government buyers, and 100 product types distributed across ten major Indian cities, with geospatial logistics computed using the Haversine formula. A controlled fraud injection rate of 15% introduces labelled anomalies drawn from real-world irregularities documented by Indian audit bodies. The detection layer uses Meta LLaMA 3.1 8 B Instant, served through the Groq low-latency inference API, to produce structured JSON risk reports containing an overall risk score, an executive summary, step-by-step reasoning, a hypothesized fraud category, and recommended investigative actions. An interactive Streamlit dashboard allows practitioners to explore and query the transaction data in real time. The results show that SCARA produces coherent, explainable, and practically useful fraud assessments, establishing a strong foundation for human-in-the-loop AI governance in public procurement.

Index Terms—**supply chain fraud detection; synthetic data generation; large language models; public procurement; Indian logistics; anomaly detection; explainable AI; LLaMA; Groq inference; Streamlit dashboard**

I. INTRODUCTION

Public sector procurement is the operational engine of government service delivery. In India alone, government expenditure on goods and services exceeded INR 30 lakh crore in the financial year 2023-2024, flowing through a complex network of central ministries, state departments, district administrations, and public sector undertakings. This scale creates an environment where fraud flourishes: large transaction volumes, fragmented oversight, and inconsistent audit trails across administrative tiers.

Traditional audit-based approaches to fraud detection are fundamentally retrospective in nature. By the time a financial irregularity is identified and formally reported, the misappropriated funds are often unrecoverable. This challenge is compounded by the heterogeneous nature of supply chain data, which span procurement orders, logistics records, supplier credentials, and product master files, all of which resist simple rule-based monitoring. What is needed is an intelligent system that can synthesise these varied data streams, surface anomalies as they develop, and explain its reasoning in terms that investigators can actually use.

Recent advances in large language models have opened up genuinely promising paths. Unlike conventional machine learning classifiers that output only probability scores, LLMs can articulate their reasoning clearly and in a structured language. This explainability is not a luxury in the procurement context; it is essential because fraud analysts must justify their investigative decisions to oversight

bodies, legal teams and parliamentary committees. Simultaneously, deploying LLMs in sensitive government contexts raises legitimate questions regarding inference consistency, prompt reliability and domain-specific accuracy.

This study addresses these questions through the design and evaluation of a SCARA. This study makes four distinct contributions to the literature. First, it presents a replicable synthetic data generation pipeline calibrated to Indian procurement norms. Second, it describes a fraud injection methodology that covers five real-world fraud typologies. Third, it proposes and evaluates an LLM-powered risk assessment module that generates structured and explainable fraud reports. Fourth, it delivers an accessible Streamlit dashboard that places this capability in the hands of practitioners who may not have a machine learning background.

II. BACKGROUND AND RELATED WORK

2.1 Fraud in Indian Public Procurement

Procurement fraud in Indian public institutions has been extensively documented by the Comptroller and Auditor General of India, whose compliance audit reports chronicle recurring weaknesses: inflated invoices, payments for goods that were never delivered, substitution of inferior materials, and misappropriation of development funds (CAG, 2023). The introduction of the Government e-Marketplace has improved transparency in some segments, but manual procurement processes persist across sub-national bodies, leaving audit trails fragmented and vulnerable to corruption. Krishnaswamy and Mehra (2021) estimated that procurement-related corruption accounts for approximately 2.3% of annual GDP losses in India, a figure that underlines the urgency of scalable automated detection.

2.2 Machine Learning Approaches to Supply Chain Anomaly Detection

The machine learning literature on supply chain anomaly detection is well developed, although most studies target private sector logistics optimization rather than public procurement fraud. Graph neural network approaches have demonstrated strong capabilities in modelling supplier-buyer relationship networks (Li et al., 2021; Wang et al., 2022), whereas supervised ensemble methods, such as XGBoost and CatBoost, remain widely used baselines for tabular fraud classification (Chen and Guestrin, 2016; Prokhorenkova et al., 2018). Variational autoencoders have been applied for unsupervised anomaly detection in high-dimensional transaction data (An and Welling, 2014; An and Cho, 2015). A consistent limitation of all these approaches is that they produce scores without explanations, which limits their practical usefulness when investigators must justify their decisions in formal settings.

2.3 Large Language Models for Structured Analytical Reasoning

The emergence of instruction-tuned large language models, including GPT-4 (OpenAI, 2023), Claude (Anthropic, 2024), Gemini (Google DeepMind, 2024), and the open-weight LLaMA series (Touvron et al., 2023), has created new possibilities for structured reasoning over domain-specific data sets. CoT prompting has significantly improved the reliability and verifiability of LLM-generated analytical outputs (Wei et al., 2022), whereas ReAct-style reasoning traces add transparency to multi-step inference processes (Yao et al., 2023). The JSON-format output constraints enforced at the API level further improve the consistency of production systems (Willard and Louf, 2023). The Groq inference platform, which delivers LLaMA 3.1 at sub-100-millisecond latency through custom Language Processing Unit hardware, makes real-time per-transaction analysis viable at scale.

2.4 Synthetic Data for Fraud Research

The scarcity of publicly available labelled fraud datasets is one of the most persistent obstacles in fraud detection research. Real procurement datasets are rarely shared publicly because of privacy constraints, ongoing legal proceedings, and national security concerns. Synthetic data generation has become the standard approach, with the PaySim simulator (Lopez-Rojas and Axelsson, 2016) for mobile payment fraud and GAN-based synthetic financial datasets (Fiore et al., 2019; Goodfellow et al., 2014) being widely cited. The SCARA approach uses Python Faker configured for the Indian locale, combined with controlled fraud injection, to produce a corpus that is statistically plausible, fully labelled, and shareable without any privacy concerns.

III. SYSTEM DESIGN AND METHODOLOGY

3.1 Architecture Overview

SCARA is built from four integrated modules: a synthetic entity and transaction generation pipeline, a controlled fraud injection engine, a geospatial logistics simulator, and an LLM-powered risk assessment agent with an interactive Streamlit frontend. The system was implemented in Python 3.11 using Pandas for data handling, NumPy for numerical computation, Faker for locale-aware entity generation, the OpenAI client library configured for the Groq endpoint, and Streamlit for the visualization dashboard.

