Roadmap for Building a Generative AI System to Translate Technical Names into Business-Understandable Terms

Introduction

In modern data-driven organizations, technical identifiers—such as database column names, API fields, and log keys—often lack the clarity needed for business users to interpret and leverage data assets effectively. As data volumes and complexity grow, bridging the gap between technical nomenclature and business semantics becomes essential for data governance, analytics, and AI adoption [1] [2] . Generative AI offers a promising solution: by leveraging schema metadata, sample values, business rules, and lineage context, it can automate the translation of cryptic technical names into meaningful, business-friendly terms. This report presents a comprehensive, actionable roadmap for designing, implementing, and maintaining such a system, integrating best practices from metadata management, prompt engineering, model selection, human-in-the-loop validation, workflow integration, and governance.

1. Overall Implementation Roadmap

The successful deployment of a generative AI system for technical-to-business name translation requires a structured, multi-stage approach. Below is a high-level roadmap, with each stage elaborated in subsequent sections:

- **Stage 1: Data Collection and Preparation**
    - Extract schema metadata
    - Profile sample values
    - Gather business rules and lineage context
- **Stage 2: Data Labeling and Ground Truth Creation**
    - Curate labeled datasets for training and evaluation
- **Stage 3: Model Selection and Context Engineering**
    - Choose between LLM prompting, fine-tuning, or retrieval-augmented generation (RAG)
    - Engineer context inputs for optimal model performance
- **Stage 4: Prompt Engineering**
    - Design templates, instructions, and examples for effective AI guidance
- **Stage 5: Model Evaluation and Validation**
    - Apply automated metrics and human-in-the-loop workflows
- **Stage 6: Workflow Integration**
    - Embed the system into data catalogs, documentation, and APIs
- **Stage 7: Governance and Maintenance**
    - Establish approval workflows, auditing, versioning, and continuous improvement
- **Stage 8: Security, Privacy, and Infrastructure**
    - Address PII handling, orchestration, monitoring, and scalability

Each stage is detailed below, with practical steps, tooling options, and industry examples.

2. Data Collection and Preparation

2.1 Schema Metadata Extraction

Objective: Gather comprehensive metadata about data assets, including table and column names, data types, constraints, and relationships.

Key Activities:

- Use automated tools (e.g., SchemaCrawler, Infometry, dbt, Apache Atlas) to extract schema details from databases, data lakes, and APIs [3] [4] [2] [5] .
- Capture technical names, data types, primary/foreign keys, and relationships.
- Document source systems, update frequencies, and ownership.

Best Practices:

- Standardize extraction processes across heterogeneous systems.
- Store metadata in a centralized repository or active metadata catalog for easy access and enrichment [1] [2] .

Industry Example: IBM Cloud Pak for Data uses automated metadata enrichment jobs to extract schema metadata and link technical assets to business terms.

2.2 Sample Values and Data Profiling

Objective: Profile actual data values to provide context for name translation and detect patterns, anomalies, or sensitive information.

Key Activities:

- Use profiling tools (e.g., Pandas, Talend Data Quality, Informatica, Great Expectations) to analyze sample values, distributions, and formats [6] [7] .
- Identify value ranges, common patterns (e.g., email formats, codes), and outliers.
- Detect and flag PII or sensitive data using automated classification and entity recognition [8] [9] .

Best Practices:

- Profile data at both column and cross-column levels to uncover semantic relationships.
- Integrate profiling results into metadata catalogs for downstream enrichment and governance [1] .

Industry Example: Google Dataplex Universal Catalog profiles BigQuery tables and uses Gemini-powered AI to generate contextual descriptions based on sample values [10] .

2.3 Business Rules and Lineage Context

Objective: Incorporate business logic, rules, and lineage information to inform accurate name translation.

Key Activities:

- Extract business rules from documentation, code, or legacy systems using rule extraction tools (e.g., IBM ADDI, Infosys BRE, custom static analysis) [11] [12] .
- Document how data is transformed, validated, or aggregated across pipelines.
- Capture lineage information to trace data flow from source to consumption, using tools like Atlan, Acceldata, or Apache Atlas [1] .

Best Practices:

- Maintain a business glossary and rule repository linked to technical assets.
- Use lineage-aware catalogs to visualize dependencies and impact analysis [1] .

Industry Example: Atlan integrates business rules and lineage tracking to support governance and semantic mapping in enterprise data estates [1] .

3. Data Labeling and Ground Truth Creation

3.1 Labeled Dataset Preparation

Objective: Create high-quality labeled datasets mapping technical names to business-understandable terms for model training and evaluation.

Key Activities:

- Curate existing mappings from business glossaries, documentation, or manual annotation.
- Use human-in-the-loop workflows to validate and enrich labels, leveraging domain experts and crowdsourcing platforms (e.g., Amazon Mechanical Turk, Scale AI) [13] [14] .
- Persist ground truth datasets in standardized formats (e.g., CSV, JSON, METS) with metadata for reproducibility and versioning [13] [14] .

Best Practices:

- Ensure diversity in labeled examples, covering edge cases, abbreviations, and ambiguous terms.
- Use idempotent dataset management to avoid duplication and maintain consistency [13] [14] .

Industry Example: TruLens and OCR-D provide frameworks for ground truth dataset persistence and metadata creation in AI workflows [13] [14] .

3.2 Annotation Guidelines and Quality Control

Objective: Define clear annotation guidelines to ensure consistency and accuracy in labeling.

Key Activities:

- Develop annotation rubrics specifying criteria for business term selection, description quality, and semantic alignment.
- Implement inter-annotator agreement checks and majority voting for expert validation.
- Track provenance and version history of labeled data for auditability [15] .

Best Practices:

- Encourage descriptive rather than prescriptive annotation to capture nuanced business meaning.
- Use feedback loops to refine guidelines based on annotation outcomes and model performance [16] .

4. Model Selection: LLM Prompting, Fine-Tuning, and RAG

4.1 Model Selection Criteria

Objective: Choose the optimal generative AI approach based on data availability, domain specificity, cost, and scalability.

Options:

Approach	Description	Pros	Cons	Best Use Cases
LLM Prompting	Use pre-trained LLMs (e.g., GPT-4o, Claude, Gemini) with structured prompts	Fast, low cost, flexible, no retraining	Limited domain adaptation, may hallucinate, prompt limits	Prototyping, general translation, rapid iteration
Fine-Tuning	Retrain LLMs on labeled data for domain-specific adaptation	High accuracy, deep customization, stable output	Expensive, requires ML expertise, retraining for changes	Regulated domains, high-stakes, custom logic
Retrieval-Augmented Gen	Combine LLMs with vector DBs for context retrieval (RAG)	Up-to-date facts, scalable, lower hallucination risk	Needs curated knowledge base, more infra complexity	Dynamic catalogs, evolving business rules
Prompt Tuning/PEFT	Lightweight adaptation via soft prompts or LoRA	Efficient, moderate customization, less infra overhead	Less deep than full fine-tuning, needs some labeled data	Multi-client, intermediate specialization

Best Practices:

- Start with prompt engineering for MVPs and rapid prototyping [17][18] .
- Progress to fine-tuning or PEFT for high-value, domain-specific tasks as data maturity improves [19][20][18] .
- Use RAG for dynamic, context-rich translation leveraging external knowledge bases and embeddings [21][22][23] .

Industry Example: Netflix fine-tuned GPT for metadata tagging, while Klarna used prompt engineering for rapid deployment across geographies [19] .

4.2 Context Engineering

Objective: Structure and manage the context window for LLMs to maximize translation accuracy.

Key Activities:

- Design context windows including schema metadata, sample values, business rules, and lineage information [24][25] .
- Use kernel/user context separation for persistent memory and dynamic message buffers.
- Integrate context retrieval tools (e.g., ChromaDB, pgvector, LangChain) for scalable RAG pipelines [21][22][16] .

Best Practices:

- Use memory blocks and file abstractions to manage long-term agent memory and context evolution [24][25] .
- Standardize context management via open specifications (e.g., LCWMS) for interoperability and scalability [25] .

---

5. Prompt Engineering: Templates, Instructions, and Examples

5.1 Designing Effective Prompts

Objective: Craft structured, precise prompts that guide the AI to generate accurate business terms.

Key Elements:

- **Purpose:** Clearly state the translation goal (e.g., "Convert technical column names to business-understandable terms").
- **Content Context:** Provide schema metadata, sample values, and business rules as input.
- **Target Keywords:** Specify important business terms or domain-specific language.
- **Audience and Tone:** Indicate the intended user (e.g., business analyst) and desired tone (formal, descriptive).
- **Platform:** Note where the output will be used (catalog, documentation, API) [26] .