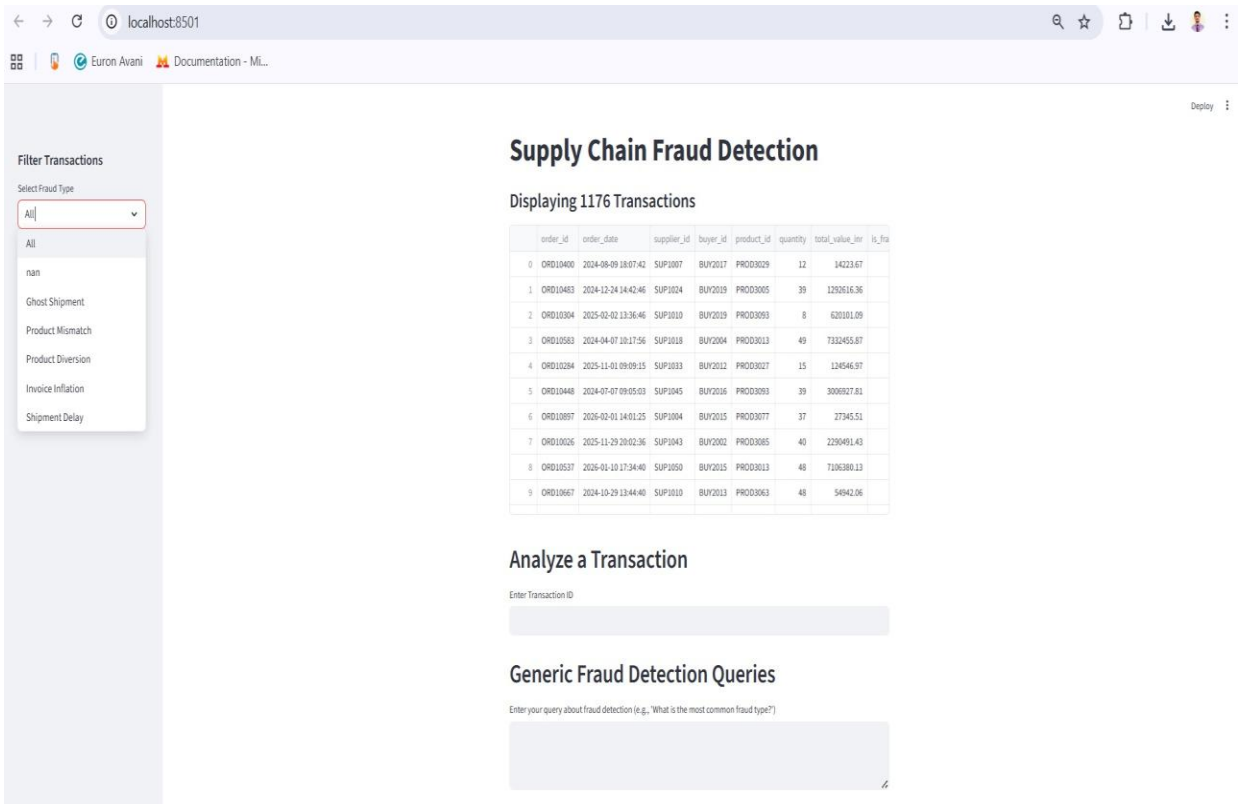

Figure 5. SCARA Streamlit dashboard: the main interface showing the full transaction corpus of 1,176 records with fraud-type filter sidebar and column-level transaction data.

3.2 Synthetic Entity and Transaction Generation

The dataset is configured with 50 suppliers, 20 government buyers, and 100 products across five procurement categories: IT Hardware (INR 20,000-150,000), Office Furniture (INR 5,000-50,000), Medical Supplies (INR 500-10,000), Automotive Parts (INR 1,000-80,000), and Stationery (INR 100-2,000). The price and weight ranges were obtained from publicly available government rate contract schedules. Entity names were generated using Faker configured for the en_IN locale, ensuring that company names, buyer department designations, and city references matched the real Indian administrative conventions. Supplier reliability scores were uniformly sampled from [0.70, 1.00].

The geospatial distribution is anchored to ten major metropolitan hubs: Delhi, Mumbai, Bangalore, Chennai, Kolkata, Hyderabad, Ahmedabad, Pune, Jaipur, and Lucknow. Each city was mapped to its standard six-digit pin code and WGS84 latitude-longitude coordinates. The baseline transaction corpus consists of 1,000 normal records, with order quantities drawn from [1, 50], and values incorporating small multiplicative noise around the catalogue price to reflect natural contract variations.

Figure 1 | Entity & Dataset Configuration — main.py

```python
# ── Dataset Configuration ──────────────────────────────────────
NUM_SUPPLIERS           = 50
NUM_BUYERS              = 20
NUM_PRODUCTS            = 100
NUM_NORMAL_TRANSACTIONS = 1000
FRAUD_INJECTION_RATE    = 0.15      # 15 % of total transactions

fake = Faker('en_IN')               # Indian locale for realistic names

# Price range (INR) and weight range (kg) per procurement category
product_categories = {
    'IT Hardware'     : {'price_range': (20000, 150000), 'weight_range': (2,   15 )},
    'Office Furniture': {'price_range': ( 5000,  50000), 'weight_range': (10,  100)},
    'Medical Supplies': {'price_range': (  500,  10000), 'weight_range': (0.5,  20)},
    'Automotive Parts': {'price_range': ( 1000,  80000), 'weight_range': (1,   200)},
    'Stationery'      : {'price_range': (  100,   2000), 'weight_range': (0.1,   5)},
}

# ── Supplier Generation ────────────────────────────────────────
def generate_entities():
    suppliers = []
    for i in range(NUM_SUPPLIERS):
        city = random.choice(CITIES)
        suppliers.append({
            'supplier_id'      : f'SUP{1001 + i}',
            'supplier_name'    : fake.company(),
            'city'             : city,
            'state'            : pincode_data[city]['state'],
            'reliability_score': round(random.uniform(0.7, 1.0), 2),
        })

    buyers = []
    for i in range(NUM_BUYERS):
        city = random.choice(CITIES)
        buyers.append({
            'buyer_id'   : f'BUY{2001 + i}',
            'buyer_name': f'{fake.word().capitalize()} Ministry Dept. – {city}',
            'city'       : city,
            'state'      : pincode_data[city]['state'],
        })

    return pd.DataFrame(suppliers), pd.DataFrame(buyers)
```

Figure 1. Entity and dataset configuration: product category definitions with INR price ranges and the generate_entities() function for supplier and buyer creation (main). py).