Prompt Example:

```
 Given the following database column metadata:
- Column name: "CUST_ID"
- Data type: VARCHAR
- Sample values: ["12345", "67890"]
- Business rule: "Unique identifier for each customer in the CRM system"
Translate the technical column name into a business-understandable term, including a description suitable for business users.
```

Best Practices:

- Use few-shot examples and chain-of-thought prompting for complex mappings [18] .
- Iterate and refine prompts based on output quality and user feedback [17] .
- Maintain a prompt library and management tool for reuse and versioning.

Industry Example: Ingeniux AI Module uses prompt engineering for metadata generation, combining help text, field labels, and user prompts for context-rich outputs [26] .

5.2 Contextual and Platform-Specific Prompting

Objective: Tailor prompts to the specific context and platform requirements.

Key Activities:

- Incorporate platform-specific constraints (e.g., SEO, catalog search, documentation standards).
- Use cross-platform optimization prompts for multi-channel metadata generation.
- Leverage help text and schema labels to provide additional context for AI processing [26] .

Best Practices:

- Prioritize help text over prompts when conflicts arise, ensuring consistent metadata generation.
- Review and update prompt templates as business requirements evolve.

6. Retrieval and Knowledge Sources: Vector DBs, Embeddings, and RAG Pipelines

6.1 Building and Using Vector Databases

Objective: Store and retrieve semantic embeddings of technical and business terms for context-rich translation.

Key Activities:

- Encode schema metadata, sample values, and business terms as embeddings using models like SentenceTransformers, OpenAI Ada, or Cohere [21] [22] .
- Store embeddings in vector databases (e.g., ChromaDB, Pinecone, Weaviate) with metadata for filtering and retrieval.
- Use similarity search (cosine, dot product) to find relevant business terms or exemplars for translation [21] [22] .

Best Practices:

- Tune vector DB parameters (e.g., HNSW index) for optimal recall and latency [22] .
- Integrate with RAG pipelines to augment LLM prompts with retrieved context.

Industry Example: MetaSynth uses a multi-agent RAG framework to retrieve top-ranked exemplars and iteratively refine meta titles and descriptions for e-commerce platforms, improving CTR and engagement [23] .

6.2 Retrieval-Augmented Generation (RAG) Pipelines

Objective: Combine LLMs with external knowledge retrieval for accurate, context-aware translation.

Key Activities:

- Implement RAG pipelines using frameworks like LangChain, Azure OpenAI, or custom Python utilities [21] [16] .
- Retrieve relevant context (e.g., business glossary, prior mappings, documentation) and inject into LLM prompts.
- Use feedback loops to update embeddings and improve retrieval accuracy over time [16] .

Best Practices:

- Use embedding adapters for domain-specific optimization without full model retraining [22] .
- Monitor retrieval quality and update knowledge bases as business terms evolve.

7. Model Evaluation: Metrics, Human Validation, and Automated Checks

7.1 Automated Evaluation Metrics

Objective: Quantitatively assess the accuracy and relevance of generated business terms.

Key Metrics:

- **BLEU, ROUGE, NDCG, MRR:** Compare generated terms/descriptions to ground truth or reference mappings [27] [23] .
- **Semantic Similarity:** Use embedding-based metrics (e.g., BERTScore, cosine similarity) to measure alignment with business definitions [28] [27] .
- **Faithfulness, Relevancy, Context Recall:** Evaluate factual consistency and appropriateness in RAG pipelines (e.g., RAGAS metrics) [28] [27] .

Best Practices:

- Use LLM-based evaluators for scalable, explainable assessment, complemented by human review for critical cases [27] [23] .
- Track performance over time and across domains to identify areas for improvement.

7.2 Human-in-the-Loop Workflows and Validation Interfaces

Objective: Ensure interpretability, accuracy, and business relevance through expert review.

Key Activities:

- Implement multi-layer HITL validation architecture: automated flagging, human review, expert validation [29] .
- Use validation interfaces in data catalogs, dashboards, or custom UIs for reviewing and approving generated terms [29] .
- Route ambiguous or high-impact cases to domain experts for consensus validation.

Best Practices:

- Embrace expert disagreement and descriptive review to capture nuanced business meaning.
- Use majority voting, probabilistic modeling, and inter-annotator agreement for robust validation.

Industry Example: PepsiCo and Airbnb use HITL validation frameworks integrated with Airflow and Great Expectations for scalable, business-user-friendly data quality checks.

8. Integration into Existing Workflows: Data Catalogs, Documentation, and APIs

8.1 Data Catalog Integration

Objective: Embed generated business terms into active metadata catalogs for discovery, governance, and analytics.

Key Activities:

- Integrate with AI-powered data catalogs (e.g., Atlan, Acceldata, Alation, Collibra, Select Star) via open APIs and connectors [1] [30] [5] .
- Map technical assets to business terms, glossary entries, and lineage information.
- Enable semantic search, impact analysis, and policy automation using enriched metadata.

Best Practices:

- Use bidirectional sync to propagate tags, classifications, and context across systems [1] .
- Automate evidence pack generation for audit readiness and compliance [1] .

Industry Example: Atlan and Acceldata provide active metadata catalogs with automated business term mapping, lineage tracking, and governance integration [1] .

8.2 Documentation and API Integration

Objective: Publish business-understandable names and descriptions in documentation and expose via APIs.

Key Activities:

- Generate and update documentation (e.g., data dictionaries, API specs) with enriched business terms using templating engines (e.g., Mustache, FreeMarker) [3] [2] .
- Expose translation functionality via REST APIs for integration with data engineering pipelines, BI tools, and external systems.
- Use schema validation tools (e.g., Apicurio Registry) to ensure consistency and trustworthiness of metadata across services.

Best Practices:

- Automate documentation updates and versioning to reflect changes in business terms and rules.
- Provide API endpoints for on-demand translation and validation, supporting workflow automation.

9. Governance, Auditing, and Versioning of Generated Names

9.1 Governance Policies and Approval Workflows

Objective: Establish robust governance for the creation, review, and publication of business terms.

Key Activities:

- Define roles and permissions (e.g., Admin, Owner, Editor, Steward) for term assignment, review, and publishing [31] .
- Implement approval workflows for draft terms, including review, edit, and publish steps [1] .
- Track ownership, change history, and escalation paths for governance artifacts.

Best Practices:

- Centralize governance committee with cross-functional representation for policy setting and conflict resolution [32] [31] .
- Use SLAs and versioning policies to ensure timely updates and auditability of business terms.

Industry Example: IBM Cloud Pak for Data uses draft/publish workflows and role-based permissions for business term governance.

9.2 Auditing, Traceability, and Version Control

Objective: Maintain comprehensive audit trails and version history for compliance and operational efficiency.

Key Activities:

- Log all changes to business terms, including creation, modification, and deletion, with timestamps and actor attribution [15] [31] .
- Provide centralized attribute management for bulk updates, normalization, and deprecation of outdated terms [15] .
- Use metadata repositories and interoperability standards (e.g., ISO 19115, DCAT) for consistent auditability [31] [32] .

Best Practices:

- Automate audit report generation and evidence pack creation for regulatory reviews.
- Monitor governance KPIs (e.g., coverage, lineage completeness, compliance rate) to measure success [31] [1] .

10. Security, Privacy, and PII Handling in Prompts and Logs

10.1 PII Detection, Masking, and Anonymization

Objective: Protect sensitive information in prompts, logs, and generated outputs to ensure compliance with privacy regulations.

Key Activities:

- Use NER models and regex-based detection to identify and mask PII in schema metadata, sample values, and prompts [8] [9] .

- Implement anonymization pipelines with placeholder substitution and mapping for secure LLM interaction [8] .
- Apply encryption and secure computation for high-risk scenarios, leveraging privacy-preserving frameworks (e.g., CipherGPT, EmojiCrypt) [8] .

Best Practices:

- Treat external LLMs as untrusted entities; sanitize prompts before submission and restore original values post-processing [8] [9] .
- Maintain secure mapping dictionaries and audit trails for de-anonymization and compliance.

Industry Example: OWASP LLM Top 10 highlights sensitive information disclosure risks in both inputs and outputs; organizations use automated PII detection and masking to mitigate exposure [9] [8] .

10.2 Secure Logging, Access Control, and Compliance

Objective: Ensure secure handling of logs, metadata, and generated terms throughout the system lifecycle.