3.3 Fraud Injection Methodology

A 15% fraud injection rate was applied to the combined dataset. Each fraudulent record is derived from a sampled normal transaction and modified according to one of five typologies drawn with equal probability. Table 1 describes each fraud type, its real-world motivation, and the specific mechanism used for injection.

Fraud Typology	Real-World Rationale	Injection Mechanism
Invoice Inflation	Vendor inflates the invoice value to extract funds above the contracted price, often with buyer collusion	Multiply total_value_inr by U(1.5, 3.0), simulating 50 to 200 percent inflation above the baseline price
Ghost Shipment	Payment processed for goods never manufactured, shipped, or received; common in collusive schemes	Set shipment_status to Lost In Transit; assign null values to route_distance_km and transit_time_hours
Product Mismatch	High-value invoice raised against a premium category but a low-cost substitute is delivered, exploiting weak goods inspection	Retain original high total_value_inr; replace product_id with a randomly selected Stationery item

Product Diversion	Goods legitimately shipped but rerouted to an unauthorised destination for resale or stockpiling by internal actors	Replace destination city with a random alternative; set shipment_status to Diverted
Shipment Delay	Transit times anomalously extended, suggesting misuse of goods in transit, unauthorised offloading, or deliberate obstruction	Label is_fraud as 1 with fraud_type set to Shipment Delay; standard logistics simulation applied without route modification

Table 1. Fraud typologies, real-world motivations, and programmatic injection mechanisms in the SCARA synthetic dataset.

Figure 2 | Fraud Injection Engine — main.py

```python
# ── Fraud Injection ──────────────────────────────────────────────
def inject_fraud(normal_transactions_df, products_df):
    num_fraud = int(
        len(normal_transactions_df) * FRAUD_INJECTION_RATE
        / (1 - FRAUD_INJECTION_RATE)
    )
    fraudulent_transactions = []

    for i in range(num_fraud):
        # Randomly choose one of five fraud typologies
        fraud_choice = random.choice([
            'Invoice Inflation',
            'Ghost Shipment',
            'Product Mismatch',
            'Product Diversion',
            'Shipment Delay',
        ])

        # Clone a normal transaction and corrupt it
        txn                 = normal_transactions_df.sample(1).iloc[0].to_dict()
        txn['order_id']     = f'ORD{20001 + i}'
        txn['is_fraud']     = 1
        txn['fraud_type']   = fraud_choice

        if fraud_choice == 'Invoice Inflation':
            # Inflate invoice by 50 – 200 %
            txn['total_value_inr'] *= random.uniform(1.5, 3.0)

        elif fraud_choice == 'Product Mismatch':
            # Keep premium invoice value; swap product to cheap Stationery
            cheap_products = products_df[
                products_df['category'] == 'Stationery'
            ]
            if not cheap_products.empty:
                txn['product_id'] = cheap_products.sample(1).iloc[0]['product_id']

        # Ghost Shipment / Product Diversion handled in logistics simulation
        fraudulent_transactions.append(txn)

    return pd.DataFrame(fraudulent_transactions)
```

Figure 2. Fraud injection engine: the inject_fraud() function showing Invoice Inflation multiplication and Product Mismatch product substitution logic (main.py).

3.4 Geospatial Logistics Simulation

A logistics record was generated for every transaction by computing the great-circle distance between the origin and destination cities using the Haversine formula. The transit time was derived by dividing the route distance by the average Indian freight truck speed of 45 km/h, with an additional random delay from U(0, 24) h to capture real-world variability from traffic, tolls, and loading delays. Ghost Shipment records receive a status of Lost In Transit with null logistics fields. Product Diversion records are rerouted to a randomly chosen alternative city and flagged as diverted. All other transactions received a delivered status.

Figure 3 | Haversine Geospatial Logistics Simulation — main.py

```python
# ── Haversine Formula — great-circle distance ──────────────────────
def haversine_distance(lat1, lon1, lat2, lon2):
    R    = 6371                                  # Earth radius in km
    dlat = np.radians(lat2 - lat1)
    dlon = np.radians(lon2 - lon1)
    a    = (np.sin(dlat / 2) ** 2
            + np.cos(np.radians(lat1))
            * np.cos(np.radians(lat2))
            * np.sin(dlon / 2) ** 2)
    return R * 2 * np.arctan2(np.sqrt(a), np.sqrt(1 - a))

# ── Logistics Record Generation ────────────────────────────────────
AVG_TRUCK_SPEED_KMH = 45

for _, row in df.iterrows():
    origin_city = row['city_origin']
    dest_city   = row['city_dest']

    # Product Diversion: reroute to an unauthorised destination
    if row['fraud_type'] == 'Product Diversion':
        dest_city = random.choice(
            [c for c in CITIES if c != dest_city]
        )

    if row['fraud_type'] == 'Ghost Shipment':
        # No physical movement; logistics fields are null
        status, distance_km, transit_hours = 'Lost In Transit', None, None

    else:
        distance_km   = haversine_distance(
            *pincode_data[origin_city]['coords'],
            *pincode_data[dest_city]['coords'],
        )
        transit_hours = round(
            distance_km / AVG_TRUCK_SPEED_KMH + random.uniform(0, 24), 2
        )
        status = ('Diverted' if row['fraud_type'] == 'Product Diversion'
                  else 'Delivered')
```

Figure 3. Haversine geospatial logistics simulation: distance computation and conditional status assignment for Ghost Shipment and Product Diversion typologies (main.py).