Key Activities:

- Implement role-based access control (RBAC/ABAC), encryption, and audit logging in metadata catalogs and vector DBs [22] [30] .
- Use policy-aware automation to enforce masking, retention, and access policies automatically.
- Monitor for prompt injection, anomalous behavior, and compliance violations using observability platforms (e.g., Monte Carlo, Sifflet) [30] .

Best Practices:

- Regularly review and update security policies to reflect evolving regulations and business needs.
- Provide transparency and explainability in AI decision-making for audit readiness [31] .

11. Tooling and Infrastructure: Orchestration, Pipelines, and Monitoring

11.1 Orchestration and Pipeline Automation

Objective: Automate data extraction, enrichment, validation, and deployment workflows for scalability and reliability.

Key Activities:

- Use orchestration tools (e.g., Apache Airflow, dbt, Dagster) to schedule and manage metadata extraction, profiling, and enrichment jobs [1] [33] .
- Integrate validation frameworks (e.g., Great Expectations, Airbnb Wall) for automated quality checks and HITL routing.
- Monitor pipeline health, schema drift, and data freshness using observability platforms and active metadata catalogs [1] .

Best Practices:

- Design metadata-driven orchestration for adaptive, self-healing pipelines that respond to schema changes and business logic updates [33] .
- Use event-driven automation to trigger governance actions, notifications, and remediation workflows [1] [33] .

Industry Example: Macy's Technology uses metadata-driven orchestration on GCP for scalable, intelligent data engineering pipelines [33] .

11.2 Monitoring, Cost, Latency, and Scalability

Objective: Ensure system performance, cost-effectiveness, and scalability for enterprise deployment.

Key Activities:

- Monitor model inference latency, API throughput, and resource utilization using dashboards and evidence packs [1] .
- Optimize vector DB parameters, batch sizes, and retrieval strategies for performance and cost [22] .
- Scale infrastructure horizontally using cloud-native deployment options (e.g., SaaS, VPC, on-prem) [22] .

Best Practices:

- Set KPIs (e.g., time-to-first-dataset, MTTR, audit prep hours) to measure impact and guide optimization [31] .
- Use feedback loops and generative feedback cycles to continuously improve model outputs and system efficiency [16] .

12. Open-Source and Commercial Tools and Vendors

12.1 Tool Comparison Table

Tool/Platform	Type	Key Features	Use Case
SchemaCrawler	Open-source	Schema extraction, diagrams, scripting	Metadata extraction
Infometry INFOFISCUS	Commercial	Automated metadata discovery, complexity analysis	ETL migration, profiling
Atlan	Commercial	Active metadata catalog, governance, lineage	Data catalog, governance
Acceldata	Commercial	AI data catalog, policy automation, lineage	Data catalog, observability
Great Expectations	Open-source	Validation, profiling, Airflow integration	Data quality, HITL validation
ChromaDB	Open-source	Vector DB, embeddings, RAG pipelines	Retrieval, context engineering
LangChain	Open-source	RAG pipeline, LLM orchestration	Retrieval-augmented generation
Apicurio Registry	Open-source	Schema validation, REST API, governance	Metadata validation, documentation
Alation, Collibra	Commercial	Data catalog, governance, AI search	Enterprise metadata management
Select Star	Commercial	Automated catalog, semantic models, lineage	AI readiness, analytics
Monte Carlo, Sifflet	Commercial	Observability, anomaly detection	Monitoring, compliance

Best Practices:

- Select tools based on integration capabilities, governance depth, explainability, and scalability needs [30] [5] .
- Combine open-source and commercial solutions for flexibility and cost optimization.

12.2 Industry Case Studies

- **IBM:** Uses generative AI for business term generation, draft/publish workflows, and role-based governance in Cloud Pak for Data [34] [35] .
- **Atlan:** Powers data governance, lineage, and business term mapping for clients like Austin Capital Bank, Kiwi.com, and Contentsquare [1] .
- **Databricks:** Integrates schema translation and semantic views for AI/BI efficiency, as discussed in Schemaster tech talks.
- **Virgin Media O2:** Implements Smart Metadata solution combining generative AI and expert crowdsourcing for scalable, governed metadata creation [10] .

13. Governance Policies: Approval Workflows, SLAs, and Ownership Roles

13.1 Policy Framework

Objective: Define clear policies for term creation, review, publishing, and maintenance.