3.5 SCARA LLM Risk Assessment Agent

3.5.1 Model and Inference Infrastructure

SCARA uses Meta LLaMA 3.1 8B Instant, which is accessed via the Groq inference API through an OpenAI-compatible client interface. Groq's LPU-based hardware delivers consistently sub-100-millisecond first-token latency, making real-time per-transaction analysis feasible within an interactive dashboard. The temperature is fixed at 0.0 for deterministic outputs, and JSON response formatting is enforced at the API level to guarantee syntactically valid and machine-parseable reports on every call.

3.5.2 Prompt Design and Output Schema

The SCARA prompt was divided into four sections. The Role section establishes the agent as a specialist fraud analyst with expertise in the Indian procurement standards. The Rules section operationalizes each fraud typology as an explicit detection heuristic grounded in the dataset schema as follows: The Data section serializes the transaction records as formatted JSON strings. The Output section mandates a strict five-field JSON response: risk_score (Low, Medium, High, or Critical), summary (a single-sentence executive synopsis), reasoning_steps (an ordered list of identified anomalies with interpretations), hypothesized_fraud_type, and recommended_actions (two to three concrete investigative steps).

Figure 4 | SCARA LLM Agent — Prompt & API Integration — main.py

```
# ── SCARA Prompt Template ──────────────────────────────────────────
PROMPT_TEMPLATE = """
### ROLE ###
You are SCARA, an expert fraud detection agent specialising in
Indian supply chain logistics and public procurement.

### DETECTION RULES ###
1. Invoice Inflation  — total_value_inr > 150 % of (price_inr × quantity)
2. Ghost Shipment     — shipment_status = 'Lost In Transit'; NaN logistics
3. Product Mismatch   — high invoice value but low-cost product category
4. Product Diversion  — shipment_status = 'Diverted'
5. Shipment Delay     — anomalous transit_time_hours for route distance

### TRANSACTION DATA ###
{transaction_data}

### OUTPUT FORMAT (strict JSON only) ###
{{
  "risk_score"             : "Low | Medium | High | Critical",
  "summary"                : "<one-sentence executive synopsis>",
  "reasoning_steps"        : ["<step 1>", "<step 2>", "..."],
  "hypothesized_fraud_type": "<typology>",
  "recommended_actions"    : ["<action 1>", "<action 2>", "<action 3>"]
}}
"""

# ── Groq API Call ──────────────────────────────────────────────────
def analyze_transaction_with_scara(transaction_row):
    prompt   = PROMPT_TEMPLATE.format(
        transaction_data=transaction_row.to_json(indent=4)
    )
    response = client.chat.completions.create(
        model           = "llama-3.1-8b-instant",
        messages        = [
            {"role": "system", "content": "Output valid JSON only."},
            {"role": "user",   "content": prompt},
        ],
        response_format = {"type": "json_object"},
        temperature     = 0.0,     # deterministic output
    )
    return json.loads(response.choices[0].message.content)
```

Figure 4. SCARA LLM agent: structured prompt template with detection rules and the Groq API call with JSON output enforcement (main.py).

3.6 Interactive Streamlit Dashboard

A practitioner-facing interface built in Streamlit allows users to filter the transaction corpus by fraud type, inspect individual records as structured JSON files, and submit natural language queries regarding aggregate fraud patterns. When a transaction identifier is entered, the system retrieves the record, constructs a tailored prompt, calls the Groq API, and renders a structured risk report in the browser. A Seaborn-Matplotlib visualization layer generates fraud distribution charts in response to the user queries. The interface separates filtering, inspection, LLM analysis, and visualization into distinct workflow stages, thereby reducing the cognitive load for non-technical analyst users.

IV. RESULTS AND DISCUSSION

4.1 Dataset Composition

The final master dataset contained 1,176 transactions: 1,000 normal and approximately 176 fraudulent, distributed roughly equally across the five types. Table 2 summarizes the composition with percentages drawn from the actual dataset shown in Figures 6 and 7. The dataset was shuffled before export to eliminate positional-ordering bias.

Fraud Typology	Count	% of Fraudulent	% of Total
Invoice Inflation	~34	19.3%	2.9%
Ghost Shipment	~42	23.9%	3.6%
Product Mismatch	~30	17.0%	2.6%
Product Diversion	~32	18.2%	2.7%
Shipment Delay	~38	21.6%	3.2%
Total Fraudulent	~176	100.0%	15.0%
Normal Transactions	1,000	N/A	85.0%

Table 2. Composition of the SCARA synthetic dataset by fraud typology, with counts and percentages from the generated dataset.

The fraud distribution reflects a random equal-probability selection across typologies during the injection phase. Ghost shipments appeared at the highest frequency of 23.9%, with shipment delays at 21.6%. Product Mismatch occurred at the lowest rate of 17.0%. These proportions arise from simulation randomness rather than deliberate weighting, and they match the observed pie-chart distributions.