Key Activities:

- Establish SLAs for term review and publishing, with escalation paths for urgent cases.
- Assign ownership roles (e.g., Data Steward, Business Owner, Governance Council) for accountability and decision-making [31] [32] .
- Document approval workflows and versioning policies for transparency and auditability.

Best Practices:

- Use federated governance for scalable, cross-domain management.
- Regularly review and update policies to reflect business and regulatory changes.

14. Maintenance and Continuous Improvement: Feedback Loops and Retraining

14.1 Feedback Loop Implementation

Objective: Continuously optimize model outputs and system performance through iterative feedback.

Key Activities:

- Collect user feedback on generated terms via catalog interfaces, dashboards, or direct annotation [16] .
- Use generative feedback loops to vectorize outputs, index in vector DBs, and adapt recommendations over time [16] .
- Retrain models or update prompt libraries based on feedback, error analysis, and evolving business needs [17] [16] .

Best Practices:

- Schedule regular audits and retraining cycles to maintain accuracy and relevance.
- Monitor analytics (e.g., engagement rates, similarity scores) to guide improvement efforts.

15. Summary Table: Roles and Responsibilities

Role	Responsibilities
Data Steward	Maintain metadata accuracy, review glossary terms
Business Owner	Define business value, approve term mappings
Data Custodian	Implement access controls, manage repositories
Metadata Admin	Oversee catalog platforms, monitor quality metrics
Governance Council	Set policies, approve standards, resolve conflicts
Domain Expert	Validate business relevance, contribute annotations
ML Engineer	Develop, fine-tune, and monitor AI models
Prompt Engineer	Design, test, and manage prompt libraries
HITL Validator	Review ambiguous cases, ensure consensus

16. Conclusion and Next Steps

Building a generative AI system for translating technical names into business-understandable terms is a multi-disciplinary endeavor, requiring robust data preparation, thoughtful model selection, precise prompt engineering, and rigorous governance. By following the roadmap outlined above, organizations can automate metadata enrichment, improve data discoverability, and empower business users with meaningful context. Integration with active metadata catalogs, documentation, and APIs ensures seamless adoption, while governance policies and feedback loops maintain quality and compliance over time. As AI and data ecosystems evolve, continuous improvement and adaptation will be key to sustaining value and trust.

Next Steps:

- Assemble a cross-functional implementation team with clear roles and responsibilities.
- Pilot the system on a representative subset of data assets, iterating on prompts, context engineering, and validation workflows.
- Scale integration across catalogs, documentation, and APIs, embedding governance and feedback mechanisms for ongoing optimization.

This roadmap synthesizes best practices and lessons from leading vendors, open-source tools, and industry case studies, providing a practical guide for organizations seeking to bridge the gap between technical data assets and business understanding in the age of generative AI.

References (35)

1 *What is Metadata Orchestration & Why It Matters in 2026.* https://atlan.com/know/metadata-orchestration/

2 *The Ultimate Guide to Data Catalog Architecture: Components ….* https://www.castordoc.com/data-strategy/the-ultimate-guide-to-data-catalog-architecture-components-integrations-and-best-practices

3 *SchemaCrawler - Free database schema discovery and comprehension tool.* https://www.schemacrawler.com/

4 *Infometry | Metadata Discovery Tool for Faster Data Migrations.* https://www.infometry.net/product/metadata-discovery-tool/

5 *Top 20 Data Catalog Tools for Analytics and AI Governance in 2025.* https://www.selectstar.com/resources/data-catalog-tools

6 *Mastering Data Profiling: A Step-by-Step Guide - codezup.com.* https://codezup.com/mastering-data-profiling-step-by-step-guide-to-extracting-insights-from-large-datasets/

7 *Data Profiling: 5 Essential Techniques and Tools for High-Quality Data.* https://www.dataexpertise.in/guide-to-data-profiling-techniques-tools/

8 *HANDLING CONFIDENTIAL DATA IN LLM PROMPTS.* https://mdu.diva-portal.org/smash/get/diva2:1980696/FULLTEXT01.pdf

9 *When Prompts Leak Secrets: The Hidden Risk in LLM Requests.* https://www.keysight.com/blogs/en/tech/nwvs/2025/08/04/pii-disclosure-in-user-request