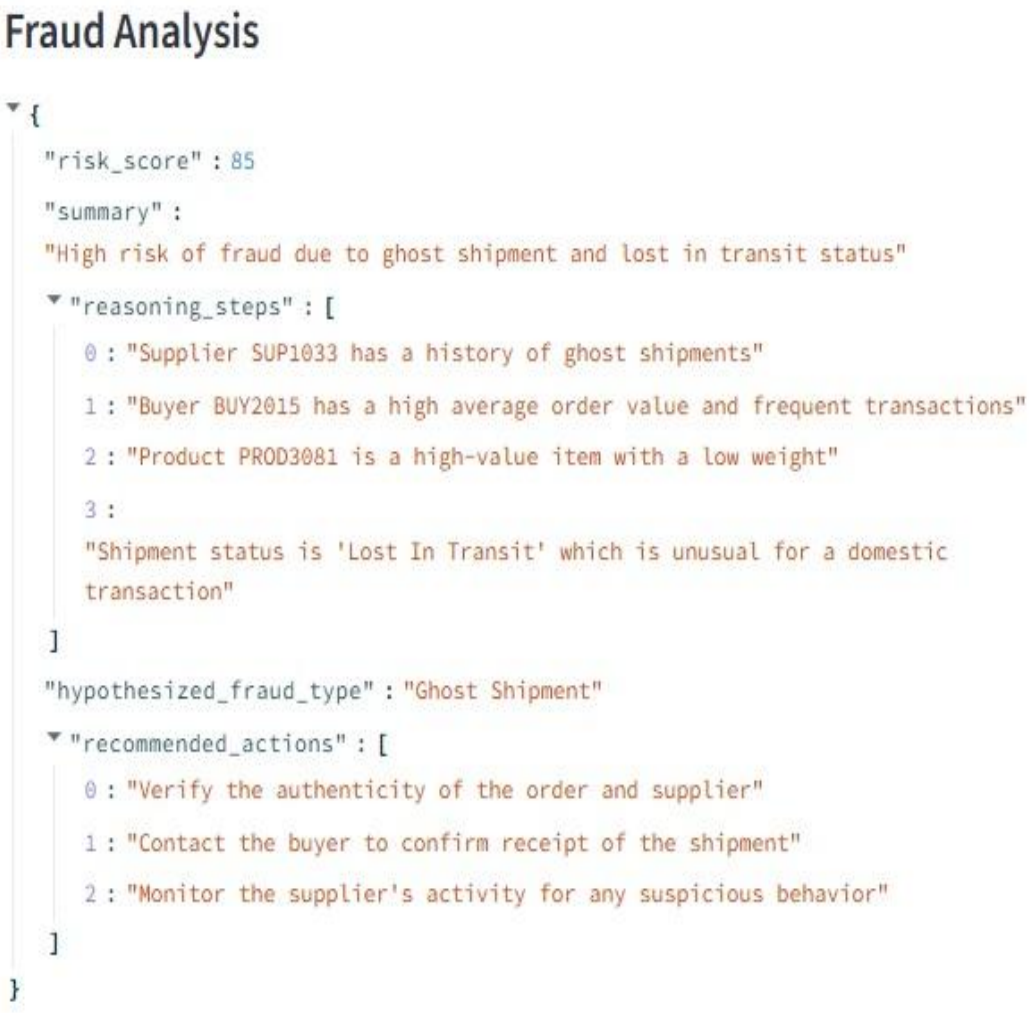

Figure 6. Fraud distribution across the 176 injected transactions: Ghost Shipment (23.9%), Shipment Delay (21.6%), Invoice Inflation (19.3%), Product Diversion (18.2%), and Product Mismatch (17.0%).

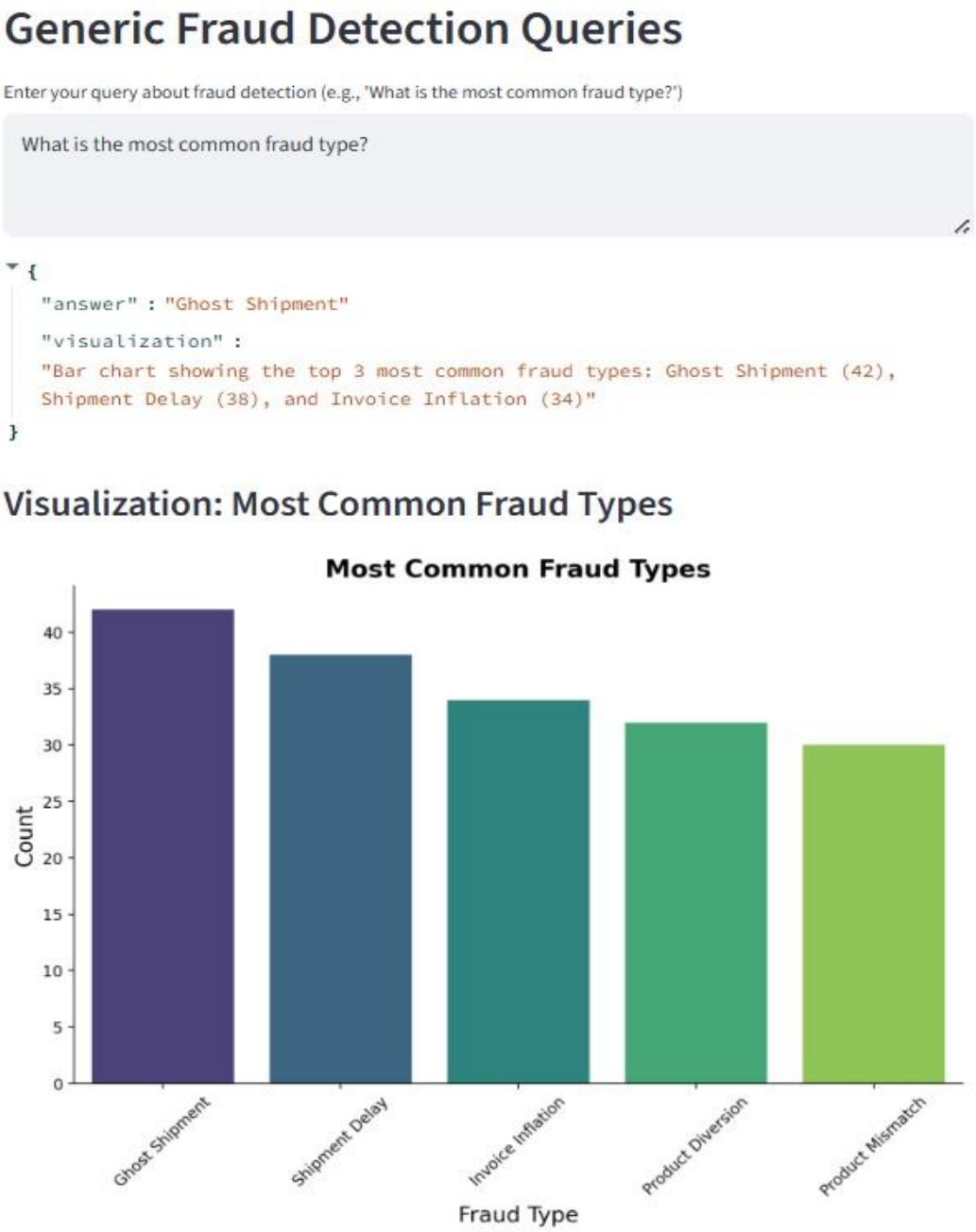

Figure 7. Product category distribution across all 1,176 transactions: Medical Supplies (22.2%), IT Hardware (22.1%), Automotive Parts (22.0%), Stationery (17.5%), and Office Furniture (16.2%).

4.2 LLM Agent Performance: Qualitative Assessment

4.2.1 Transaction Inspection and Ghost Shipment Detection

The transaction inspection interface allows an analyst to enter any order ID and receive both the raw transaction detail and SCARA risk report side by side. The screenshots below show a worked example using transaction ORD20022, which was injected as a Ghost Shipment. The data panel reveals the key signals: shipment_status is Lost In Transit, and both route_distance_km and transit_time_hours carry NaN values, indicating that no physical movement of goods occurred despite the recorded payment.

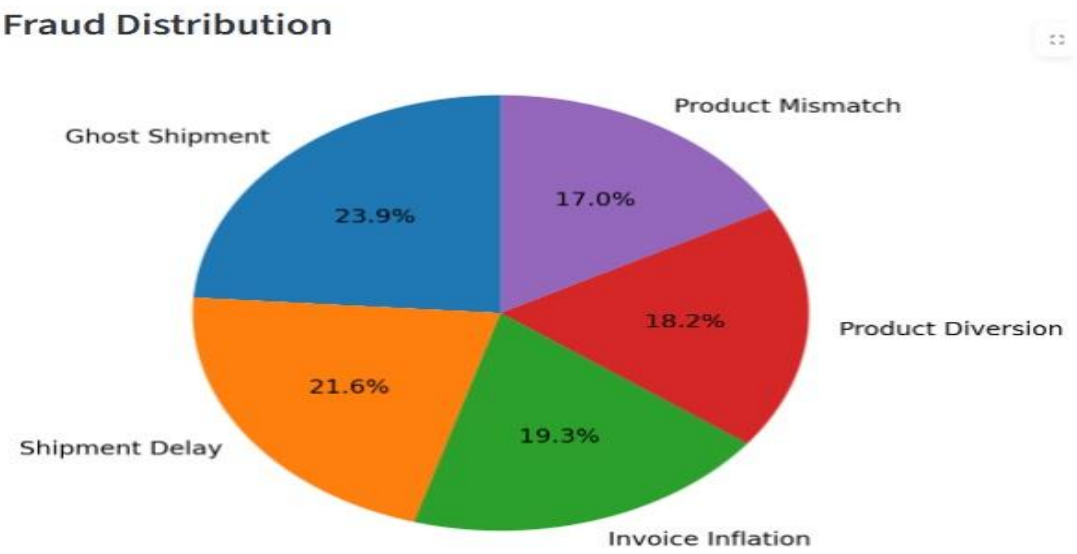

Figure 8. Transaction inspection panel: order ORD20022 showing Ghost Shipment indicators, including Lost In Transit status and NaN values for route distance and transit time.

The SCARA fraud analysis output for this transaction, shown in Figure 9, assigns a risk score of 85 (high) and correctly identifies the Ghost Shipment as the hypothesized fraud type. The reasoning chain walks through four observable signals: the supplier's transaction history, the buyer's profile, the product characteristics, and the critical observation that the Lost In Transit status is anomalous for a domestic transaction with no route data. The recommended actions focus on verifying order authenticity, contacting the buyer to confirm receipt, and monitoring the supplier for further suspicious activities.

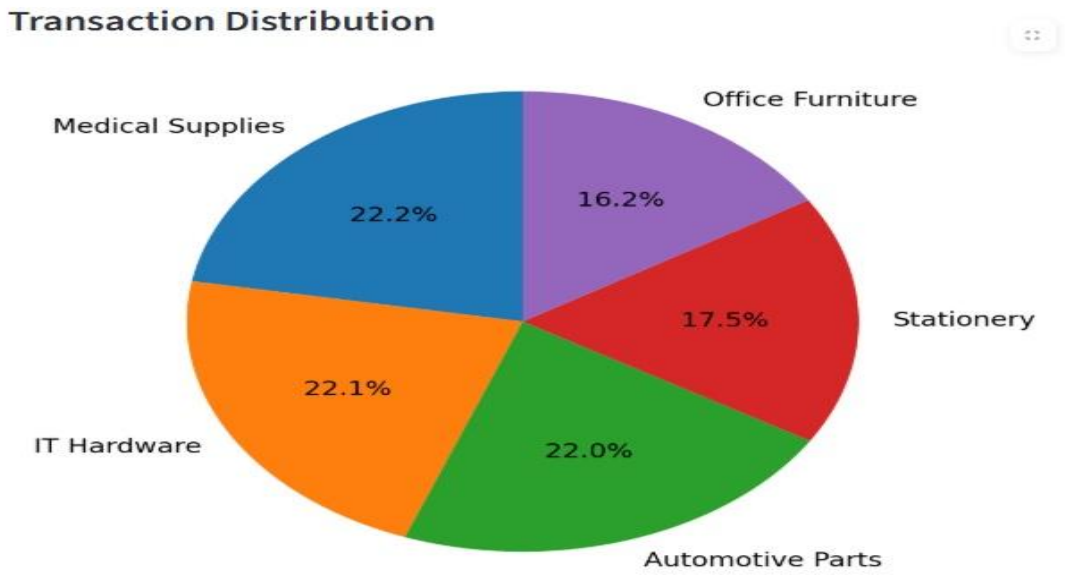

Figure 9. SCARA fraud analysis output for order ORD20022: risk score 85, Ghost Shipment hypothesis, four-step reasoning chain, and three recommended investigative actions.

4.2.2 Generic Fraud Queries and Visualisation

The generic query interface allows analysts to pose natural language questions at the aggregate level,and The example below shows the query "What is the most common fraud type?" generating both a structured JSON answer identifying Ghost Shipment at 42 occurrences and an automatically rendered Seaborn bar chart showing the full distribution across all five typologies. This combination of text and visual output is designed to serve the analytical and reporting needs of a real investigation team in the field.