10 *Generate metadata automatically in Google Data Cloud - Google Cloud Blog.* https://cloud.google.com/blog/products/data-analytics/generate-metadata-automatically-in-google-data-cloud

11 *Business rules extraction for mainframe applications - Infosys.* https://www.infosys.com/modernization/documents/business-rules-extraction.pdf

12 *Extracting Business Rules from Existing Systems.* https://www.bpminstitute.org/resources/articles/extracting-business-rules-existing-systems

13 *Persist Groundtruth Datasets - TruLens.* https://www.trulens.org/getting_started/quickstarts/groundtruth_dataset_persistence/

14 *GitHub - OCR-D/gt-repo-template: A template for creating a ground truth ….* https://github.com/OCR-D/gt-repo-template

15 *Enhancing Metadata Governance Traceability and Centralized Management ….* https://www.tetrascience.com/blog/enhancing-metadata-governance-traceability-and-centralized-management-for-labels

16 *Feedback Loops in GenAI with Azure Functions, Azure OpenAI and Neon ….* https://techcommunity.microsoft.com/blog/azuredevcommunityblog/feedback-loops-in-genai-with-azure-functions-azure-openai-and-neon-serverless-po/4407399

17 *Prompt Engineering vs Fine-Tuning for LLMs: Choosing the Right Approach.* https://dev.to/mikesays/prompt-engineering-vs-fine-tuning-for-llms-choosing-the-right-approach-56de

18 *Prompt Engineering vs Fine Tuning | Best LLM Strategy 2025.* https://dextralabs.com/blog/prompt-engineering-vs-fine-tuning/

19 *Fine-Tuning vs. Prompt Engineering: What Works Best in LLM ... - LinkedIn.* https://www.linkedin.com/pulse/fine-tuning-vs-prompt-engineering-what-works-best-llm-simran-jaiswal-jjgzf

20 *LLM Fine-Tuning vs Prompt Engineering for Consumer Products.* https://www.ijsat.org/papers/2025/2/3098.pdf

21 *Embeddings and Vector Databases With ChromaDB - Real Python.* https://realpython.com/chromadb-vector-database/

22 *Leveraging ChromaDB for Vector Embeddings - A Comprehensive Guide.* https://airbyte.com/data-engineering-resources/chroma-db-vector-embeddings

23 *MetaSynth: Multi-Agent Metadata Generation from Implicit Feedback in ....* https://arxiv.org/html/2510.01523

24 *Anatomy of a Context Window: A Guide to Context Engineering.* https://www.letta.com/blog/guide-to-context-engineering

25 *InventorSingh/llm-context-management-specifications - GitHub.* https://github.com/InventorSingh/llm-context-management-specifications

26 *Prompt Engineering for AI Metadata Generation.* https://support.ingeniux.com/docs/base/igx-cms/v10/topics/concept/ai-metadata-prompt-engineering.dita

27 *Evaluation metrics | Microsoft Learn.* https://learn.microsoft.com/en-us/ai/playbook/technology-guidance/generative-ai/working-with-llms/evaluation/list-of-eval-metrics

28 *List of available metrics - Ragas.* https://docs.ragas.io/en/stable/concepts/metrics/available_metrics/

29 *Maximizing Data Extraction Precision with Dual LLMs Integration and ....* https://techcommunity.microsoft.com/blog/azure-ai-foundry-blog/maximizing-data-extraction-precision-with-dual-llms-integration-and-human-in-the/4236728

30 *List of Top AI-Powered Data Catalog Software - Dec 2025 Reviews ....* https://www.softwareworld.co/ai-data-catalog-software/

31 *Metadata Management Best Practices: A Complete 2025 Guide.* https://www.ovaledge.com/blog/metadata-management-best-practices

32 *Metadata Framework for Governance: Complete Guide.* https://www.ovaledge.com/blog/metadata-framework

33 *Metadata-Driven Orchestration: The Future of Scalable Data Engineering ....* https://www.analyticsinsight.net/tech-news/metadata-driven-orchestration-the-future-of-scalable-data-engineering-on-gcp

34 *Detailed Business Model Of IBM 2026 | IIDE.* https://iide.co/case-studies/business-model-of-ibm/

35 *IBM Company Case Studies | Industry Case Study | Business, Management.* https://www.icmrindia.org/casestudies/Case_Studies_Company_Wise.asp?Company=IBM