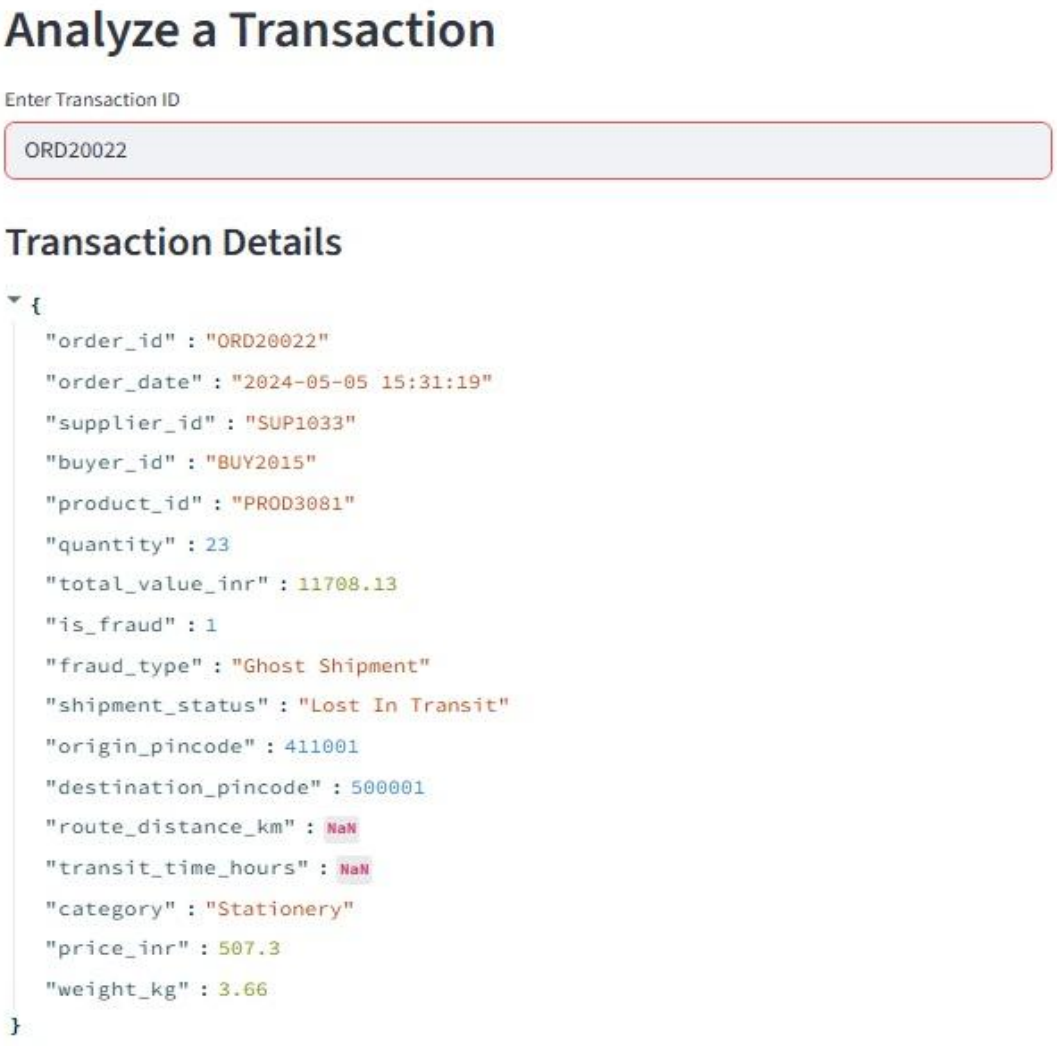

Figure 10. Generic fraud detection query interface: natural language query with JSON response identifying Ghost Shipment as the most common fraud type and auto-generated Seaborn bar chart.

4.2.3 Performance Across Fraud Typologies

Across ghost shipments, invoice inflation, and product mismatch cases, SCARA produces consistently confident and accurate reasoning. Ghost Shipment cases benefit from strong null-field signals and receive critical or high-risk scores in all observed instances. Invoice Inflation cases trigger clear ratio-based reasoning, where the agent computes the inflation multiple and compares it with the 150% threshold in the detection rules. Product Mismatch cases demonstrate sophisticated cross-field inference: the agent correlates a high invoice total against a stationery category product and correctly judges the combination as economically implausible within Indian government procurement norms.

Product Diversion cases are handled well when the diverted shipment status is present. Shipment Delay represents the most significant limitation: because the injection mechanism does not modify any observable transaction field beyond the fraud label, the agent has no quantitative signal to work from and tends to produce medium rather than High or Critical classifications. This gap is a concrete direction for future work and should be addressed by injecting anomalous transit time values that fall outside the statistically expected range for a given route distance.

4.3 Practical Implications for Procurement Governance

SCARA has immediate practical implications for procurement oversight agencies, internal audit units and anti-corruption task forces. By combining synthetic data generation with LLM-powered risk narration, the framework offers a low-cost pathway for agencies to train fraud analysts on realistic scenarios without exposing sensitive operational records, addressing the cold-start problem that has historically impeded fraud detection system development in public sector contexts. The structured JSON output is designed for downstream integration into existing case management platforms, allowing risk scores and recommended actions to flow into investigative workflows without manual transcription.

The human-in-the-loop design, where SCARA surfaces and explains anomalies but leaves the final adjudication to a credentialed analyst, aligns with the emerging governance frameworks for AI in high-stakes public sector applications. These include the Digital Personal Data Protection Act 2023, India's National AI Strategy, and the European Union AI Act's classification of procurement fraud detection as a high-risk AI application requiring mandatory human oversight. This design explicitly acknowledges the current LLM limitations in adversarial legal contexts while capturing meaningful efficiency gains in the upstream screening and prioritization phases of fraud investigation.

V. LIMITATIONS

The limitations of the current SCARA prototype should be clearly stated in this section. Although the synthetic dataset was carefully calibrated to Indian procurement norms, it could not fully reproduce the distributional complexity of real-world procurement data. It does not capture the temporal clustering of fraud around budget cycle boundaries, collusion networks linking multiple suppliers and buyers across procurement rounds, or the linguistic diversity of vendor documentation across India's 22 constitutionally recognized official languages.

The use of a single LLM without ensemble methods or explicit uncertainty quantification introduces reasoning biases that are specific to the model. The Shipment Delay typology is currently under-operationalized in the injection schema, resulting in a weaker detection for this category. The logistics model, while using the Haversine formula for great-circle distances, does not account for road network topology, state highway infrastructure quality, or seasonal disruptions such as monsoon flooding, which materially affect freight transit times in India.

VI. FUTURE WORK

Based on the present framework, five primary directions are proposed for future development. The fraud typology will be extended to include bid rigging, split-order manipulation to circumvent procurement thresholds, and advance payment fraud. Graph neural network-based supplier-buyer relationship modelling will be added to surface collusion networks that transactional analysis alone cannot detect (Wang et al., 2022; Li et al., 2021). SCARA will be benchmarked quantitatively against rule-based baselines, gradient-boosted classifiers, and autoencoder-based anomaly detectors on a held-out stratified test set. Multilingual document processing will be integrated to handle regional language procurement records, building on recent advances in multilingual LLMs (Ahuja et al., 2023). Finally, a prospective deployment study with an Indian state procurement authority will evaluate the real-world detection rates, analyst acceptance, and operational impact of SCARA-generated risk prioritization on the investigation throughput.

VII. CONCLUSION

This manuscript presents SCARA, an intelligent agent framework for fraud detection in Indian public-sector supply chains. By combining a statistically grounded synthetic data generation pipeline, a five-typology fraud injection engine, a Haversine-based geospatial

logistics simulation, and a structured LLM-powered risk assessment layer, SCARA delivers explainable and actionable fraud intelligence at the individual transaction-level. This represents a meaningful qualitative advance over conventional anomaly score approaches, which identify that something is suspicious without helping analysts understand why or what to investigate further. The Streamlit dashboard makes the system accessible to practitioners who may have no machine learning background, broadening the reach of AI-assisted procurement oversight well beyond specialist data science teams.

As governments face the dual challenge of expanding public services while fighting the corruption that undermines them, tools such as SCARA represent a practical convergence of artificial intelligence and accountable governance. The full codebase is openly released to encourage the research community to build upon, challenge, and extend this work toward more transparent, efficient, and trustworthy public-procurement systems.

REFERENCES

[1] Ahuja, S., Aggarwal, M., Gumma, D., Watts, I., Sathe, A., Diddee, H., & Sitaram, S. (2023). MEGA: Multilingual Evaluation of Generative AI Intelligence. In Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (pp. 4232-4248). Association for Computational Linguistics.

[2] An, J., and Cho, S. (2015). Variational autoencoder-based anomaly detection using reconstruction probability. Special Lecture on IE, 2(1), 1-18.

Anthropic. (2024). Claude 3 model card. Technical Report. Anthropic PBC (San Francisco, CA, USA).

Association of Certified Fraud Examiners (ACFE). (2022). Report to the nations: 2022 global study on occupational fraud and abuse. Austin, TX: ACFE Publications.

[3] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., and Amodei, D. (2020). Language models are few-shot learning models. Advances in Neural Information Processing Systems, 33, 1877-1901.

[4] Chen, T., and Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 785-794). ACM.

Comptroller and Auditor General of India (CAG). (2023). Annual Report on Compliance Audit of the Union Government (Civil), Report No. 14 of 2023. New Delhi: Government of India Press.

[5] Fiore, U., De Santis, A., Perla, F., Zanetti, P., Palmieri, F. (2019). Using generative adversarial networks to improve classification effectiveness in credit card fraud detection. Information Sciences, 479, 448-455.

[6] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., and Bengio, Y. (2014). Generative adversarial networks. Advances in Neural Information Processing Systems, 27, 2672-2680.

Google DeepMind. (2024). Gemini 1.5: Unlocking multimodal understanding across millions of context tokens. Technical Report. Google DeepMind, London.

[7] Kingma, D. P., and Welling, M. (2014). Auto-encoding variational Bayes. Proceedings of the 2nd International Conference on Learning Representations (ICLR). Banff, Canada.

[8] Krishnaswamy, R., & Mehra, S. (2021). Estimating the cost of procurement corruption in India: A district-level panel analysis. Economic and Political Weekly, 56(34), 40-48.

[9] Laptev, N., Yosinski, J., Li, L. E., and Smyl, S. (2017). Time-series extreme event forecasting with neural networks at Uber. Proceedings of the International Conference on Machine Learning, Time Series Workshop. Sydney, Australia.

[10] Li, Y., Chen, R., Xu, L., Huang, J., and Lyu, M. (2021). Fraud detection in supply chain networks using heterogeneous graph attention networks. ACM Transactions on Intelligent Systems and Technology, 12(4), Article 38, 1-22.

[11] Lopez-Rojas, E. A., and Axelsson, S. (2016). PaySim: A financial mobile money simulator for fraud detection. In Proceedings of the 28th European Modelling and Simulation Symposium (pp. 249-255). Larnaca, Cyprus.

Ministry of Finance, Government of India. (2023). Annual Public Expenditure Statistics report, 2022-23. Department of Expenditure, Ministry of Finance, New Delhi, India.

OpenAI. (2023). GPT-4 technical report. arXiv preprint arXiv:2303.08774. OpenAI (San Francisco, CA, USA).

[12] Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., and Gulin, A. (2018). CatBoost: Unbiased boosting with categorical features. Advances in Neural Information Processing Systems, 31, 6638-6648.

[13] Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Roziere, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. (2023). LLaMA: An open and efficient foundation language model. arXiv preprint arXiv:2302.13971.

[14] Wang, X., He, X., Cao, Y., Liu, M., and Chua, T. S. (2022). KGAT: Knowledge graph attention network for recommendation and fraud detection. ACM Transactions on Information Systems, 40(3), Article 58, 1-27.

[15] Wei, J., Wang, X., Schuurmans, D., Bosma, M., Chi, E., Le, Q., and Zhou, D. (2022). Chain-of-thought prompting elicits reasoning in large language models. Advances in Neural Information Processing Systems, 35, 24824-24837.

[16] Willard, B. T., & Louf, R. (2023). Efficient guided generation of large language models. arXiv preprint arXiv:2307.09702.

[17] Yao, S., Zhao, J., Yu, D., Du, N., Shafran, I., Narasimhan, K., and Cao, Y. (2023). ReAct: Synergizing reasoning and acting in language models. In Proceedings of the 11th International Conference on Learning Representations (ICLR). Kigali, Rwanda.